Objavovanie znalostí v databázach

Ján Paralič

Košice
2003

Ing. Ján Paralič, PhD.
Katedra kybernetiky a umelej inteligencie
Fakulta elektrotechniky a informatiky
Technická univerzita v Košiciach
Jan.Paralic@tuke.sk

Edícia vedeckých spisov Fakulty elektrotechniky a informatiky TU Košice

Lektorovali: doc. RNDr. Gabriela Andrejková, CSc.
Ing. Kristína Machová, CSc.

ISBN ??-?????-??-?

Predslov

Najrôznejšie komerčné zariadenia a vedecké prístroje chŕlia denne neuveriteľné množstvá stále zložitejších dát. To isté platí o exponenciálne sa rozrastajúcom množstve dát publikovaných v rôznych typoch médií, ako napr. web. Možnosti analýzy takýchto množstiev dát už dávno presahujú ľudské možnosti, a tak súčasná doba je charakteristická množstvom elektronicky dostupných dát na jednej strane, ale často nedostatkom znalostí na strane druhej.

Túto medzeru sa snažia vyplniť metódy objavovania znalostí, ktorých snahou je extrahovať nové, platné a potenciálne užitočné znalosti z veľkých objemov dostupných dát. Toto spravidla nie je možné dosiahnuť plne automatickým postupom, ale do procesu vstupuje často človek, ktorý môže svojimi zásahmi dosiahnuť optimálne alebo takmer optimálne výsledky v podobe nových znalostí využiteľných pre zvýšenie ziskov z predaja, zaujímavé inovácie svojho produktu, odhaľovanie podvodov, podpore rozhodovania a pod.

Keďže rozsah poznatkov v oblasti objavovania znalostí v posledných 10-12 rokoch rýchlo narastá a jednou z vied, ktorá do tejto multidisciplinárnej oblasti výrazne prispieva výsledkami výskumu je umelá inteligencia, Katedra kybernetiky a umelej inteligencie otvorila v školskom roku 2001/2002 v študijnom odbore Umelá inteligencia predmet s rovnakým názvom, t.j. Objavovanie znalostí. Cieľom tohto predmetu je oboznámiť študentov s hlavnými poznatkami v tejto prudko sa rozvíjajúcej a aplikačne veľmi širokej oblasti.

Tento učebný text by mal študentom poskytnúť základný študijný materiál v slovenčine, keďže podľa mojich poznatkov takýto v súčasnej dobe nie je k dispozícii, ale poskytujú aj množstvo odkazov na ďalšie zaujímavé publikácie a zdroje v cudzojazyčnej literatúre. Skriptá môžu byť cenným zdrojom teoretických poznatkov pre odborníkov z praxe, zaoberajúcich sa problematikou „data mining“ a objavovania znalostí v širšom.

Okrem dostupnej literatúry a ďalších zdrojov autor pri písaní týchto skrípt vychádzal aj zo svojich skúseností získaných počas riešenia výskumných projektov GOAL[1] (On-line analýza geografických informácií, integrácia geoinformačných systémov a technológií dátových skladov), CLUE[2] (Vyhodnotenie metód zhlukovania) a TEXAS[2] (Analýza textov boli), ako zodpovedný riešiteľ a projekt Integrácia prostriedkov inteligentných technológií[3] ako riešiteľ.

Na tomto mieste by som chcel vyjadriť poďakovanie obom recenzentkám za starostlivé prečítanie rukopisu, opravu mnohých formálnych aj vecných chýb a za cenné pripomienky a námety, ktoré obohatili obsah predkladaného učebného textu.

V Košiciach, január 2003 autor

[1] INCO Copernicus projekt č. 977091 financovaný Európskou komisiou, riešený v rokoch 1998 – 2001
[2] Bilaterálne rakúsko-slovenské výskumné projekty financované Rakúskym ústavom pre východnú a juhovýchodnú Európu. CLUE bol riešený v rokoch 1999-2000 a následne TEXAS v rokoch 2001-2002
[3] VEGA projekt 1/5032/98 financovaný MŠ SR

Obsah

1 Proces objavovania znalostí v databázach

V prvej kapitole budú vysvetlené základné pojmy, najmä definícia procesu objavovania znalostí a stručná charakteristika jeho jednotlivých krokov. V ďalšom potom budú predstavené niektoré najčastejšie sa objavujúce aplikačné oblasti, ako aj konkrétne príklady systémov využívajúcich metódy objavovania znalostí v praxi.

1.1 Základné pojmy

Objavovanie znalostí v databázach (***KDD*** – Knowledge Discovery in Databases) [5], [9], [10] je ***proces*** (semi-)automatickej extrakcie znalostí z databáz. Extrahované znalosti musia byť:

- ***platné*** (v štatistickom zmysle)
- ***doposiaľ neznáme*** a
- ***potenciálne užitočné*** (pre dané použitie).

1.1.1 Niektoré základné vlastnosti procesu KDD

To, že objavovanie znalostí v databázach je *proces*, znamená, že sa skladá z viacerých krokov. Jednotlivé kroky procesu KDD budú stručne popísané v ďalšej časti. Aj keď ich postupnosť je pevne daná, realizácia samotného procesu už nie je len jednoduchý automatický prechod jednotlivými deterministickými krokmi, ale má dva podstatné atribúty.

Tento proces spravidla nie je možné plne automatizovať, resp. jeho plnou automatizáciou len ťažko dosiahneme optimálne výsledky v podobe skutočne cenných znalostí, ktoré môžu priniesť reálny zisk, resp. prínos. Z tohto dôvodu je účelná, ba priam nevyhnutná *asistencia človeka*, ktorý rozhoduje o výbere vhodných operácií, algoritmov a ich parametrov v rámci jednotlivých krokov procesu KDD. Naviac len človek môže s konečnou platnosťou rozhodnúť, ktorá z objavených znalostí je natoľko nová a užitočná, aby sa dala s úspechom aplikovať v skúmanej aplikácii.

Z vyššie uvedeného vyplýva, že tento proces je v rámci svojich jednotlivých krokov nedeterministický, s mnohými možnými alternatívnymi rozhodnutiami v každom zo svojich krokov. Výber alternatívnej operácie, algoritmu, alebo zmena jeho parametrov potom nutne vedú k iným výstupom, čo následne ovplyvňuje ďalšie kroky. Preto v rámci procesu KDD často dochádza k *iteráciám* v rámci jedného alebo viacerých jeho krokov s cieľom dosiahnuť čo najlepší výsledok.

Proces objavovania znalostí z databáz je preto ***iteratívny*** a ***interaktívny***.

Ďalšou významnou charakteristikou procesu KDD je jeho ***multidisciplinárnosť***. Už na prvých konferenciách venovaných tejto problematike[1] bola zjavná participácia vedcov z rôznych oblastí výskumu. Najsilnejšie zastúpené oblasti sú:

[1] V roku 1989 sa uskutočnil prvý medzinárodný workshop, od roku 1995 medzinárodná konferencia o KDD v USA, od roku 1997 potom aj Európska konferencia o KDD (PKDD) a Pacificko-ázijská konferencia o KDD (PAKDD) a pribúdajú aj ďalšie.

- *štatistika* (vhodná najmä na spracovanie numerických dát, testovanie hypotéz a pod.)
- *umelá inteligencia*, zastúpená najmä strojovým učením (prehľadávanie, zameranie na symbolické dáta)
- *databázové systémy* (škálovateľnosť na obrovské množstvá dát, nové dátové typy, integrácia s komerčnými databázovými systémami)

Ale významnú úlohu hrajú napr. aj rôzne techniky vizualizácie a pod.

1.1.2 „Data mining“

Je potrebné upozorniť na rôzne používanie niektorých s KDD súvisiacich pojmov. Najmä v komerčnej literatúre sa často používa skoro výhradne pojem „data mining“ (DM), čo by sme mohli preložiť ako *dolovanie dát*. Pritom sa často myslí celý proces vrátane získania a predspracovania dát, ako aj vyhodnotenia získaných znalostí a nielen aplikácia inteligentných metód na ich získanie, ako je to akceptované väčšinou vedeckej komunity v oblasti objavovania znalostí.

Okrem toho sa sporadicky vyskytujú aj ďalšie označenia, ako „knowledge mining from databases“, „knowledge extraction“, „data/pattern analysis“, „data archeology“ [10] a pod.

Uvedený pojem dolovanie dát má ešte jednu nevýhodu, a síce, že celkom nezodpovedá významovému rámcu slovesa *dolovať*, ktoré sa spája spravidla podstatným menom vyjadrujúcim cieľ tohto procesu (napr. dolovať zlato alebo dolovať železnú rudu), nie jeho prostriedok (nedoluje sa skala).

Preto v ďalšom texte bude anglický pojem „data mining“ (DM) prekladaný slovenským ekvivalentom ***dolovanie v dátach*** a budeme pod ním chápať iba jeden krok v procese KDD tak, ako je to akceptované väčšinou vedeckej komunity v tejto oblasti.

Názov skrípt je ešte všeobecnejší, keďže sa neobmedzuje len na objavovanie znalostí z databáz, aj keď tejto problematike je venovaná väčšia časť skrípt, ale pojednáva aj o objavovaní znalostí z textov.

1.2 Jednotlivé kroky procesu objavovania znalostí v databázach

Jednotliví autori uvádzajú rôzne počty krokov v procese KDD, avšak možno v nich vysledovať jednoznačné súvislosti. Členenie jednotlivých krokov procesu KDD podľa piatich vybraných zdrojov ([5], [9], [12], [13] a [15]) je zhrnutých v nasledujúcej tabuľke v snahe zosúladiť a porovnať ich spoločné vlastnosti.

Z tabuľky je zrejmé, že jednotliví autori sa na proces KDD dívajú na rôznej úrovni podrobnosti, niektoré kroky tohto procesu sa u rôznych autorov prekrývajú, resp. sú pomenované trocha odlišnými názvami. Ako výsledok ich dôkladnejšej analýzy a snahy identifikovať naozaj kľúčové kroky možno nakoniec definovať nasledovné členenie procesu KDD (viď. Tabuľke 1.1 – okrem tu uvedených definícií boli zohľadnené aj [6] a [10]).

1. Definícia a analýza cieľovej úlohy
2. Získanie relevantných dát a ich porozumenie
3. Predspracovanie dát
4. Dolovanie v dátach
5. Vyhodnotenie nájdených vzorov a identifikácia znalostí

6. Aplikácia získaných znalostí a vyhodnotenie ich použitia

Simoudis [15]	Mannila [13]	Fayyad et al. [9]	Brachman a Anand [5]	Klösgen a Zytkow [12]
	pochopenie domény	štúdium aplikačnej domény	identifikácia úlohy	Definícia a analýza cieľovej úlohy
výber dát		tvorba cieľovej množiny dát	identifikácia dát	pochopenie a príprava dát
transformácia dát	príprava dátovej množiny	čistenie a predspracovanie dát	čistenie dát	
		redukcia a projekcia dát	vývoj modelu	
		výber funkcie dolovania		nastavenie vyhľadávania znalostí
DM	objavovanie vzorov - DM	výber DM algoritmu	Analýza dát	vyhľadávanie znalostí
		DM		zjemňovanie znalostí
interpretácia výsledkov	spracovanie objavených vzorov	interpretácia	generovanie výstupu	Aplikácia znalostí pri riešení úlohy
	využitie výsledkov v praxi	použitie objavených znalostí		nasadenie a praktické vyhodnotenie riešenia

Tabuľka 1.1 Analógia medzi rôznymi definíciami procesu KDD

1.2.1 Definícia a analýza cieľovej úlohy

Samotný proces KDD začína už ***analýzou konkrétnej reálnej úlohy***, pochopením ***existujúcich znalostí*** o danej doméne a starostlivým ***definovaním cieľa*** procesu, keďže má ísť o nájdenie doteraz neznámych a potenciálne užitočných znalostí. Príklady niektorých najčastejšie sa vyskytujúcich reálnych úloh, pri ktorých sa proces KDD oplatí použiť, sú uvedené v nasledujúcej časti.

1.2.2 Získanie relevantných dát a ich porozumenie

Keď je už zrejmé, čo by sa malo aplikáciou procesu KDD dosiahnuť, aký typ znalostí by sa mal objaviť, je potrebné sa zamerať na ***získanie relevantných dát***, t.j. napr. pre každý atribút v existujúcich databázach je potrebné rozhodnúť, či je relevantný pre danú úlohu, alebo nie. Rovnako je nutné sa zamyslieť nad tým, ktoré dáta mimo dostupných databáz danej organizácie (t.j. z externých zdrojov) by mohli byť užitočné pri hľadaní nových znalostí.

Dátam je potrebné tiež dobre ***porozumieť***, aby z nich extrahované znalosti mohli naozaj viesť k riešeniu cieľovej úlohy. To znamená, že je

potrebné rozhodnúť, či je k dispozícii dostatočný počet atribútov a dostatočne relevantná vzorka príkladov pre extrahovanie platných znalostí.

1.2.3 Predspracovanie dát

Predspracovanie dát zahŕňa podľa [10] okrem výberu relevantných dát ďalšie tri hlavné činnosti, a síce

- *Čistenie dát*, t.j. odstránenie zašumených a nekonzistentných dát. Až potom je možné relevantné dáta z rôznych zdrojov integrovať.
- *Integrácia dát* z viacerých, často heterogénnych zdrojov. Tu je potrebné riešiť aj otázku fyzického sprístupnenia jednotlivých dátových zdrojov, spôsob ich presunu a uloženia do integrovaného zdroja, či už je to existujúca komerčná databáza, externý súbor alebo dátový sklad.
- *Transformácia dát* do reprezentácie vhodnej pre daný cieľ KDD. Okrem cieľovej úlohy a typu znalostí, ktoré chceme objavovať má na výber transformačných operácií značný vplyv aj konkrétny algoritmus DM, ktorý sa vyberá v nasledujúcom kroku.

1.2.4 Dolovanie v dátach

Kľúčovým krokom procesu KDD je ***dolovanie v dátach (DM)***, teda aplikácia inteligentných metód pre získanie platných vzorov (patterns). V tejto fáze procesu teda ešte nehovoríme o znalostiach, ktoré vznikajú až vhodným výberom z vygenerovaných vzorov a ich aplikovaním v kontexte riešenej úlohy.

Medzi hlavné úlohy DM patria:

- ***Popisné dolovanie v dátach*** (*zovšeobecňovanie*, t.j. popis skupiny príbuzných objektov), resp. *diskriminácia* (t.j. porovnanie jednej alebo viacerých rozdielnych skupín objektov)
- ***Prediktívne dolovanie v dátach*** predstavuje kontrolované učenie, v rámci ktorého sa tvorí model pre *klasifikáciu* (v prípade ak je cieľový atribút symbolický), alebo *predikciu* (ak je cieľový atribút numerický) hodnôt cieľového atribútu pre nové objekty.
- Medzi ďalšie úlohy DM patria napr. ***zhlukovanie*** (identifikácia skupín podobných objektov) a ***asociačné pravidlá*** (hľadanie zaujímavých asociácií v spravidla veľkých transakčných databázach)

1.2.5 Vyhodnotenie nájdených vzorov a identifikácia znalostí

Výsledkom aplikácie algoritmov DM je ***množina vzorov*** (ich forma závisí na type zvolenej úlohy DM, ako aj na vybranom algoritme DM), ktoré je potrebné ***analyzovať*** a z ktorých je nevyhnutné ***vybrať*** tie, ktoré reprezentujú skutočne nové a potenciálne užitočné znalosti vedúce k riešeniu cieľovej úlohy.

Už samotné algoritmy DM robia selekciu medzi nájdenými vzormi a to s prihliadnutím na ich platnosť (napr. hľadajú len štatisticky dostatočne významne zastúpené vzory) a do istej miery užitočnosť (ak je možné formalizovať nejakú mierku užitočnosti ako napr. spoľahlivosť u asociačných pravidiel, alebo priemernú presnosť u klasifikačných a predikčných modelov).

Konečné slovo ale patrí človeku, expertovi na danú aplikačnú oblasť, ktorý posúdi skutočnú hodnotu objavených vzorov a ich využiteľnosť pri riešení

cieľovej úlohy. Ak kvalita získaných vzorov nie je uspokojivá, celý proces pokračuje ďalšou iteráciou. To môže znamenať návrat k predchádzajúcemu kroku (DM) a zmenu nastavených parametrov, resp. výber iného algoritmu, ale aj hlbšiu zmenu, napr. v rozsahu, alebo reprezentácii dát vstupujúcich do DM, t.j. návrat až ku kroku 3, alebo dokonca 2.

1.2.6 Aplikácia získaných znalostí a vyhodnotenie ich použitia

Ak je expert s výberom vzorov v predchádzajúcom kroku spokojný a rozhodne o účelnosti ich použitia pri riešení cieľovej úlohy, znamená to, že nájdené zdroje sa stali novými a potenciálne užitočnými znalosťami v problémovej doméne.

Avšak ich skutočnú hodnotu môže preukázať až ich využitie v praxi, preto do procesu KDD je zahrnutá aj táto, finálna fáza, predstavujúca akúsi reálnu spätnú väzbu celého procesu.

Najskôr je potrebné objavené znalosti ***pretransformovať na konkrétne riešenie*** a nasadiť ho do reálnej praxe. Toto sa potom s primeraným časovým odstupom ***vyhodnocuje***, aby sa vyčíslil konkrétny efekt riešenia navrhnutého pomocou procesu KDD.

1.3 Smerom k štandardizácii procesu KDD

Z predchádzajúcej časti je zrejmé, že existuje mnoho názorov na členenie a popis procesu KDD. Všetky tieto definície ([6], [9], [10], [12], [13], [15]) zdieľajú základné aspekty na vyššej úrovni, ale líšia sa do určitej miery na detailnejších úrovniach. Jedným z nebezpečenstiev takéhoto stavu je to, že každá organizácia sleduje svoj vlastný procesný model a systematické porovnanie a vyhodnotenie procesov a nástrojov pre objavovanie znalostí je len veľmi ťažko možné.

Preto sú logické snahy o štandardizáciu procesu KDD. Hlavnou aktivitou v tomto smere je iniciatíva **CRISP-DM** [4] (Cross Industry Standard Process for Data Mining). Výsledkom tejto iniciatívy je veľmi nádejný krok smerom k definícii štandardnej metodológie, ako aj počiatočné vodítko pri uskutočňovaní projektov objavovania znalostí.

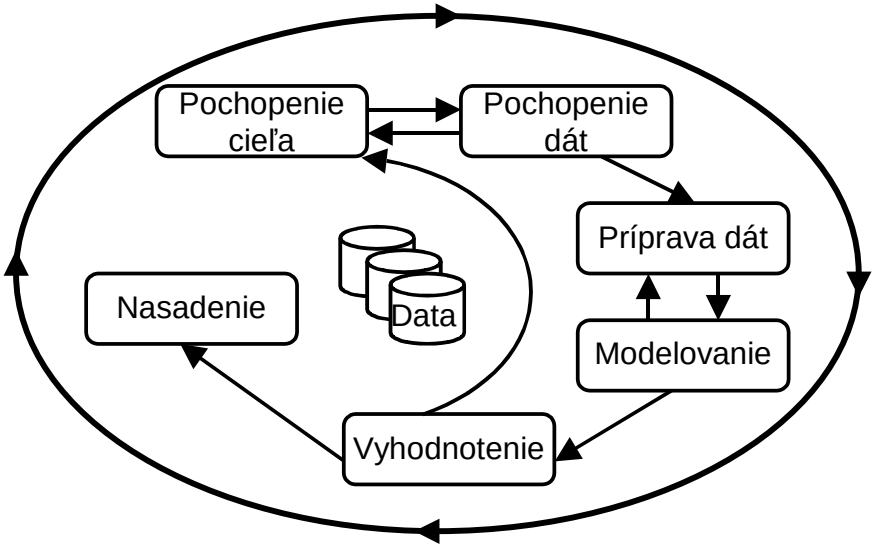
Pohľad CRISP-DM [14] na objavovanie znalostí a dolovanie v dátach pozostáva zo 6 fáz (viď. Obr. 1.1), medzi ktorými existujú úzke vzťahy a vždy sa vyžaduje prechod jedným aj druhým smerom medzi jednotlivými fázami (viď. [4]). Ide o nasledovných 6 fáz:

Fáza ***pochopenia (reálneho) problému*** sa zameriava na pochopenie obchodných alebo iných cieľov a požiadaviek a snaží sa ich pretransformovať na konkrétnu definíciu úlohy dolovania.

Fáza ***pochopenia dát*** začína počiatočným zberom dát, pokračuje aktivitami pre podrobné oboznámenie sa s dátami, zistenie ich kvality a charakteru, prípadne zistením podmnožiny dát zaujímavých pre ďalší výskum.

Fáza ***prípravy dát*** zahrňuje všetky aktivity potrebné pre vytvorenie množiny dát pre modelovanie. Operácie vykonávané v rámci prípravy dát väčšinou prebiehajú viackrát v nepredpísanom poradí. Patrí sem výber tabuliek, príkladov, atribútov, ako aj ich transformácia a čistenie.

Fáza ***modelovania*** je charakterizovaná výberom a aplikovaním nejakej metódy modelovania, nastavením a vyladením jej parametrov na optimálne hodnoty.



Obr. 1.1 CRISP-DM procesný model pre KDD

Keďže väčšina metód má špecifické požiadavky na formu dát, tu je nevyhnutná úzka interakcia s predchádzajúcou fázou prípravy dát.

Fáza ***vyhodnotenia***. V tejto etape sú už k dispozícii veľmi kvalitné modely z pohľadu dolovania v dátach, ale je potrebné ich vyhodnotiť z pohľadu stanovených obchodných alebo iných cieľov.

Fáza ***nasadenia*** získaných modelov môže byť pomerne jednoduchá, ak ide o vygenerovanie správy, ale môže to byť aj niečo zložitejšie, ako napr. implementácia opakovateľného procesu KDD pre danú aplikáciu.

V nasledujúcich častiach sú stručne popísané hlavné úlohy vykonávané v rámci jednotlivých fáz procesu KDD podľa štandardu CRISP-DM [14]. Pre každú úlohu sú uvedené jej vstupy a výstupy.

1.3.1 Pochopenie problému – 1. fáza

Táto fáza má splniť štyri hlavné úlohy. Prvou je ***stanovenie obchodných cieľov***. Zanedbanie tejto úlohy totiž môže veľmi ľahko viesť k namáhavému hľadaniu správnych odpovedi na zle formulované otázky. Výstupom tejto úlohy sú:

- *pozadie* – akékoľvek informácie o pozadí daného problému, ktoré sú známe na jeho začiatku a môžu byť eventuálne využité v procese KDD,
- *obchodné ciele* – popis primárnych cieľov z obchodného pohľadu,
- *kritériá obchodného úspechu* – veličiny, podľa ktorých je možné presne vyhodnotiť obchodný prínos projektu.

Druhou úlohou je ***ohodnotenie situácie***. Tu ide o detailné zhodnotenie dostupných zdrojov, ohraničení, predpokladov a iných faktorov, ktoré je potrebné brať do úvahy pri stanovení cieľov DM a tvorbe plánu projektu. K výstupom z tejto úlohy patrí:

- *inventár zdrojov* – zoznam všetkých dostupných zdrojov, a to personálnych, dátových, výpočtových a softvérových
- *požiadavky, predpoklady a ohraničenia* – ide o zoznam všetkých obmedzení projektu vrátane času ukončenia; zrozumiteľnosť a kvalita výsledkov, bezpečnosť, ale aj právne otázky, predpoklady o dátach, podmienky platnosti výsledkov a ohraničenia na dostupnosť zdrojov
- *možné riziká* – zoznam možných rizík a zodpovedajúcich akcií
- *terminológia* – ide o glosár použitej obchodnej ale aj DM terminológie
- *analýza nákladov a prínosov* – porovnáva náklady projektu a potenciálne prínosy pre činnosť firmy (organizácie)

Treťou úlohou je ***určenie cieľov dolovania v dátach***. Hlavnou snahou v rámci tejto úlohy je transformovať obchodné ciele na ciele DM. Obchodné ciele totiž používajú obchodnú terminológiu, zatiaľ čo ciele DM sú formulované v čisto technických pojmoch (napr. klasifikácia, zhlukovanie a pod.). Takže výstupmi sú:

- *ciele dolovania v dátach* a
- *kritériá úspechu* v technických pojmoch (typickým príkladom je požadovaná minimálna presnosť predikcie).

Poslednou úlohou prvej fázy procesu KDD podľa štandardu CRISP-DM je ***vytvoriť plán projektu***. Ide vlastne o *špecifikáciu plánu projektu*, ktorý povedie k splneniu cieľov DM, a tým aj splneniu pôvodne stanovených obchodných cieľov.

1.3.2 Pochopenie dát – 2. fáza

Aj druhá fáza procesu KDD podľa CRISP-DM pozostáva zo štyroch úloh. Prvou z nich je ***počiatočný zber dát***. Ide tu vlastne o sprístupnenie všetkých relevantných dát uvedených v inventári zdrojov (viď. vyššie). Výstupom tejto úlohy je *správa o počiatočnej množine dát* popisujúca metódy použité pre získanie dát a prípadné problémy, ktoré sa pritom vyskytli.

Druhou úlohou je ***popis dát***. Jej výsledkom je *správa popisujúca dáta* (ich formát, množstvo, označenia jednotlivých atribútov a rôzne ďalšie prehľadové vlastnosti (podrobnejšie viď v nasledujúcej kapitole venovanej dátam).

Verifikácia kvality dát pozostáva z kontroly kvality dát všímajúc si ich úplnosť, korektnosť, chýbajúce dáta a pod. Výsledná *správa o kvalite dát* uvádza všetky zistenia a diskutuje možné riešenia objavených problémov.

Prieskum dát zahŕňa už prvé analýzy dát pomocou rôznych vizualizačných techník, uvažuje distribúciu kľúčových atribútov, ich vzájomné vzťahy, vlastnosti výrazných podskupín dát a jednoduché štatistické analýzy. Všetky zistenia sa zhrnú do *výskumnej správy*, vrátane prvých zistení a hypotéz a ich vplyvu na zvyšok projektu.

1.3.3 Príprava dát – 3. fáza

Vo všeobecnosti má tretia fáza procesu KDD podľa CRISP-DM metodológie dva hlavné výstupy, a síce *množinu dát* pripravenú pre ďalšie fázy *a popis* tejto

množiny, ktorý charakterizuje vykonané prípravné operácie. Možno tu však identifikovať päť špecifickejších úloh.

Výber dát pokrýva manuálnu a automatickú selekciu atribútov, ale aj redukciu počtu hodnôt, napr. pomocou diskretizácie, a pod. Kritériá pre výber sú relevancia vzhľadom k stanoveným cieľom DM, technické ohraničenia i požiadavky na kvalitu. Výstupom je odôvodnenie výberu/vylúčenia daných dát.

Čistenie dát na základe správy o kvalite dát z predchádzajúcej fázy je úlohou zvýšiť kvalitu dát na požadovanú úroveň. Ide napr. o výber vhodnej podmnožiny dát, doplnenie chýbajúcich hodnôt a pod. Výsledná správa o čistení dát popisuje rozhodnutia a operácie na ich základe vykonané s dátami, ako aj diskusiu o ich možnom vplyve na výsledky analýz.

Konštrukcia dát zahrňuje napr. generovanie odvodených premenných, celkom nových záznamov alebo transformované hodnoty existujúcich premenných. Výstupmi sú potom všetky *odvodené premenné*, *vygenerované záznamy* a *transformované dáta*.

Integrácia dát sa snaží kombinovať informácie z viacerých tabuliek alebo záznamov pre generovanie nových záznamov alebo hodnôt. Výstupom sú *zlúčené dáta* (obsahujúce rôzne informácie o rovnakých objektoch z viacerých tabuliek) alebo *agregované dáta* (zahŕňajú operácie pre výpočet nových hodnôt sumarizáciou informácií z viacerých tabuliek alebo záznamov).

Formátovanie dát predstavuje najmä syntaktické modifikácie dát, ktoré nemenia ich význam, ale vyžaduje si ich modelovací nástroj. Typickými výstupmi sú *preusporiadané atribúty* alebo *záznamy*, resp. preformátované dáta.

1.3.4 Modelovanie – 4. fáza

Táto fáza procesu KDD podľa CRISP-DM zahŕňa opäť štyri úlohy.

Výber techniky modelovania, ktorého výstupmi sú *technika modelovania* (prípadne viac techník) a *špecifické predpoklady*, ktoré musia dáta spĺňať (napr. že sa nesmú vyskytnúť chýbajúce hodnoty, klasifikačný atribút musí byť symbolický a pod.)

Vygenerovať ***návrh testovania***, t.j. skôr ako sa vytvorí model je potrebné vygenerovať procedúru, ktorou sa bude testovať kvalita vygenerovaných modelov (napr. oddelenie trénovacej a testovacej množiny). Výsledný *návrh testov* teda obsahuje zamýšľaný plán trénovania, testovania a vyhodnotenia modelov.

Vybudovanie modelu spočíva v aplikácii vybranej techniky na pripravené dáta a vytvorenie jedného alebo viacerých modelov. Výstupom sú *použité nastavenia parametrov* danej techniky modelovania, spolu s odôvodnením ich výberu. Samotný *model* (modely) a jeho (ich) *popis*(y) obsahujúci očakávanú presnosť, robustnosť, ako aj možné nedostatky.

Ohodnotenie modelu spočíva v interpretovaní výsledkov modelovania v kontexte kritérií úspechu DM (stanovených v rámci prvej fázy) a navrhnutého testovania (vyššie). Typicky ide o použitie štatistických vyhodnotení a iných dostupných mechanizmov pre danú úlohu DM. Výsledné ohodnotenie modelov sumarizuje výsledky tejto úlohy a obsahuje rebríček vygenerovaných modelov usporiadaných podľa ich kvality. Výsledkom môže byť aj zmena nastavených parametrov a následná nová iterácia tejto úlohy.

1.3.5 Vyhodnotenie – 5. fáza

Prvou úlohou je ***vyhodnotenie výsledkov***. Zatiaľ čo ohodnotenie modelu v predchádzajúcej fáze interpretuje výsledky z pohľadu kritérií úspechu DM, vyhodnotenie v tejto predposlednej fáze procesu KDD podľa CRISP-DM vyhodnocuje model podľa pôvodných obchodných cieľov a kritérií úspechu. Výstupom je *celkové vyhodnotenie* s definitívnym stanoviskom, či projekt už naplnil pôvodne stanovené obchodné ciele.

Posúdenie procesu zhodnotí celý proces (nielen výsledný model). Cieľom je určiť, či existuje nejaký dôležitý faktor alebo úloha, ktorá bola zanedbaná. Výstupom sú *doporučenia ďalších aktivít*, alebo aj *generická procedúra* pre tvorbu relevantných modelov v budúcnosti.

Stanovenie ďalších krokov projektu. Ide o rozhodnutie alebo ukončiť projekt a prejsť do fázy nasadenia, alebo iniciovať ďalšie iterácie, alebo pripraviť nový DM projekt. Výstupom je *zoznam možných akcií* spolu s dôvodmi pre a proti každej uvedenej alternatívy. Záverom je vybraná jedna z nich – finálne *rozhodnutie*.

1.3.6 Nasadenie – 6. fáza

Prvou úlohou je ***vypracovať plán nasadenia*** výsledkov DM do praxe. Výstupom je teda *plán nasadenia* popisujúci stratégiu nasadenia.

Plán monitorovania a údržby má pomôcť vyhnúť sa zbytočne dlhým obdobiam nekorektného používania výsledkov DM. Ide o spôsob monitorovania zavádzania výsledkov DM do každodenného života a ako ho ďalej efektívne sledovať.

Záverečná správa na konci projektu môže obsahovať sumár projektu a skúseností, alebo finálnu prezentáciu výsledkov DM, alebo oboje.

Posúdenie celého projektu a zhodnotenie, čo bolo dobré a čo zlé, čo je potrebné zlepšiť. Výstupom je dokumentácia skúseností, obsahujúca napr. „pasce“, zavádzajúce postupy, tipy pre výber najvhodnejšej DM metódy v podobných situáciách a pod.

1.4 Typické aplikácie objavovania znalostí v databázach

1.4.1 Marketing

V tejto oblasti ide najčastejšie o analýzu databázových dát o zákazníkoch, napr. segmentácia trhu, t.j. identifikácia rôznych skupín zákazníkov (to umožňuje presnejšie zamerať určitú marketingovú kampaň, alebo vhodne nastaviť ponuku výpredajov a pod.), alebo predpovedanie budúceho správania sa zákazníkov (napr. identifikácia zákazníkov, ktorí sú náchylní zmeniť svojho predajcu).

Systém *Spotlight* [2] napríklad analyzuje dáta o predajoch zo supermarketov, nachádza signifikantné zmeny v predajných množstvách niektorého produktu a odhaľuje súvislosti medzi týmito zmenami a ich príčinami ako napríklad zmena ceny.

1.4.2 Individuálna reklama a elektronický obchod

Mnohé firmy ponúkajú pre svojich zákazníkov služby na internete zadarmo. Zákazníci dostanú pre takéto služby svoj osobný účet, cez ktorý svoj prístup na

internet realizujú. Takýto účet umožňuje firme vytvoriť si veľmi presný profil používateľa, ktorý umožňuje prispôsobiť reklamy, ktoré sa používateľovi na obrazovke zjavujú, jeho surfovaniu na webe, alebo jeho osobným údajom ako vek, pohlavie, alebo bydlisko.

Konkrétnym príkladom je *AltaVista FreeAccess* [1], ponúkajúci zákazníkovi prístup na internet zadarmo, ak sa registruje a zadá svoje osobné údaje. Každý jeho/jej prístup na internet je potom sledovaný. Pomocou techník objavovania znalostí je správanie sa zákazníkov na internete analyzované. Výstupy takejto analýzy sú na jednej strane zaujímavé pre firmy, ktoré chcú ponúkať svoje produkty cez AltaVista a môžu takto spoznať svojich potenciálnych zákazníkov a na druhej strane možno tieto znalosti využiť na to, aby sa zákazníkovi neprezentovali všetky reklamy, ale iba tie, ktoré pravdepodobne zodpovedajú zákazníkovým potrebám.

Iným príkladom je denník *New York Times* [11], ktorý ponúka prístup cez www zadarmo. Opäť sa ale musí používateľ takejto služby registrovať a zadať osobné údaje ako e-mail, vek, pohlavie, príjem a poštové smerovacie číslo. Za pomoci týchto údajov a informácií o pohybe na www je možné robiť predpovede, nakoľko je pravdepodobné že dotyčný si kúpi nejaký výrobok, alebo asociácie medzi tým, čo používatelia kupujú a aké články v denníku čítajú.

Amazon.com [3] využíva informácie o poštovom smerovacom čísle a emailovej adrese na vytváranie tzv. nákupných skupín (t.j. skupiny podobných zákazníkov ako napr. mesto Los Angeles, alebo Harvardská univerzita a pod.). Členom takejto skupiny potom posiela ponuky typických produktov, ktoré sa v danej skupine často kupujú.

1.4.3 Odhaľovanie podvodov

Odhaľovanie podvodov napr. pri bankových prevodoch, alebo operáciách s kreditnými kartami, ale aj pri telefonátoch v mobilných sieťach, je jednou z najvýznamnejších aplikácií objavovania znalostí. Len v Spojených štátoch napríklad ročne stoja „klonované podvody“ pri telefonátoch v mobilných sieťach tamojších operátorov a zákazníkov niekoľko sto miliónov dolárov!

NYNEX [7] napríklad vyvinul systém na rozpoznávanie podvodov pri mobilných telefonátoch. Tento systém si vytvára pre každého zákazníka profil jeho typického správania pri telefonovaní a pri významnej odchýlke od tohto správania generuje alarm. Tento systém využíva techniky objavovania znalostí ako sú generovanie pravidiel a neurónové siete.

1.4.4 Astronómia

Za pomoci teleskopov sa získava obrovské množstvo najrôznejších dát, ktoré nie je možné manuálne spracovať. Napr. v rámci projektu Palomar Observatory Sky Survey sa získali 3 TB obrazových dát obsahujúcich približne 2 miliardy astronomicky relevantných objektov.

Systém *SKYCAT* [8] najskôr vykonáva segmentáciu obrazov a určí pre každý nájdený objekt 40 rôznych atribútov. Tieto objekty sa potom za pomoci rozhodovacieho stromu automaticky klasifikujú do skupín (napr. rôzne typy hviezd, resp. galaxií), čo je základom pre ďalšiu (manuálnu) astronomickú analýzu. Systém SKYCAT je na jednej strane omnoho rýchlejší ako manuálna

klasifikácia a na druhej strane umožňuje klasifikovať aj veľmi vzdialené objekty, ktoré už nemožno manuálne klasifikovať.

1.5 Použitá literatúra

[1] AltaVista Free Access: http://microav.com/ (1999)

[2] Anand, T. and Kahn, G.: SPOTLIGHT: A Data Explanation System. Proc. of the 8th IEEE Conference on Applied AI, pp. 2 – 8 (1992)

[3] Beck, R.: Amazon.com Starts Speciality Service. *CBS Market Watch* (1999)

[4] CRISP-DM: http://www.crisp-dm.org/

[5] Brachman, R.J.; Anand, T.: *The Process of Knowledge Discovery in Databases*. In Advances in Knowledge Discovery & Data Mining, Fayyad, U.M. - Piatetsky-Shapiro, G. - Smyth, P. - Uthurusamy, R., Eds. AAAI/MIT Press, Cambridge, Massachusetts, (1996)

[6] Ester, M., Sander, J.: Knowledge Discovery in Databases – Techniken und Anwendungen. Springer Verlag, (2000)

[7] Fawcett, T., Provost, F.: Adaptive Fraud Detection. Data Mining and Knowledge Discovery, Vol. 1 (1997), pp. 291 – 316

[8] Fayyad, U.M., Haussler, D. and Stolorz, P.: KDD for Science Data Analysis: Issues and Examples. Proc. of the 2nd Int. Conference on Knowledge Discovery and Data Mining, pp. 50 – 56 (1996)

[9] Fayyad, U.M., Piatetsky-Shapiro, G., Smyth, P.: The KDD Process for Extracting Useful Knowledge from Volumes of Data. Comm. of the ACM, N o.11 (1996), pp. 27–34

[10] Han, J., Kamber, M.: Data Mining – Concepts and Techniques. Morgan Kaufmann Publishers, (2000)

[11] Himelstein, L., Hof, R.D. and Kunii, I.M.: The Information Gold Mine. Business Week, No. 30, http://www.businessweek.com/1999/99_30/b3639018.htm (1999)

[12] Klösgen, W. and Zytkow, J.M.: The Knowledge Discovery Process. In Handbook of Data Mining and Knowledge Discovery. Oxford University Press, pp. 10 – 21, (2002)

[13] Mannila, H.: *Methods and Problems in Data Mining.* In the proceedings of International Conference on Database Theory, Afrati, F. - Kolaitis, P., Delphi, Springer-Verlag, (1997)

[14] Reinartz, T.: Stages of the Discovery Process. In Handbook of Data Mining and Knowledge Discovery. Oxford University Press, pp. 185 – 192, (2002)

[15] Simoudis, E.: *Reality Check for Data Mining.* IEEE EXPERT, Vol.11, No.5, (1996)

2 Dáta

Druhá kapitola je venovaná samotným dátam, ako zdroju pre získavanie znalostí. Z pohľadu procesu KDD, predstavenému v predchádzajúcej kapitole, pôjde najmä o druhý a tretí krok, t.j. 1) Získanie relevantných dát a ich porozumenie, resp. 2) Predspracovanie dát. Ak sa na to pozrieme cez formujúci sa štandard CRISP-DM, tak ide o jeho druhú a tretiu fázu, t.j. Pochopenie dát a Príprava dát.

V kapitole sú najskôr uvedené základné súvislosti medzi dátami a reálnym svetom, zdroje chýb pri ich meraní a podmienky, ktoré musia byť splnené, aby sme mohli považovať nájdené znalosti v takýchto dátach za zodpovedajúce realite.

Samostatná časť je venovaná popisu najčastejších typov dát, s ktorými sa pri objavovaní znalostí môžeme stretnúť.

V ďalšej časti sú potom uvedené niektoré základné charakteristiky dát, ktoré je účelné o dátach zistiť hneď na začiatku s cieľom lepšieho porozumenia a prehľadu o dátach a ich využiteľnosti pre konkrétne ciele KDD.

Zvyšná časť kapitoly je potom venovaná jednotlivým operáciám predspracovania dát, ktoré sa najčastejšie používajú za účelom prípravy dátových vstupov pre samotné dolovanie v dátach.

2.1 Dáta a reálny svet

Dáta skúmame s cieľom objaviť znalosti o svete. K tomu, aby sa na základe dát objavené znalosti mohli považovať za hodnoverný obraz reálneho sveta, nutne predpokladáme nasledovné skutočnosti [10].

- Znalosti je možné objaviť v množine dát.
- Objavené znalosti sú užitočné a použiteľné vo svete.
- Dáta majú trvalý vzťah (reláciu) k svetu, z ktorého boli získané.
- Vzťahy vyskytujúce sa v dátach môžu byť zmysluplne vzťahované k fenoménom reálneho sveta.

Pritom **svet** pozostáva z **objektov** (nielen fyzických), ktoré vieme identifikovať. Objavovanie znalostí skúma **vzťahy medzi existujúcimi objektmi**. Objekty identifikujeme spravidla na základe množiny ich **vlastností**, ktoré je možné merať. Napríklad auto s piatimi vlastnosťami – značka, počet dverí, farba, počet valcov a počet miest.

Inštancia daného objektu teda predstavuje množinu konkrétnych, nameraných hodnôt vlastností za určitých overujúcich okolností. Napríklad auto typu Škoda-Fabia, ktoré má päť dverí, tmavomodrú farbu, štvorvalcový motor a kapacitu 5 miest je vlastne inštanciou objektu „auto“, pričom namerané hodnoty vyššie uvedených piatich atribútov sú postupne: Škoda-Fabia, 5, tmavomodrá, 4, 5. Tieto hodnoty majú určitý typ platnosti. V tomto prípade je to čas, kedy boli zistené (namerané). Časový okamžik kedy boli dáta zistené v tomto prípade tvorí overujúce okolnosti.

Merania pritom predpokladajú existenciu veličín, ktoré možno merať a prístrojov, voči ktorým sa merania kalibrujú [10]. Je potrebné si uvedomiť, že pri meraní vznikajú rôzne druhy chýb. Ich najčastejšími zdrojmi sú nasledovné:

- Nepresnosť kalibračného prístroja (napr. pravítko dlhšie ako je štandardná dĺžka)
- Veličina nie je korektne porovnaná s kalibrátorom (napr. posun pravítka a pod.)
- Nevyhnutná chyba presnosti (zaokrúhľovanie na merateľné jednotky)

Prvý zdroj chýb vyvoláva tzv. bias, čiže posun, systematickú chybu merania. Druhý a tretí zdroj chýb vyvolávajú nepresnosti rôznej veľkosti. Tieto pri opakovanom meraní vytvárajú akýsi zhluk bodov okolo toho „správneho bodu“, zodpovedajúceho reálnemu objektu sveta.

Okrem chýb meraní, s ktorými sa stretávame, je tu ešte jeden dôležitý fenomén, a to sú chýbajúce dáta. Je však potrebné rozlišovať medzi skutočne chýbajúcimi a prázdnymi hodnotami.

Chýbajúce hodnoty sú také, ktoré síce v dátach nie sú uvedené, ale určite existujú v reálnom svete, kde meranie prebiehalo. Príkladom môže byť atribút pohlavie, ktorý pri zbieraní údajov o zákazníkoch zabudol predavač zaznamenať pri predávaní tovaru.

Prázdne hodnoty prichádzajú do úvahy u premenných, kde nemožno očakávať reálnu hodnotu, resp. jej nevyplnenie nesie špecifickú informáciu. Ak je napríklad sledovanou premennou typ šalátu, ktorý si zamestnanec kúpi k obedu. To, že hodnota takýto atribút pre daný záznam nenadobúda žiadnu hodnotu neznamená že ide o chýbajúci hodnotu, ale že zamestnanec si šalát nekúpil. Takáto hodnota je prázdna a tiež nesie určitú informáciu.

2.2 Typy dát

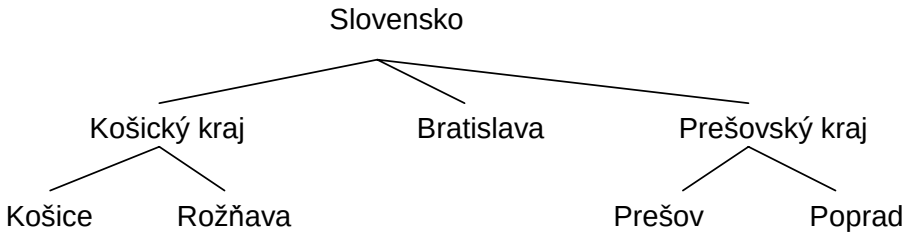
Dáta v databázach môžu mať veľmi rozmanité formy. Vo všeobecnosti možno hovoriť o numerických a symbolických entitách [1]. Prvá kategória dát, teda **numerické atribúty** spravidla predstavujú číselné údaje (napr. výška 185 cm), vektory čísel (číselné vyjadrenie RGB zložiek farebného odtieňa), alebo vo všeobecnosti mnohorozmerné polia čísel (satelitný snímok určitého územia v niekoľkých frekvenčných kanáloch). Pritom definičné obory hodnôt numerických atribútov môžu byť alebo spojité (ak ide o množinu reálnych, prípadne racionálnych čísel), alebo diskrétne (v prípade množiny celých čísel).

Symbolické atribúty sa používajú pre popis nejakých kvalitatívnych vlastností (napr. hodnoty atribútu viditeľnosť môžu byť výborná, dobrá, zhoršená, obmedzená, zlá a pod.).

Okrem toho existujú ešte aj **zložené typy dát**, t.j. atribúty ktoré tvoria nejakú hierarchiu konceptov (napr. miesto trvalého bydliska, viď. Obr. 2.1).

Najdôležitejšia otázka z pohľadu procesu KDD je, ako možno manipulovať s jednotlivými typmi dát. Obzvlášť zaujímavý je problém, ako porovnať dve hodnoty konkrétneho atribútu, napr. ako vyjadriť ich vzdialenosť.

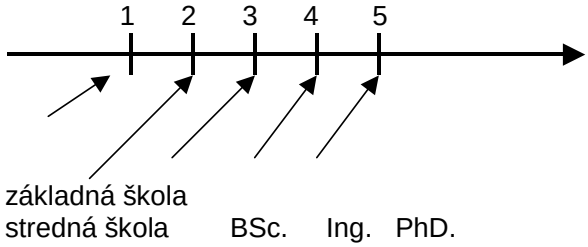
Pri narábaní s numerickými dátami je to pomerne priamočiare, nakoľko je možné použiť Euklidovskú, Minkowského, alebo prípadne v niektorých aplikáciách (textové dokumenty) kosínusovú vzdialenosť a pod. Problém vzniká pri symbolických a zložených typoch dát.



Obr. 2.1 Príklad zloženého dátového typu, stromová štruktúra atribútov (trvalé bydlisko)

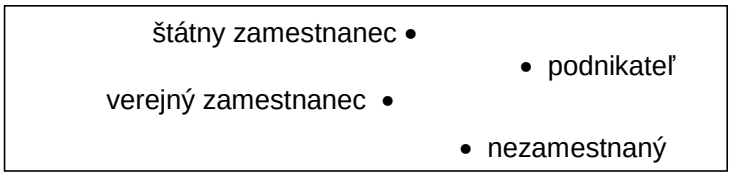
Niektoré definičné obory symbolických atribútov vytvárajú prirodzené usporiadanie, ide o tzv. **ordinálne atribúty** (viď. Obr. 2.2).

U iných definičných oborov symbolických atribútov takéto usporiadanie neexistuje. To sú tzv. **nominálne atribúty** (viď. Obr. 2.3).



Obr. 2.2 Ordinálny atribút vzdelanie.

Špeciálny prípadom nominálnych atribútov sú dvojhodnotové, t.j. **binárne atribúty** (s dvojicou hodnôt true, false, resp. áno, nie), ktoré môžu byť za určitých okolností tiež považované za numerické atribúty (reprezentované dvojicou hodnôt 1, 0).



Obr. 2.3 Nominálny atribút zamestnanie.

2.3 Základné charakteristiky dát

Skôr ako je možné vyjadriť sa ku kvalite a základným vlastnostiam určitej množiny dát, je potrebné sa s dátami bližšie oboznámiť. Keďže objemy dát,

s ktorými sa v procese KDD narába sú veľké, nemožno očakávať, že na objektívne oboznámenie sa s dátami bude stačiť napr. pohľad do databázy.

Pre prvé priblíženie dostupných dát najlepšie poslúžia rôzne štatistické charakteristiky, z ktorých najdôležitejšie uvedieme.

2.3.1 Charakteristiky pre jednotlivé atribúty

Najskôr je potrebné preskúmať jednotlivé atribúty (vlastnosti objektov) samostatne a až potom v ich vzájomných vzťahoch. V ďalšom je použité nasledovné označenie.

Databáza obsahuje ***n*** záznamov (údaje o n objektoch). Pre jednotlivé objekty boli namerané hodnoty x_1, x_2, ..., x_n atribútu ***X***. Charakteristiky, ktoré slúžia pre posúdenie rozloženia hodnôt sledovaných objektov pre danú vlastnosť, sú najmä tieto ([2], [3], [4]).

***Početnosť výskytu jednotlivých hodnôt a* atribútu *X* –** *f(a)*, t.j. koľkokrát sa daná hodnota *a* atribútu *X* vyskytla v databáze. Je možné použiť aj relatívnu frekvenciu výskytu hodnôt, t.j. *f(a)/n*.

Pre nominálne atribúty je to zároveň jediný spôsob ako zistiť charakter rozloženia jednotlivých hodnôt.

Táto charakteristika odhalí aj počet záznamov, v ktorých hodnota daného atribútu chýba (posúdiť, či ide o chýbajúce alebo prázdne hodnoty musí podľa charakteru atribútu a spôsobu jeho merania rozhodnúť človek).

Veľmi vhodnú vizualizáciu tejto charakteristiky poskytuje *histogram*.

Modus – x_{mod} je najčastejšie sa vyskytujúca hodnota atribútu v danej databáze.

Pre určenie priemernej (strednej) hodnoty numerického atribútu *X* v danej databáze je možné použiť nasledovné dve charakteristiky.

Aritmetický priemer – $\bar{x} = \frac{1}{n} \cdot \sum_{i=1}^{n} x_i$

Medián – x_{med} Ak predpokladáme, že jednotlivé namerané hodnoty x_1, x_2, ..., x_n atribútu *X* sú usporiadané vzostupne, potom:

pre *nepárne n* $x_{med} = x_{\frac{n+1}{2}}$

pre *párne n* $x_{med} = \frac{x_{\frac{n}{2}} + x_{\frac{n}{2}+1}}{2}$

Pre určenie rozptylu hodnôt numerického atribútu *X* okolo jeho priemernej (strednej) hodnoty je možné použiť nasledovné tri charakteristiky.

Rozptyl – $\bar{s}^2 = \frac{1}{n-1} . \sum_{i=1}^{n} (x_i - \bar{x})^2$ Rozptyl je vlastne štvorec priemernej odchýlky nameraných hodnôt atribútu *X* od ich aritmetického priemeru. Častejšie sa ale používa jeho odmocnina (čiže štandardná odchýlka), aby sa zachovala mierka pôvodného atribútu, ktorá umocnením na druhú do určitej miery stráca pre človeka výpovednú hodnotu.

Štandardná odchýlka – $\bar{s} = \sqrt{\frac{1}{n-1} \cdot \sum_{i=1}^{n} (x_i - \bar{x})^2}$ Štandardná odchýlka, prípadne variancia sa používajú v kombinácii s aritmetickým priemerom, voči ktorému vlastne určujú rozptyl hodnôt.

Percentily zase slúžia ako analógia štandardnej odchýlky pre medián. Medián je taká hodnota atribútu *X* v databáze, pre ktorú platí, že 50% ostatných hodnôt tohto atribútu leží pod touto hodnotou. *K*-ty percentil znamená takú hodnotu x_K atribútu *X*, pre ktorú *K* percent ostatných hodnôt tohto atribútu v databáze je menších ako x_K.

Pre *K*=25 (označovaný tiež Q_1) a *K*=75 (označovaný tiež Q_3) hovoríme o prvom, resp. treťom kvartile. Druhý kvartil je teda medián.

Okrem uvedených charakteristík existujú aj ďalšie, napríklad rôzne charakteristiky šikmosti a špicatosti.

Charakteristiky **šikmosti** udávajú, či sú hodnoty okolo zvoleného stredu rozložené súmerne, alebo je rozdelenie hodnôt zošikmené na jednu alebo druhú stranu (pri pohľade napr. na histogram). Všetky charakteristiky šikmosti nejakým spôsobom využívajú vzťah medzi aritmetickým priemerom, mediánom a modusom. Presnejšie vyjadrenie týchto veličín je možné nájsť v [4].

Charakteristiky **špicatosti** udávajú, aký priebeh má graf rozdelenia hodnôt okolo zvoleného stredu rozdelenia. Čím je rozdelenie špicatejšie, tým sú hodnoty viac sústredené okolo daného stredu rozdelenia (podrobnejšie, viď [4]).

2.3.2 Závislosť medzi jednotlivými atribútmi

Pri skúmaní závislosti medzi dvoma atribútmi sa používajú u symbolických atribútov tzv. kontingenčné tabuľky a u numerických rôzne korelačné koeficienty.

Nech *X* a *Y* sú dva symbolické (resp. diskrétne) atribúty s hodnotami x_i, $1 \le i \le k$, a y_j, $1 \le j \le m$. $h_{ij} = h(x_i, y_j)$ označuje absolútnu početnosť výskytov kombinácie hodnôt (x_i, y_j). Okrajové početnosti výskytov hodnôt atribútov *X*, resp. *Y* sú definované nasledovne: $h_{\bullet j} = \sum_{i=1}^{k} h_{ij}$, resp. $h_{i\bullet} = \sum_{j=1}^{m} h_{ij}$.

Kontingenčná tabuľka pre atribúty *X* a *Y* reprezentuje absolútne početnosti všetkých kombinácií hodnôt (x_i, y_j) a všetky okrajové početnosti *X* a *Y* (viď. Tabuľka 2.1).

y_i / x_i	y_1	...	y_m	
x_1	h_{11}	...	h_{1m}	$h_{1.}$
x_2	h_{21}	...	h_{2m}	$h_{2.}$
...	...		...	...
x_k	h_{k1}	...	h_{km}	$h_{k.}$
	$h_{.j}$	...	$h_{.m}$	n

Tabuľka 2.1 Kontingenčná tabuľka pre atribúty *X* a *Y*.

Ak sú atribúty *X* a *Y* úplne nezávislé, potom platí, že $\frac{h_{ij}}{n} = \frac{h_{i\bullet}}{n} \cdot \frac{h_{\bullet j}}{n}$, t.j. $h_{ij} = \frac{h_{i\bullet} \cdot h_{\bullet j}}{n}$. Rozdiel medzi touto nezávislosť predpokladajúcou a skutočne pozorovanou hodnotou h_{ij} poskytuje mierku pre závislosť atribútov *X* a *Y*. Používa sa pritom **χ^2-test nezávislosti** [5]. Pri tomto teste sa najprv vypočítava globálna charakteristika *G* veľkosti rozdielov pozorovaných a hypotetických (prípad nezávislosti *X* a *Y*) združených početností ako súčet relatívnych štvorcov odchýliek: $G = \sum_{i=1}^{k} \sum_{j=1}^{m} \frac{(h_{ij} - \frac{h_{i\bullet} \cdot h_{\bullet j}}{n})^2}{\frac{h_{i\bullet} \cdot h_{\bullet j}}{n}}$. *G* nadobúda hodnotu z intervalu [0, *n.h*], kde *h* je menšie z čísel *k*-1 a *m*-1. Testovaná hypotéza o nezávislosti atribútov *X* a *Y* sa potom zamietne pri extrémne vysokých hodnotách testovacieho kritéria.

Pri platnosti hypotézy o nezávislosti a za predpokladu, že všetky hypotetické početnosti sú väčšie ako 5, má testovacie kritérium *G* približne rozdelenie χ^2 s (*k*-1)(*m*-1) stupňami voľnosti. Na hladine významnosti α sa potom hypotéza o nezávislosti zamieta, ak *G* > $\chi^2_{1-\alpha}$[(*k*-1)(*m*-1)], t.j. ak je hodnota testovacieho kritéria *G* väčšia než 100(1-α)%-ný kvantil rozdelenia χ^2 s (*k*-1)(*m*-1) stupňami voľnosti. Príslušné hodnoty jednotlivých kvantilov pre rozdelenie pravdepodobnosti χ^2 s daným stupňom voľnosti možno nájsť v štatistických tabuľkách.

Príklad 2.1

Úlohou je zistiť mieru závislosti atribútu vzdelanie a stupeň nezamestnanosti, ak je daná nasledovná kontingenčná tabuľka.

	Strednodobo nezamestnaný	Dlhodobo nezamestnaný	
Bez vzdelania	19	18	37
Vzdelanie	43	20	63
	62	38	100

Tabuľka 2.2 Príklad kontingenčnej tabuľky atribútov vzdelanie a stupeň nezamestnanosti.

Porovnajme napríklad skutočné hodnoty početností h_{11} a h_{12} s tými, ktoré by mali byť za predpokladu, že dané dva atribúty by boli nezávislé.

$$19 = h_{11} \neq \frac{h_{1\bullet} \cdot h_{\bullet 1}}{n} = \frac{37 \cdot 62}{100} = 22{,}94 \qquad 18 = h_{12} \neq \frac{h_{1\bullet} \cdot h_{\bullet 2}}{n} = \frac{37 \cdot 38}{100} = 14{,}06$$

Dosadením do hore uvedeného vzťahu možno získať hodnotu $G = \frac{(19-22{,}94)^2}{22{,}94} + \frac{(18-14{,}06)^2}{14{,}06} + \frac{(43-39{,}06)^2}{39{,}06} + \frac{(20-23{,}94)^2}{23{,}94} = 2{,}83$. Túto je ešte potrebné následne porovnať s príslušným kvantilom χ^2 rozdelenia s 1 stupňom voľnosti, nakoľko v tomto prípade *k* = 2 aj *m* = 2, t.j. (*k*-1)(*m*-1) = 1. Z tabuľky

kvantilov rozdelenia χ^2 [5] vyplýva, že ak zvolíme obvyklú 5%-nú hladinu významnosti (α = 0,05), bude kritickou hodnotou kvantilu rozdelenia χ^2 s 1 stupňom voľnosti hodnota 3,84. Nakoľko *G* < 3,84, na 5%-nej hladine významnosti nemožno zamietnuť hypotézu o nezávislosti atribútov vzdelanie a stupeň nezamestnanosti. Avšak na 10%-nej hladine významnosti (α = 0,1), bude kritickou hodnotou kvantilu rozdelenia χ^2 s 1 stupňom voľnosti 2,71. Keďže teraz je splnená podmienka G > 2,71 pre zamietnutie hypotézy o nezávislosti uvedených atribútov, s 10%-ným rizikom omylu možno povedať, že stupeň nezamestnanosti závisí od najvyššieho dosiahnutého vzdelania.

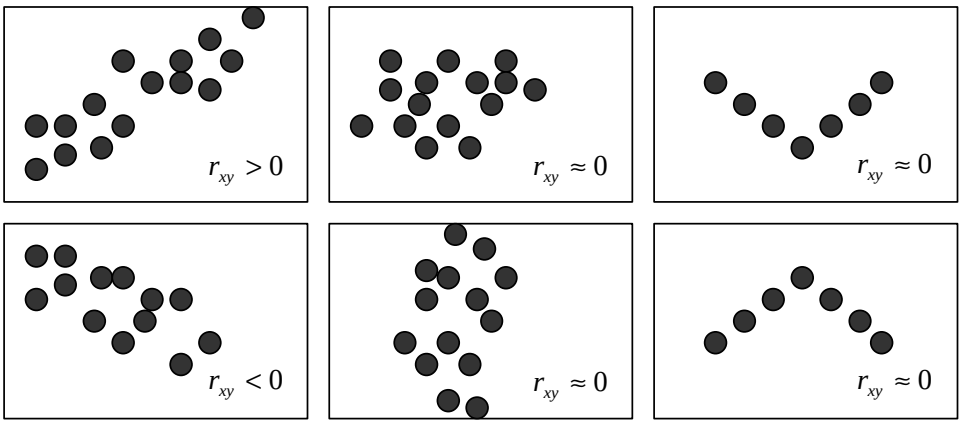
Nech majú skúmané atribúty *X* a *Y* spojité numerické hodnoty x_i a y_j. Nech $\bar{x}, \bar{y}$ sú aritmetické priemery ich hodnôt. (Empirický) korelačný koeficient r_{XY} je definovaný nasledovne.

$$r_{XY} = \frac{\sum_{i=1}^{n}(x_i - \bar{x}) \cdot (y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2 \cdot \sum_{i=1}^{n}(y_i - \bar{y})^2}}, r_{XY} \in [-1,1]$$

Sila korelácie medzi atribútmi *X* a *Y* sa potom hodnotí nasledovne: ak $|r_{XY}| < 0{,}5$ hovoríme o slabej korelácii, ak

$0{,}5 \le |r_{XY}| < 0{,}8$ ide o strednú koreláciu a ak

$0{,}8 \le |r_{XY}|$ ide o silnú koreláciu medzi atribútmi *X* a *Y*.



Obr. 2.4 Rôzne rozdelenia hodnôt atribútov *X*, *Y* s korelačnými koeficientmi

Príklad 2.2

Na nasledujúcom obrázku sú pre názornosť uvedené rôzne rozdelenia hodnôt dvojice atribútov *X* a *Y* a ich zodpovedajúci korelačný koeficient.

Silná korelácia medzi dvoma atribútmi neznamená nutne kauzálny vzťah medzi nimi. Môže existovať len tzv. zdanlivá korelácia (nepodložená skutočným

obsahom), ktorá môže napríklad vzniknúť tým, že bol prehliadnutý iný, tretí atribút, ktorý je silne korelovaný s oboma pôvodnými atribútmi.

Príklad 2.3

Nech atribút *X* znamená slovnú zásobu dieťaťa a *Y* jeho výšku v cm. Nech databáza obsahuje tieto príklady:

x_i:	37	30	20	28	35
y_i:	130	112	108	114	136

Pre uvedenú množinu objektov vychádza $r_{XY} = 0{,}863$, čo znamená silnú koreláciu medzi atribútmi *X* a *Y*. Ale pritom existuje aj premenná *Z*, ktorej hodnoty v jednotlivých prípadoch sú nasledovné:

z_i:	12	7	6	7	13

Pritom vychádza $r_{XZ} = 0{,}868$ a $r_{YZ} = 0{,}996$, t.j. atribút *Z* je silnejšie korelovaný tak s *X*, ako aj s *Y*! Atribúty *X* a *Y* sú teda len zdanlivo korelované. Korelácie premennej *Z* s *X* a *Y* zodpovedajú skutočným kauzálnym vzťahom.

2.4 Predspracovanie dát

Vzhľadom na fakty uvedené v úvode tejto kapitoly o vzťahu dát a reálneho sveta, ako aj skutočnosti, že jednotlivé metódy dolovania v dátach majú rôzne požiadavky na formu dát, je tieto potrebné pred samotným aplikovaním metód dolovania v dátach náležite predspracovať. K základným operáciám, ktoré sa spravidla v tejto fáze na dátach realizujú patria nasledovné [3].

Čistenie dát je zamerané na odstránenie chýbajúcich hodnôt, vyhladenie šumov, odstránenie výrazne odchýlených hodnôt a korekciu nekonzistentností v dátach.

Integrácia dát z viacerých zdrojov je často nevyhnutným krokom pre získanie všetkých relevantných dát.

Transformácia dát predstavuje aplikovanie takých operácií ako normalizácia, agregácia, alebo generalizácia, s cieľom dosiahnuť čo najlepší výsledok vo fáze dolovania v dátach.

Redukcia dát sa snaží zredukovať často obrovské množstvo dát takým spôsobom, aby sa zachovali (prakticky) rovnaké výsledky analýzy.

Niektoré operácie môžu mať rôzne ciele (napríklad vyhladzovanie dát je možné robiť s cieľom odstrániť šumy, alebo s cieľom dáta zredukovať, resp. diskretizovať a pod.), preto sa môžu spomínať na viacerých miestach v rámci nasledujúceho podrobnejšieho rozboru. Každá metóda je však podrobnejšie popísaná vždy len na jednom mieste, podľa kontextu, v ktorom sa najčastejšie využíva.

2.4.1 Čistenie dát

Existuje viacero spôsobov, ako sa vysporiadať s **chýbajúcimi hodnotami**, napr. tieto.

- *Ignorovať záznam* (objekt), v ktorom sa chýbajúca hodnota vyskytla. Toto nie je veľmi efektívna metóda, pokiaľ záznam neobsahuje viacero chýbajúcich hodnôt. Je problematická najmä vtedy, keď počet záznamov s chýbajúcimi hodnotami je percentuálne väčší.

- *Vyplniť chýbajúce hodnoty manuálne* – je realizovateľné len pri veľmi malom počte chýbajúcich hodnôt; časovo náročné.
- *Použiť konštantnú hodnotu* na vyplnenie chýbajúcich hodnôt daného atribútu v celej databáze, napr. hodnotou „neznáme“. Tu je však nebezpečenstvo, že algoritmus dolovania v dátach chybne interpretuje takýto reťazec.
- *Použiť aritmetický priemer* hodnôt daného atribútu pre vyplnenie chýbajúcich hodnôt.
- *Použiť aritmetický priemer hodnôt príkladov patriacich do tej istej triedy* ako záznam, u ktorého hodnota daného atribútu chýba. Toto je možné aplikovať napr. u klasifikačných úloh, kde je k dispozícii informácia o triede.
- *Použiť najpravdepodobnejšiu hodnotu* pre doplnenie chýbajúcej hodnoty. Táto môže byť vypočítaná napr. regresiou, Bayesovským klasifikátorom, alebo indukciou rozhodovacích stromov.

Posledné štyri uvedené prístupy vnášajú do dát určitý bias, lebo nahradené hodnoty nemusia byť správne. Vzniknutá chyba je spravidla najmenšia pri použití posledného prístupu, keďže ten využíva pri stanovení náhradnej hodnoty informácie z ostatných atribútov (predposledný prístup využíva informáciu z atribútu trieda).

Pre ***vyhladenie šumu v dátach*** sa zvyknú používať rôzne techniky „*binningu*“. Ide o metódu, pri ktorej sa vyhladzovanie dát robí na základe ich usporiadanej postupnosti, ktorá sa následne rozdelí na intervaly (akoby sa dáta rozdelili do samostatných nádob – podľa angl. „bin“). Všetky pôvodné hodnoty sa potom nahradia novými, ktorých počet je spravidla výrazne nižší ako pôvodný počet hodnôt. Keďže pôvodné hodnoty sa nahrádzajú novými, ktoré sú odvodené len od vlastností príslušného intervalu, preto sa táto technika nazýva aj *lokálne vyhladzovanie*.

Podľa spôsobu vytvárania intervalov na usporiadanej množine dát a výberu spôsobu, akým sa nahradia pôvodné hodnoty v rámci intervalu vznikajú rôzne varianty binningu. Z hľadiska vytvárania intervalov z usporiadanej postupnosti hodnôt možno použiť tieto prístupy:

- Rozdelenie na *rovnakú hĺbku intervalu* (t.j. každý interval obsahuje rovnaký počet hodnôt).
- Rozdelenie na *rovnakú šírku intervalu* (t.j. každý interval má rovnakú šírku, bez ohľadu na to, koľko hodnôt do neho spadá).

Spôsoby, akými je možné nahrádzať pôvodné hodnoty novými, sú napríklad tieto:

- Vyhladzovanie *na stred intervalov*, t.j. všetky hodnoty z daného intervalu sa nahradia jedinou, a to aritmetickým stredom daného intervalu.
- Vyhladzovanie *podľa mediánov*, t.j. všetky hodnoty z daného intervalu sa nahradia jeho mediánom.
- Vyhladzovanie *podľa hraníc intervalov*, t.j. každá hodnota v intervale sa nahradí tou z hraníc daného intervalu, ktorá je k nej bližšie.

Príklad 2.4

Daná je usporiadaná množina hodnôt atribútu cena (v Sk):

4 8 15 21 21 24 25 28 34

Pri použití rozdelenia na rovnakú hĺbku vzniknú tri intervaly s rovnakým počtom hodnôt – tri. Prvý interval obsahuje hodnoty (4, 8, 15), druhý interval bude (21, 21, 24) a tretí interval (25, 28, 34).

Aplikovaním vyhladzovania na stred intervalov vznikne nasledovná množina vyhladených dát. Prvý interval má strednú hodnotu 9, takže jeho tri hodnoty budú (9, 9, 9), druhý interval má stred 22, takže nové hodnoty budú (22, 22, 22) a hodnoty v treťom intervale budú nahradené (29, 29, 29).

Pri vyhladzovaní podľa hraníc intervalov budú ale nové hodnoty vyzerať trocha ináč. Prvý interval bude obsahovať nové hodnoty (4, 4, 15), druhý interval bude (21, 21, 24) a tretí interval (25, 25, 34).

Za vyhladzovanie je možné považovať aj metódy *diskretizácie*, ktoré budú popísané nižšie, v časti venovanej transformácii dát.

Iným prístupom k vyhladzovaniu dát je napr. *regresia*. V tomto prípade sa dáta vyhladzujú podľa funkcie, ktorú popisujú (napríklad pre nájdenie lineárnej funkcie popísanej dvoma, alebo viacerými premennými, je možné použiť lineárnu regresiu). Predpokladom použitia takýchto prístupov je samozrejme fakt, že uvažované atribúty sú numerické a je reálne očakávať, že medzi nimi takáto funkčná závislosť existuje.

Výrazne odchýlené hodnoty (outliers) je možné odhaliť aplikovaním metód zhlukovania, v rámci ktorého sa identifikujú skupiny „podobných" záznamov (objektov). Tie záznamy, ktoré nepatria do žiadneho zhluku, sú považované za výrazne odchýlené a v určitých aplikáciách KDD (ako napr. segmentácia trhu, analýza väčších skupín zákazníkov) je možné ich odstrániť. V iných zase sa ďalšia analýza sústreďuje práve na ne (odhaľovanie podvodov). O zhlukovaní pojednáva druhá časť poslednej kapitoly, kde je zhlukovanie predstavené ako jeden z hlavných typov dolovania v dátach.

Nekonzistentnosť dát je možné odhaliť samozrejme ručne (použitím externých informácií a zdrojov), alebo znalostným prístupom. Ak sú napríklad známe určité funkčné závislosti medzi atribútmi, potom na ich základe je možné nájsť všetky hodnoty, ktoré odporujú daným funkčným ohraničeniam.

Iné typy nekonzistentností vznikajú pri integrácii dát z viacerých zdrojov, o čom pojednáva nasledujúca časť.

2.4.2 Integrácia dát

Integrácia dát je veľmi dôležitým procesom najmä v prípade, ak je snahou integrovať historické údaje do **dátového skladu** pre analytické účely a pre podporu rozhodovania manažmentu. Dátové sklady ([3], [6]) a operácie OLAP (on-line analytical processing) pre prístup k sumarizovaným dátam dátových skladov niektorí dokonca nazývajú manuálnym dolovaním v dátach ([3]). Problematika dátových skladov však presahuje rámec tohto učebného textu, aj keď bez ujmy na všeobecnosti sa predpokladá, že dátový sklad môže byť výborným zdrojom dát pre proces KDD a môže viesť ku skráteniu etapy predspracovania dát, keďže obsahuje už integrované a vyčistené dáta.

Pri integrácii je potrebné riešiť viacero problémov. Jedným z nich je napríklad problém ***identifikácie rovnakých entít***. Ide o to, akým spôsobom identifikovať rovnaké reálne entity vo viacerých dátových zdrojoch. Ako si môže byť analytik istý, že napr. atribút *customer_id* v jednej databáze sa odkazuje na tú istú entitu ako *cust_number* v inej databáze? Na riešenie takýchto problémov pri integrácii schém sa využívajú metadáta, ktoré obsahuje každá databáza i dátový sklad. Okrem identifikácie redundantných atribútov tu pristupuje aj identifikácia tých istých objektov v rôznych databázach, prípadne rovnakých (duplicitných) záznamov.

Iným problémom pri integrácii je ***redundancia***. Niektorý atribút môže byť redundantný (nadbytočný), keď sa dá odvodiť z inej tabuľky (napr. čistý príjem možno odvodiť z hrubého po odčítaní odvodov a dane). Niektoré redundancie možno odvodiť aj korelačnou analýzou (viď. vyššie).

Tretím dôležitým problémom je odstránenie ***rozdielov v dátach spôsobených rôznymi veličinami***. Napríklad pri integrovaní databáz z oblastí, kde sa používajú iné metrické systémy (napr. britský). Ale rozdiely môžu vznikať aj z iných dôvodov. Napríklad niektoré hotely účtujú cenu izby vrátane raňajok, iné zase zvlášť. Aj takúto sémantickú heterogénnosť je potrebné riešiť pri integrácii dát.

2.4.3 Transformácia dát

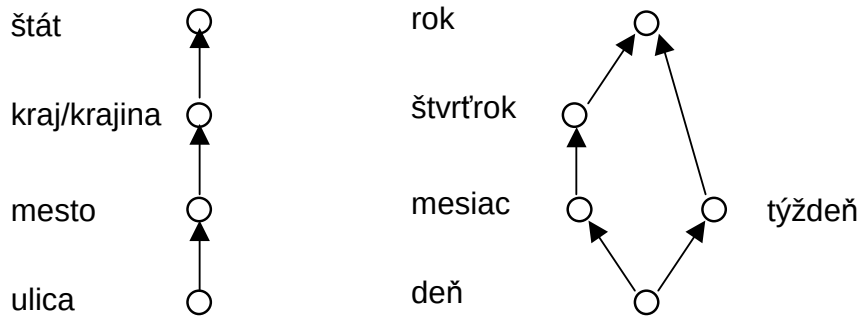
Cieľom transformácie dát je vhodnými operáciami s dátami dosiahnuť takú ich podobu, aby bolo možné aplikovať zvolenú metódu dolovania v dátach (napr. niektoré metódy vedia pracovať len so symbolickými, resp. iba s numerickými atribútmi, iné zase vyžadujú aby tieto boli normalizované a pod.). Naviac je snaha dosiahnuť čo najlepšiu kvalitu získaných znalostí (presnosť klasifikačného modelu, spoľahlivosť asociačných pravidiel a pod.).

K najčastejšie používaným metódam transformácie dát patria nasledovné.

Vyhladzovanie (využívajúce techniky ako binning, zhlukovanie, alebo regresiu) bolo popísané vyššie, pri čistení dát.

Generalizácia, kedy dáta na nižšej úrovni abstrakcie sú nahrádzané všeobecnejšími konceptami. Napr. symbolický atribút „ulica“ môže byť nahradený všeobecnejšími ako „mesto“, alebo „kraj“. Podobne numerické atribúty ako napr. „vek“ v rokoch môže byť namapovaný na všeobecnejšie koncepty ako mladý, stredný vek, senior a pod.

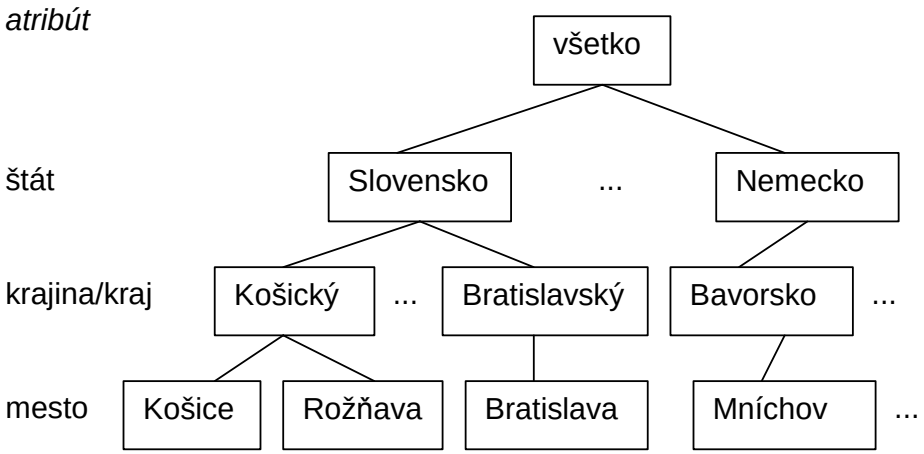
Generalizácia veľmi často využíva existenciu nejakej hierarchie konceptov. *Hierarchia konceptov* definuje postupnosť mapovaní z množiny konceptov nižšej úrovne (špecifickejšie koncepty) na množinu konceptov vyššej



Obr. 2.5 Príklad úplne a čiastočne usporiadanej schémy hierarchie konceptov.

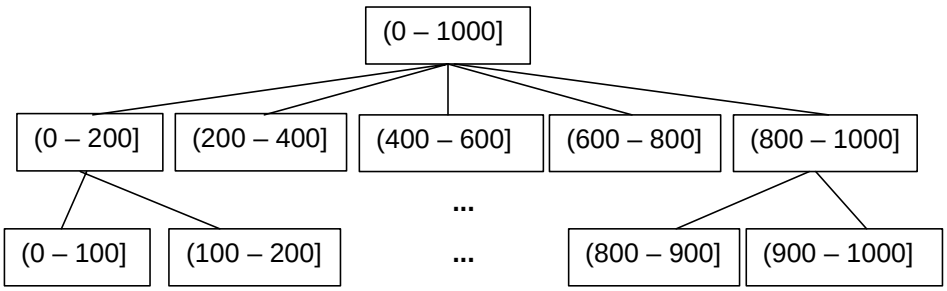
úrovne (všeobecnejšie). Hierarchie konceptov sú väčšinou základom jednotlivých dimenzií v dátových skladoch [3]. Príkladom môže byť hierarchia konceptov pre určenie polohy nejakého objektu (Obr. 2.5 a Obr. 2.6).

Takéto hierarchie konceptov existujú spravidla iba implicitne v databáze a sú tvorené množinou atribútov. V príklade z Obr. 2.5 ide o atribúty *mesto*, *krajina/kraj* a *štát*, ale mohli by tam byť aj ďalšie, ako napr. *ulica* a *číslo domu*.



Obr. 2.6 Hierarchia konceptov dimenzie poloha.

Atribúty v tomto prípade vytvárajú úplné usporiadanie. Nie vždy však tomu tak musí byť. V prípade dimenzie čas napríklad by sme mohli mať atribúty *mesiac* a *týždeň*, ktoré netvoria jednoznačné usporiadanie. Vzniká tak čiastočne usporiadaná hierarchia konceptov (Obr. 2.5).



Obr. 2.7 Hierarchia konceptov pre atribút „cena“.

Práve *definovanie čiastočného alebo úplného usporiadania medzi množinou atribútov* je jednou z najčastejšie používaných metód generalizácie.

Inou možnosťou je *definovať časť hierarchie konceptov explicitným zoskupením dát*. Napr. {Košický, Prešovský} $\subset$ Východ, {Banskobystrický, Nitriansky, Trenčiansky} $\subset$ Stred, {Bratislavský, Trnavský, Žilinský} $\subset$ Západ.

Hierarchie konceptov je možné stanoviť aj pre numerické atribúty (podrobnejšie v nasledujúcom odstavci).

Diskretizácia je vlastne proces transformácie numerického atribútu na symbolický. Ide o operáciu ktorá je nevyhnutná v prípade, ak algoritmus dolovania v dátach je schopný narábať len so symbolickými atribútmi (mnohé algoritmy strojového učenia na objavovanie napr. klasifikačných modelov). Okrem toho sa využíva aj ako redukčná metóda, podobne ako generalizácia, keďže znižuje počet hodnôt daného spojitého numerického atribútu jeho rozdelením na intervaly.

Mnohé diskretizačné techniky sa aplikujú rekurzívne s cieľom poskytnúť možnosť rozlíšenia hodnôt atribútu na viacerých úrovniach všeobecnosti, t.j. vyššie uvedené hierarchie konceptov. Hierarchia konceptov pre daný numerický atribút definuje vlastne jeho diskretizáciu. Numerické hodnoty (nachádzajú sa na najnižšej úrovni v hierarchii), je potom možné nahradiť konceptami vyššej úrovne. Takto diskretizované dáta sú vlastne všeobecnejšie a tým pádom často aj zrozumiteľnejšie. Príkladom môže byť hierarchia konceptov pre atribút „cena“ (Obr. 2.7).

Ručné definovanie takýchto hierarchií konceptov môže byť časovo náročné, preto sú často používané automaticky generované hierarchie, alebo dynamicky spresňované na základe štatistickej analýzy dát (napr. vyššie uvedené metódy binning, histogramy, alebo zhlukovanie).

Často sa zvykne v praxi používať aj tzv. *pravidlo 3-4-5*. Toto pravidlo vedie na segmentáciu numerických dát do relatívne „prirodzených“ intervalov. Podľa tohoto pravidla dané rozpätie numerických dát sa rozdelí na 3, 4 alebo 5 intervalov rovnakej šírky podľa hodnoty cifry na najvýznamnejšom mieste nasledovne.

- Ak daný interval pokrýva 3, 6, 7, alebo 9 rôznych hodnôt na mieste najvýznamnejšej cifry, potom sa rozdelí na 3 intervaly (pre 3, 6 a 9 to znamená tri intervaly s rovnakou šírkou, resp. v prípade 7-ky sú to 3 intervaly so zoskupením 2-3-2 rôznych hodnôt na ráde najvýznamnejšej cifry).

- Ak daný interval pokrýva 2, 4, alebo 8 rôznych hodnôt na mieste najvýznamnejšej cifry, potom sa rozdelí na 4 intervaly rovnakej šírky.
- Ak daný interval pokrýva 1, 5, alebo 10 rôznych hodnôt na mieste najvýznamnejšej cifry, potom sa rozdelí na 5 intervalov rovnakej šírky.

Príklad 2.5

Predpokladajme, že potrebujeme diskretizovať atribút, ktorý vyjadruje zisk jednotlivých pobočiek veľkej firmy za rôzne časové obdobia. Hodnoty tohoto atribútu sa v databáze pohybujú od -351,9 do 4.700.896,5 tis. Sk. Používateľ chce automaticky vygenerovať hierarchiu konceptov pre tento atribút s použitím pravidla 3-4-5.

Na prvej úrovni segmentácie možno uvažovať ako východiskové hodnoty 5-ty a 95-ty (nech sú to hodnoty –159,8, resp. 1.838.761,0). Ak sa zaokrúhlia obe tieto hodnoty na najvýznamnejšiu cifru (t.j. milióny), dostaneme -1.000.000,- resp. +2.000.000,- pre dolnú a hornú hranicu. Keďže medzi týmito dvoma hodnotami sú 3 rôzne cifry na najvýznamnejšom mieste (0, plus 1 a plus 2 milióny), interval sa rozdelí na prvej úrovni hierarchie na 3 rovnako široké intervaly, a síce (-1.000.000, 0], (0, 1.000.000] a (1.000.000, 2.000.000]. Teraz je potrebné ešte ošetriť minimálnu a maximálnu hodnotu. Kým minimálna hodnota z databázy leží v prvom z uvedených intervalov (-351,9 > -1.000.000), maximálna prekračuje posledný interval (4.700.896,5 > 2.000.000). Preto pre prvý interval stačí primerane upraviť spodnú hranicu zaokrúhlením na najvyššiu platnú cifru minimálnej sumy, t.j. stotisíce na (-400.000, 0]. Kvôli maximálnej hodnote je potrebné vytvoriť nový interval, ktorého horná hranica vznikne zaokrúhlením maximálnej hodnoty na jej najvyššiu platnú cifru, teda milióny na (2.000.000, 5.000.000].

Rekurzívne každý zo štyroch intervalov prvej úrovne vznikajúcej hierarchie konceptov možno ďalej rozdeliť podľa pravidla 3-4-5 na druhú úroveň hierarchie konceptov nasledovne.

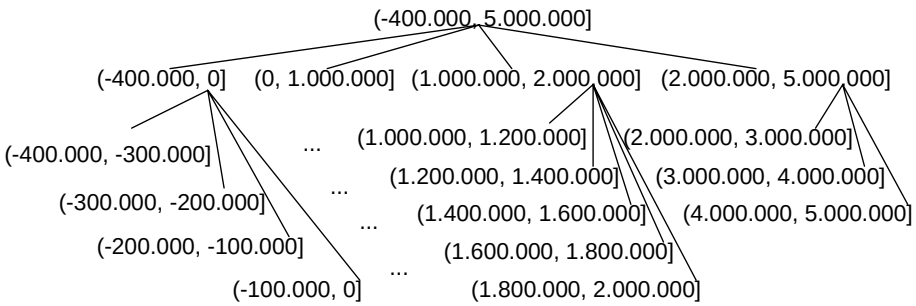
- Prvý interval (-400.000, 0] obsahuje 4 rôzne hodnoty na ráde najvýznamnejšej cifry, preto sa rozdelí na 4 podintervaly: (-400.000, -300.000], (-300.000, -200.000], (-200.000, -100.000] a (-100.000, 0].
- Druhý interval (0, 1.000.000] obsahuje 10 rôznych hodnôt na ráde najvýznamnejšej cifry, preto sa rozdelí na 5 podintervalov: (0, 200.000], (200.000, 400.000], (400.000, 600.000], (600.000, 800.000] a (800.000, 1.000.000].
- Tretí interval (1.000.000, 2.000.000] obsahuje podobne ako predchádzajúci sa rozdelí na 5 podintervalov: (1.000.000, 1.200.000], (1.200.000, 1.400.000], (1.400.000, 1.600.000], (1.600.000, 1.800.000] a (1.800.000, 2.000.000].
- Štvrtý interval (2.000.000, 5.000.000] obsahuje 3 rôzne hodnoty na ráde najvýznamnejšej cifry, preto sa rozdelí na 3 podintervaly: (2.000.000, 3.000.000], (3.000.000, 4.000.000] a (4.000.000, 5.000.000].

Výsledná hierarchia konceptov využiteľná pre diskretizáciu na dvoch rôznych úrovniach všeobecnosti, je znázornená na Obr. 2.8. Podobne je možné v prípade potreby rekurzívne pokračovať v aplikovaní pravidla 3-4-5 na získanie ďalších úrovní v hierarchii konceptov pre jemnejšiu diskretizáciu.

Normalizácia, kde hodnoty atribútu sú preškálované na malý, vopred špecifikovaný rozsah, napr. interval <-1,1>, alebo <0,1> ([3], [8]). Táto operácia je veľmi dôležitá napr. pri použití neurónových sietí (napr. na predikciu), ale aj pri použití metód založených na meraní vzdialeností (klasifikácia založená na k najbližších susedov – kNN, alebo niektoré algoritmy zhlukovania), kde sa takto eliminuje vplyv atribútov s najvyššími absolútnymi hodnotami na úkor atribútov s malými nominálnymi hodnotami [8].

Používajú sa napr. tieto tri techniky normalizácie.

- *Min-max normalizácia* – lineárna transformácia pôvodných dát. Nech min_A a max_A sú minimálna, resp. maximálna hodnota atribútu *A*. Min-max normalizácia mapuje pôvodné hodnoty *v* na hodnoty *v'* z novo stanoveného intervalu <new_min_A, new_max_A> nasledovným spôsobom: $v' = \frac{v - \min_A}{\max_A - \min_A} \cdot (new\_\max_A - new\_\min_A) + new\_\min_A$



Obr. 2.8 Automaticky vygenerovaná hierarchia konceptov atribútu „zisk“ pomocou pravidla 3-4-5.

- *Normalizácia na nulový stred* – hodnoty atribútu A sú normalizované vzhľadom na ich aritmetický priemer $\overline{x_A}$ a štandardnú odchýlku $\overline{s_A}$ nasledovne: $v' = \frac{v - \overline{x_A}}{\overline{s_A}}$. Tento spôsob normalizácie je vhodný najmä vtedy, ak nie sú známe minimálne a maximálne možné hodnoty atribútu *A*, alebo ak by pri normalizácii min-max dominovali výrazne odchýlené hodnoty.
- *Normalizácia decimálnym škálovaním* – normalizuje posunom desatinnej čiarky v hodnotách atribútu *A*. Počet desatinných miest, o ktoré sa desatinná čiarka posunie, závisí na maximálnej absolútnej hodnote *A*. $v' = \frac{v}{10^j}$, kde *j* je najmenšie celé číslo také, že $Max(|v'|) < 1$

Je potrebné zdôrazniť, že normalizácia môže značným spôsobom zmeniť dáta. Naviac pri normalizácii na nulový stred a pri decimálnom škálovaní je nevyhnutné uložiť použité parametre normalizácie, aby bolo možné rovnakým spôsobom normalizovať aj budúce dáta.

Konštrukcia atribútov, keď sa z existujúcich atribútov skonštruuje a pridá k nim nový, s cieľom napomôcť procesu dolovania v dátach zvýšením presnosti a porozumenia mnohorozmerným dátam. Binárne atribúty možno spájať napr. logickým operátorom *AND*, numerické zas napríklad pomocou aritmetického súčinu (napr. ak máme atribúty výška a šírka, ich súčinom je možné vytvoriť nový atribút plocha).

Takto vytvorené atribúty môžu napríklad zmierniť problém fragmentácie pri odvodzovaní klasifikačných modelov vo forme rozhodovacích stromov, keď nejaký atribút je opakovane testovaný pozdĺž určitej cesty v generovanom strome. Takisto je možné týmto postupom získať chýbajúcu informáciu o vzťahoch medzi dátami, ktorá môže byť užitočná v procese KDD.

2.4.4 Redukcia dát

Pri aplikácii procesu KDD je potrebné spravidla spracovať obrovské množstvá dát. Analýza veľkého množstva zložitých dát môže trvať dlhý čas, čo je nepraktické, alebo niekedy dokonca až nerealizovateľné. Techniky redukcie dát slúžia na to, aby sa z pôvodných dát získala ich primerane redukovaná reprezentácia, ktorá ale zachováva vlastnosti pôvodných dát. Pritom sa zvyknú využívať nasledovné metódy.

Agregácia sa najčastejšie využíva v procese konštrukcie dátovej kocky v dátovom sklade. Deje sa tak vlastne výberom vyššej úrovne abstrakcie v hierarchii konceptov asociovanej s niektorou z dimenzií dátovej kocky. Výsledkom je sumarizácia všetkých hodnôt na nižšej úrovni abstrakcie na hodnoty zodpovedajúce zvolenej vyššej úrovni. Napr. ak máme dátový sklad s údajmi o predajoch s dimenziou čas (ktorej je priradená hierarchia konceptov podľa Obr. 2.5), môžeme prechodom z úrovne štvrťrokov na úroveň rokov získať zosumarizované údaje, kde pre každý rok sú vlastne spočítané údaje za všetky jeho štyri štvrťroky.

Redukcia dimenzií, kde sa zisťujú a následne odstraňujú atribúty (resp. v prípade dátových skladov dimenzie), ktoré nie sú relevantné, sú slabo relevantné, alebo redundantné v danej úlohe DM. Ponechanie takýchto atribútov môže byť škodlivé, lebo často nepriaznivo ovplyvňuje algoritmus dolovania v dátach a vedie k výsledkom slabej kvality. Okrem toho samozrejme takéto redundantné atribúty zväčšujú zbytočne objem dát, ktoré je potrebné spracovať a preto spomaľujú dolovanie v dátach. Naviac výsledky DM dosiahnuté s menším počtom atribútov sú často jednoduchšie, a teda zrozumiteľnejšie.

Cieľom *výberu podmnožiny atribútov* je vybrať minimálnu podmnožinu pôvodných atribútov tak, aby sa zachovala výsledná distribúcia pravdepodobnosti jednotlivých tried. Nech databáza obsahuje *d* atribútov. Potom ale existuje 2^d možných podmnožín z týchto atribútov. To znamená, že pre väčšie d nie je možné v dôsledku časovej zložitosti vyskúšať všetky

alternatívy. Preto sa zvyknú používať heuristiky, založené na výbere lokálne optimálnej alternatívy počas prehľadávania. Nezaručujú síce nájdenie optimálneho riešenia, ale v praxi sa k nemu často približujú dosť blízko.

Základné heuristické postupy sú tri:

- *Postupný dopredný výber* – začína prázdnou množinou atribútov a v každom kroku pridá „najlepší“ z tých atribútov, ktoré zostali.
- *Postupná spätná eliminácia* – začína s úplnou množinou atribútov a v každom kroku odoberie „najhorší“ zo zostávajúcich atribútov.
- *Kombinácia dopredného výberu a spätnej eliminácie*.

Kritérium pre výber atribútu v každom kroku môže byť napr. test štatistickej významnosti, alebo informačný zisk ak ide o prediktívnu úlohu. Kritérium pre ukončenie procesu výberu atribútov môže byť tiež rôzne, spravidla je daná nejaká prahová hodnota určitej veličiny, po prekročení ktorej sa vo výbere už nepokračuje a aktuálna podmnožina atribútov je vyhlásená za najlepšiu.

Inou metódou pri dátach, kde existuje atribút zaradzujúci príklady do určitej triedy, je napr. *použitie indukcie rozhodovacích stromov* [9] (ako napr. ID3 alebo C4.5). Po vygenerovaní rozhodovacieho stromu sa potom vylúčia všetky tie atribúty, ktoré sa v strome nevyskytli (ako testovacie atribúty v nelistových uzloch).

Redukcia početnosti dát, kde sú pôvodné dáta nahradené alternatívnou, menšou reprezentáciou, a to buď *parametricky* alebo *neparametricky*.

V prípade parametrickej redukcie dát sa ukladajú len parametre modelu, napr. parametre regresnej krivky popisujúcej funkčnú závislosť medzi dvoma alebo viacerými atribútmi. Okrem nich sa môžu ukladať aj výrazne sa odchyľujúce príklady.

K neparametrickým metódam redukcie patria vzorkovanie (sampling), zhlukovanie a použitie histogramov. *Histogramy* používajú vlastne binning (viď. vyššie) na aproximáciu distribúcie a sú populárnou formou redukcie dát. Histogram pre atribút *A* rozdelí distribúciu dát do disjunktných podmnožín (buckets). Tieto sú potom zobrazované na horizontálnej osi, kým výška stĺpca grafu vyjadruje obvykle priemernú frekvenciu výskytu hodnôt z danej podmnožiny. Táto podmnožina môže byť jednoprvková, ale častejšie vyjadruje súvislý interval hodnôt spojitého atribútu (čím vlastne dochádza k redukcii počtu hodnôt).

Zhlukovanie sa snaží rozdeliť objekty z databázy do skupín tak, aby objekty v rámci skupiny si boli čo najviac podobné a objekty z rôznych skupín zase čo najviac nepodobné. Pritom pod podobnosťou sa tu má na mysli ich blízkosť v zmysle nejakej funkcie vzdialenosti. V prípade použitia zhlukovania pre účely redukcie vlastne reprezentácia zhluku nahrádza reprezentáciu jednotlivých objektov v ňom.

Vzorkovanie vlastne znamená výber podmnožiny záznamov (objektov) z databázy *D* veľkosti *N*. Niektoré z najčastejšie používaných vzorkovacích prístupov sú nasledovné.

- Jednoduchá *náhodná vzorka veľkosti n bez nahradenia* – ide o náhodný výber *n* záznamov z *N* v *D* ($n < N$), pričom

pravdepodobnosť zaradenia ľubovoľného záznamu z *D* je 1/*N*, t.j. všetky záznamy sú rovnako pravdepodobné.

- Jednoduchá *náhodná vzorka veľkosti n s nahradením* – ide o podobný postup ako v predchádzajúcom prípade, ale zakaždým, keď je nejaký záznam vybraný, je zaznamenaný a opäť sa vráti do *D*, takže môže byť neskôr vybraný opätovne.
- Rozvrstvená vzorka (stratified sample) – ak je databáza *D* rozdelená na vzájomne disjunktné vrstvy, potom sa náhodný výber uskutočňuje z každej takejto vrstvy. Tým sa zabezpečí reprezentatívna vzorka (napr. ak jednotlivé vrstvy reprezentujú rôzne skupiny zákazníkov, tak aj málo početné skupiny budú mať zastúpenie vo výslednej vzorke).

Výhodou uvedených metód vzorkovania je jednak to, že nevnášajú do dát žiaden umelý bias (keďže ide o náhodný výber, nie systematický) a tiež to, že ich časová zložitosť je priamo úmerná veľkosti vzorky *n* nie veľkosti celej databázy *N*. Nevýhodou je nutnosť stanoviť hodnotu *n* vopred.

Náhodné vzorkovanie založené na konvergencii variability je možné použiť pre numerické atribúty. Pri tomto type vzorkovania prebieha náhodný výber vzoriek podľa prvej z vyššie uvedených stratégií, ale nie po dosiahnutie stanoveného počtu n, ale do dosiahnutia určitej hladiny spoľahlivosti, s ktorou sa dáta vo vzorke svojou variabilitou blížia k variabilite dát celej databázy. Používateľ si takto stanoví vlastne požadovanú hladinu spoľahlivosti, vzorkovanie vyberá náhodne príklady do vzorky dovtedy, kým zmeny variancie dát vo vzorke neklesnú pod zodpovedajúcu hodnotu.

Diskretizácia a generovanie hierarchie konceptov, kde pôvodné dáta sú nahrádzané intervalmi, alebo konceptami na vyššej úrovni všeobecnosti. Tieto metódy boli popísané v predchádzajúcom bode venovanom transformácii dát.

2.5 Použitá literatúra

[1] Cios, K., Pedrycz, W. and Swiniarski, R.:Data Mining Methods for Knowledge Discovery. Kluwer Academic Publishers, (1998)

[2] Ester, M., Sander, J.: Knowledge Discovery in Databases – Techniken und Anwendungen. Springer Verlag, (2000)

[3] Han, J., Kamber, M.: Data Mining – Concepts and Techniques. Morgan Kaufmann Publishers, (2000)

[4] Hanousek, J. a Charamza, P.: Moderní metody zpracování dat – Matematická štatistika pro každého, Grada, (1992)

[5] Hindls, R., Hronová, S. a Novák, I.: Analýza dat v manažerském rozhodování. Grada, (1999)

[6] Kimball, R.: The Data Warehouse Toolkit. New York: John Wiley & Sons, (1996)

[7] Klösgen, W. and Zytkow, J.M. (eds.): Handbook of Data Mining and Knowledge Discovery. Oxford Univesity Press, (2002)

[8] Lukasová, A. a Šarmanová, J.: Metody shlukové analýzy. SNTL – Nakladatelství technické literatury, Praha (1985)

[9] Machová, K.: Strojové učenie. Princípy a algoritmy. Elfa, Košice, 117 s. ISBN 80 89066-51-8 (2002)

[10] Pyle, D.: Data Preparation for Data Mining. Morgan Kaufman Publishers, Inc., San Francisco, (1999)

3 Popisné dolovanie v dátach

Táto a nasledujúce dve kapitoly sú venované kľúčovému kroku v procese KDD, a síce dolovaniu v dátach. Dolovanie v dátach (DM) bolo stručne charakterizované v úvodnej kapitole, kde boli rozdelené aj najčastejšie sa vyskytujúce úlohy DM do troch skupín: popisné dolovanie v dátach, prediktívne dolovanie v dátach a ďalšie úlohy DM. Tomu zodpovedá aj členenie zvyšných kapitol. Táto kapitola sa teda venuje popisnému dolovaniu v dátach

3.1 Základné pojmy

Popisné dolovanie v dátach ([2], [1]) sa snaží popísať určitú dátovú množinu stručným a výstižným spôsobom. Ide teda vlastne o generalizáciu (zovšeobecňovanie).

Generalizácia je proces abstrakcie veľkej množiny dát z databázy relevantných pre danú úlohu z relatívne nízkej konceptuálnej úrovne na vyššie konceptuálne úrovne. Generalizácia ako metóda predspracovania dát bola stručne popísaná v druhej kapitole. Táto kapitola sa však zameriava na **popis**, ako cieľ procesu KDD a na automatické metódy pre jeho získanie.

Popis môže mať dve základné formy:

- *Charakterizácia*, t.j. stručný a výstižný popis danej množiny dát (sumarizácia), alebo
- *Porovnanie (diskriminácia)*, t.j. popis porovnávajúci 2 alebo viac množín dát.

3.2 Metódy charakterizácie

K popisnému DM možno zahrnúť do určitej miery aj **operácie OLAP** (On-line analytické spracovanie, t.j. Online Analytical Processing) nad dátovými skladmi. V tomto prípade ide o tzv. *manuálne zovšeobecňovanie*, keďže používateľ riadi krok po kroku celý proces generalizácie výberom jednotlivých operácií a pohľadov [2]. Dôležitým obmedzením tohto prístupu je rozsah dátových typov, keďže operácie OLAP je možné vykonávať len nad numerickými dátami. Veličiny v dátových skladoch sú vždy numerické atribúty, dimenzie sú popisované nenumerickými dátami.

Druhú skupinu tvoria **metódy automatickej generalizácie**, kde je proces zovšeobecňovania riadený algoritmom, ktorý používa parametre nastavené používateľom. Najdôležitejšou metódou z tejto skupiny je atribútovo orientovaná indukcia. Táto metóda bude najprv predstavená neformálne [2], potom vo forme presného algoritmu [1] a nakoniec vysvetlená na konkrétnom príklade.

3.2.1 Atribútovo orientovaná indukcia - neformálny popis

Celý postup začína *výberom relevantnej množiny dát* (tzv. základná relácia), t.j. napr. cieleným SQL dotazom do relačnej databázy, ktorý vyselektuje cieľovú skupinu dát. Môže ísť napríklad o skupinu zákazníkov s určitými vlastnosťami, množinu podozrivých transakcií v databáze, množinu vysokoškolských učiteľov v danom regióne a pod.

Potom je potrebné **preskúmať stupeň generalizácie dát** v základnej relácii vytvorenej v predchádzajúcom kroku, t.j. zistiť počet navzájom rôznych hodnôt pre jednotlivé atribúty, ako aj ich existujúce hierarchie konceptov.

Samotná **generalizácia** sa realizuje potom postupne aplikáciou dvoch možných krokov.

- *Odstránenie atribútu* v prípade, ak je u neho veľké množstvo navzájom rôznych hodnôt a súčasne buď neexistuje hierarchia konceptov pre tento atribút (napr. atribút poradové alebo telefónne číslo, meno a pod.), alebo sú koncepty vyššej úrovne vyjadrené prostredníctvom iných atribútov.
- *Generalizácia atribútu* v aktuálnej relácii, ak je u neho veľa rôznych hodnôt a súčasne existuje hierarchia konceptov, tak sa hodnoty v aktuálnej relácii nahradia hodnotami na vyššej úrovni všeobecnosti podľa hierarchie konceptov.

V prípade aplikovania generalizácie atribútu sa následne zlúčia všetky rovnaké záznamy (riadky) v takto získanej relácii a pridá sa údaj o ich počte.

Podmienka ukončenia tohoto procesu generalizácie môže byť rôzna. Môže to byť napr. stanovenie prahu pre generalizovanie atribútu určením maximálneho počtu rôznych hodnôt, ktoré musia jednotlivé atribúty nadobúdať (obvykle je to číslo medzi 2 a 8). Inou možnosťou je zase stanoviť prah pre generalizáciu relácie, t.j. určiť maximálny počet záznamov (riadkov) vo výslednej generalizovanej relácii (obvykle je to číslo medzi 10 a 30).

Krok generalizácie sa potom aplikuje opakovane dovtedy, kým výsledná generalizovaná relácia nespĺňa stanovenú podmienku ukončenia.

Posledným krokom je **zobrazenie výslednej generalizovanej relácie**. Tu existuje niekoľko spôsobov, ako zovšeobecnenú reláciu prezentovať (napr. grafom, alebo pravidlami).

3.2.2 Atribútovo orientovaná indukcia - formálny popis

Nech *R* je *relácia* $R \subseteq D_1 x D_2 x \ldots x D_d$ s atribútmi $A_1, A_2, \ldots, A_d$, pričom $D_1, \ldots, D_d$ sú domény alebo definičné obory ich hodnôt. V ďalšom sa predpokladá, že pre každý atribút A_i je definovaná *hierarchia konceptov* C_i. Nech $|R|$ označuje počet záznamov (riadkov) v relácii *R* a m_i je počet rôznych hodnôt atribútu A_i, $1 \le i \le d$ v *R*.

Stupeň generalizácie atribútu A_i, označený ako g_i, je definovaný ako spoločná úroveň všeobecnosti hodnôt atribútu A_i. *Úroveň všeobecnosti h* zodpovedá úrovni, na ktorej sa hodnoty hodnoty daného atribútu v strome k nemu asociovanej hierarchie konceptov. Táto definícia predpokladá, že všetky hodnoty toho istého atribútu A_i v *R* majú rovnakú úroveň všeobecnosti (vzhľadom na koncept C_i). Atribút s najšpecifickejšími hodnotami (na úrovni listov v hierarchii konceptov) má stupeň generalizácie 0. *Stupeň generalizácie celej relácie R*, označovaný G_R, je definovaný ako $G_R = \sum_{i=1}^{d} g_i$.

Podpora (početnosť) záznamu je definovaná ako počet záznamov z *R*, ktoré boli na daný (zovšeobecnený) záznam generalizované. Všetky záznamy v základnej relácii majú podporu 1.

R sa nazýva *základnou reláciou*, keď všetky jej atribúty majú stupeň generalizácie 0, t.j. ešte nebola vykonaná žiadna generalizácia. *R* sa nazýva generalizovanou reláciou, ak jej stupeň generalizácie je väčší ako 0 ($G_R > 0$). Každá generalizovaná relácia obsahuje jeden dodatočný atribút, ktorý vyjadruje podporu jednotlivých záznamov (t.j. počet záznamov základnej relácie, ktoré sa generalizáciou premietli na daný záznam generalizovanej relácie).

Relácia *R* sa nazýva *plne generalizovanou vzhľadom na celé číslo T* a označuje sa R_T, keď $|R| \leq T$ (ak je podmienkou ukončenia počet záznamov), resp. ak $\forall i, 1 \leq i \leq d : m_i \leq T$ (ak je podmienkou ukončenia počet rôznych hodnôt atribútov).

Pre oba druhy podmienok ukončenia sú dané nasledovné **vstupné parametre**:

- základná relácia *R*,
- pre každý atribút A_i k nemu asociovaná hierarchia konceptov C_i (nemusí byť)
- celé číslo *T*.

Výstupom je plne generalizovaná relácia vzhľadom na počet záznamov *T*, resp. vzhľadom na počet *T* rôznych hodnôt atribútov.

```
Generalizácia_na_počet_záznamov (Relácia R, Integer T)
     Gen := R;
     Pre každé i urči hodnotu mi a inicializuj pre každé
        i premennú gi := 0;
     while |Gen| > T do
          Ai := Výber_nasledujúceho_atribútu(R);
          if hodnoty atribútu Ai môžu byť ďalej generalizované
          then // je k dispozícii Ci a súčasne Ai ≠ All
               for each záznam v Gen do
                    nahradiť hodnotu atribútu Ai v zázname jeho
                       predchodcom v Ci;
                    znovu vypočítaj mi;
               inkrementuj gi;
          else odstráň atribút Ai z Gen;
          usporiadaj Gen;
          eliminuje redundantné záznamy a aktualizuj podporu
             zostávajúcich záznamov v Gen;
     return Gen;
```

Pre výber nasledujúceho atribútu je možné použiť rôzne stratégie, napríklad:

- Ak je cieľom minimálny stupeň generalizácie výslednej relácie, potom je vhodné vyberať atribút, ktorého generalizácia povedie k najväčšej redukcii počtu záznamov. Dobrou stratégiou môže byť vyberať atribút s maximálnou aktuálnou hodnotou m_i.
- Ak je cieľom podobný stupeň generalizácie u všetkých atribútov, potom je potrebné vyberať taký atribút A_i, ktorý má minimálny stupeň generalizácie g_i.

- Používateľ si môže na základe konkrétnej aplikácie stanoviť iné pravidlá výberu atribútu na generalizáciu.

Pre ilustráciu môže zvolená stratégia výberu atribútu vyzerať napr. nasledovne.

```
Výber_nasledujúceho_atribútu (Relácia R)
      vyber atribút Ai s maximálnou hodnotou mi;
      if nie je možné určiť Ai jednoznačne
      then vyber z Ai, ktoré prichádzajú do úvahy, ten s
         minimálnym gi;
            if nie je možné určiť Ai jednoznačne
            then uskutočni výber z Ai podľa lexikografického
               poradia;
      return Ai;
```

3.2.3 Ilustračný príklad

Daná je nasledovná základná relácia R (Tabuľka 3.1) a hierarchie konceptov asociované k atribútom „Birth place“ a „Age“ (Obr. 3.1).

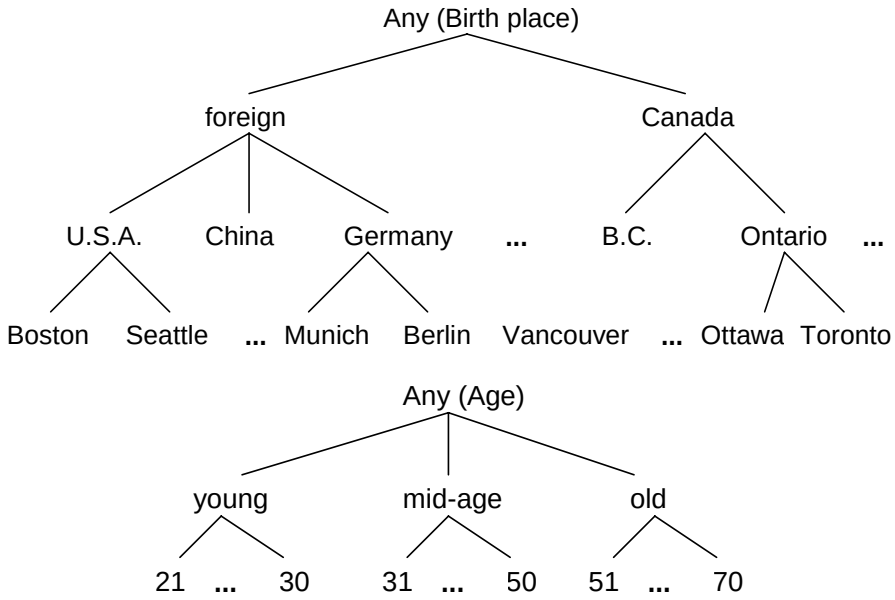
Name	Sex	Age	Birth place	Department	Position	Salary
Anderson	female	26	Burnaby	computer science	secretary	26.000
Bach	male	38	Ottawa	Electrical engineering	lab manager	41.000
Barton	female	30	Toronto	Chemistry	junior lecturer	28.000
Benson	male	45	Vancouver	computer science	full professor	63.000
...	...	...	...	...	...	...
Winton	male	38	Seattle	civil engineering	associated professor	55.400
Young	male	55	Bonn	german	full professor	68.000

Tabuľka 3.1 Základná relácia R – vstup pre generalizáciu.

V prvom kroku sa vyberie atribút A_1, t.j. „Name“, keďže má najväčší počet navzájom rôznych hodnôt m_1. Nakoľko ale pre tento atribút neexistuje hierarchia konceptov, atribút sa odstráni z R a pridá sa nový stĺpec označujúci podporu jednotlivých záznamov v generalizovanej relácii (Tabuľka 3.2).

Sex	Age	Birth place	Department	Position	Salary	Support
female	26	Burnaby	computer science	secretary	26.000	1
male	38	Ottawa	electrical engineering	lab manager	41.000	1
female	30	Toronto	chemistry	junior lecturer	28.000	1
male	45	Vancouver	computer science	full professor	63.000	1
...	...	...	...	...	...	...
male	38	Seattle	civil engineering	associated professor	55.400	1
male	55	Bonn	german	full professor	68.000	1

Tabuľka 3.2 Relácia R po prvom kroku generalizácie.



Obr. 3.1 Hierarchie konceptov pre atribúty „Birth place“ a „Age“.

V ďalšej iterácii má najviac rôznych hodnôt atribút A_3, t.j. „Birth place“ pôvodnej relácie *R*. Pre tento atribút existuje hierarchia konceptov C_3 (Obr. 3.1), takže jeho hodnoty (aktuálne na úrovni listových uzlov) sa nahradia svojimi predchodcami v C_3, t.j. hodnotami štátu, v ktorom sa dotyčný (dotyčná) narodil(a). Výsledkom bude nová generalizovaná relácia s aktualizovanými číslami podpory pre jednotlivé nové záznamy (Tabuľka 3.3).

Sex	Age	Birth place	Department	Position	Salary	Support
female	26	B.C.	computer science	secretary	26.000	1
male	38	Ontario	electrical engineering	lab manager	41.000	3
female	30	Ontario	chemistry	junior lecturer	28.000	2
male	45	B.C.	computer science	full professor	63.000	1
...	...	...	...	...	...	...
male	38	U.S.A.	civil engineering	associated professor	55.400	2
male	55	Germany	german	full professor	68.000	1

Tabuľka 3.3 Relácia R po druhom kroku generalizácie.

Postupným iterovaním procedúry `Generalizácia_na_počet_záznamov(R)` sa postupne generalizuje atribút „Salary“ podľa danej hierarchie konceptov, potom sa odstránia atribúty „Department“ a „Position“, pre ktoré neexistuje hierarchia konceptov a preto ich nie je možné generalizovať. Nakoniec sa ešte generalizuje opätovne atribút „Birth place“ o jednu úroveň vyššie (na úroveň

štátov), čím sa splní podmienka ukončenia a počet záznamov sa zníži pod želanú hodnotu.

Výslednú, plne generalizovanú reláciu znázorňuje Tabuľka 3.4.

Sex	Age	Birth place	Salary	Support
Male	old	Canada	high	20
Male	old	foreign	high	15
Male	mid-age	Canada	medium	75
Male	mid-age	foreign	medium	130
Female	mid-age	Canada	medium	25

Tabuľka 3.4 Úplne generalizovaná relácia.

3.3 Metóda porovnávania

V prípade *automatického porovnávania (diskriminácie)* je postup veľmi podobný charakterizácii s tým rozdielom, že sa paralelne generalizujú rovnakým spôsobom dve, prípadne viac základných relácií, z ktorých každá reprezentuje inú skupinu dát. Cieľom je vlastne porovnanie dvoch alebo viacerých skupín objektov z pohľadu ich spoločných, resp. rozdielnych vlastností.

Rozdiel je najmä v spôsobe spracovania a prezentácii výsledkov, teda úplne generalizovaných relácií. Ak sú napr. porovnávané skupiny dve, je potrebné spracovať výsledky porovnávania do podoby klasifikačných. Podobne, ako v predchádzajúcej časti, táto metóda bude najprv predstavená neformálne, potom vo forme presného algoritmu [1] a nakoniec ukázaná na konkrétnom príklade.

3.3.1 Porovnávanie atribútovo orientovanou indukciou - neformálny popis

Celý postup je v prvej časti zhodný s postupom pri charakterizácii s tým rozdielom, že na začiatku je potrebné vybrať *dve relevantné množiny dát – cieľovú a kontrastnú*, tzv. cieľová relácia a kontrastná relácia, obe nad rovnakou množinou atribútov.

Potom je potrebné ***preskúmať stupeň generalizácie dát*** v oboch reláciách, t.j. zistiť počet navzájom rôznych hodnôt pre jednotlivé atribúty, ako aj ich existujúce hierarchie konceptov a vykonať ***generalizáciu*** rovnakým spôsobom ako v prípade charakterizácie, ale jednotlivé kroky generalizácie musia prebiehať rovnakým spôsobom paralelne v oboch reláciách, teda cieľovej i kontrastnej.

Po získaní ***výsledných generalizovaných relácií*** sa najprv vyznačia prekrývajúce sa záznamy.

Nakoniec sa ***vygenerujú klasifikačné pravidlá*** a určí sa ich spoľahlivosť na základe ich podpory v cieľovej, resp. kontrastnej relácii.

3.3.2 Porovnávanie atribútovo orientovanou indukciou - formálny popis

Nech sú dané dve relácie – cieľová C a kontrastná K, obe s atribútmi $A_1, A_2, ..., A_d$. Pre jednotlivé atribúty A_i môže byť definovaná *hierarchia konceptov* C_i.

Cieľom je nájsť klasifikačné pravidlá a ich spoľahlivosť. Záznam sa nazýva prekrývajúci, ak je obsiahnutý tak v cieľovej relácii *C*, ako aj v kontrastnej relácii *K*.

```
Porovnanie_dvoch_tried (Relácia C, Relácia K, Integer T)
     Ciel := C; Kontrast := K;
     for i = 1 to d do
          Urči hodnotu m_i;
          Inicializuj premennú g_i := 0;
     while |Ciel| > T do
          A_i := Výber_nasledujúceho_atribútu(Ciel);
          if hodnoty atribútu A_i môžu byť ďalej generalizované
          then // t.j. existuje C_i a hodnoty A_i ≠ All
               for each záznam v Ciel do
                    nahraď hodnotu atribútu A_i v zázname jeho
                      predchodcom v C_i;
                    znovu vypočítaj m_i;
                    inkrementuj g_i;
               for each záznam v Kontrast do
                    nahraď hodnotu atribútu A_i v zázname jeho
                    predchodcom v C_i;
          else odstráň atribút A_i z Ciel aj Kontrast;
          usporiadaj Ciel;
          eliminuj redundantné záznamy a aktualizuj podporu
             zostávajúcich záznamov v Ciel;
          usporiadaj Kontrast;
          eliminuj redundantné záznamy a aktualizuj podporu
             zostávajúcich záznamov v Kontrast;
          pritom označ všetky prekrývajúce sa záznamy v Ciel a
             Kontrast;
     generuj z Ciel a Kontrast KlasifikačnéPravidlá;
     return KlasifikačnéPravidlá;
```

3.3.3 Ilustračný príklad

V predchádzajúcej časti bola podrobne vysvetlená metóda charakterizácie, preto sa v tejto časti zameriame už len na tie kroky, v ktorých sa metóda porovnávania odlišuje od charakterizácie, t.j. označenie prekrývajúcich sa záznamov a generovanie klasifikačných pravidiel a ich spoľahlivosti z výsledných generalizovaných relácií *C* a *K*.

Predpokladajme, že obe základné relácie, t.j. cieľová *C* i kontrastná *K* sú odvodené z rovnakej databázy vysokoškolských zamestnancov ako v predchádzajúcom príklade. Cieľovou reláciou pritom bude skupina profesorov a kontrastnou skupinou ostatní zamestnanci. Nech výsledkom paralelného aplikovania atribútovo orientovanej indukcie sú nižšie uvedené úplne generalizované relácie *Ciel* a *Kontrast* (Tabuľka 3.5 a Tabuľka 3.6).

Nasledujúcim krokom je vyznačenie prekrývajúcich sa záznamov. V tomto prípade sú takéto záznamy dva, a síce muži v strednom veku so stredne vysokým príjmom a to tak Kanaďania, ako aj cudzinci.

Sex	Age	Birth place	Salary	Support	Mark
Male	old	Canada	high	20	
Male	old	Foreign	high	15	
Male	mid-age	Canada	medium	75	*
Male	mid-age	Foreign	medium	130	*
Female	mid-age	Canada	medium	25	

Tabuľka 3.5 Úplne generalizovaná cieľová relácia Cieľ.

Sex	Age	Birth place	Salary	Support	Mark
Male	Young	Canada	Low	30	
Male	mid-age	Canada	medium	25	*
Female	Young	Canada	low	12	
Male	mid-age	foreign	medium	10	*

Tabuľka 3.6 Úplne generalizovaná kontrastná relácia Kontrast.

Nakoniec je potrebné vygenerovať klasifikačné pravidlá a určiť ich spoľahlivosť. Pre každý záznam generalizovanej cieľovej relácie *Cieľ* je možné vygenerovať jedno klasifikačné pravidlo. Jeho spoľahlivosť bude 100% (v prípade, že ide o neprekrývajúci sa záznam), alebo percentuálny podiel podpory záznamu v *Cieľ* voči všetkým záznamom (teda voči súčtu podpory tohto záznamu v *Cieľ* a s ním sa prekrývajúceho záznamu v Kontrast).

Tak napríklad v prípade prvého a druhého záznamu relácie *Cieľ* (neprekrývajúce sa záznamy) budú mať vygenerované klasifikačné pravidlá stopercentnú spoľahlivosť a budú vyzerať nasledovne.

Sex=male $\wedge$ Age=old $\wedge$ Birth place=Canada $\wedge$ Salary=High
$\rightarrow$ Professor (100%)
Sex=male $\wedge$ Age=old $\wedge$ Birth place=foreign $\wedge$ Salary=High
$\rightarrow$ Professor (100%)

V prípade tretieho a štvrtého záznamu v *Cieľ* ide o prekrývajúce sa záznamy, keďže rovnaké záznamy sa vyskytujú aj v relácii *Kontrast*. Spoľahlivosť z nich vygenerovaných klasifikačných pravidiel preto už nebude 100%, ale v prípade tretieho záznamu $\frac{75}{75+25} = 75\%$ a v prípade štvrtého záznamu potom $\frac{130}{130+10} = 93\%$. Takže:

Sex=male $\wedge$ Age=mid-age$\wedge$ Birth place=Canada $\wedge$ Salary=Medium
$\rightarrow$ Professor (75%)
Sex=female $\wedge$ Age=mid-age $\wedge$ Birth place=foreign $\wedge$ Salary=Medium
$\rightarrow$ Professor (93%)

A nakoniec z piateho záznamu bude opäť odvodené klasifikačné pravidlo so stopercentnou spoľahlivosťou, keďže ide o neprekrývajúci sa záznam.

Sex=female $\wedge$ Age=mid-age $\wedge$ Birth place=Canada $\wedge$ Salary=Medium
$\rightarrow$ Professor (100%)

3.4 Prezentácia získaných znalostí

3.4.1 Charakterizácia

Okrem *priameho zobrazenia generalizovanej relácie* sú aj ďalšie možnosti prezentácie znalostí získaných atribútovo orientovanou indukciou.

Prvou možnosťou je použitie ***krížovej tabuľky***. V dvojrozmernej krížovej tabuľke každý riadok reprezentuje hodnotu jedného atribútu a každý stĺpec reprezentuje hodnotu iného atribútu. V *n*-rozmernej krížovej tabuľke ($n > 2$) stĺpec môže reprezentovať kombináciu hodnôt viac ako jedného atribútu, pričom hodnota v tabuľke vyjadruje súčet pre určité zoskupenie dvojíc atribút – hodnota. Táto reprezentácia je veľmi podobná tabuľkovému procesoru. Je jednoduché mapovať priamo hodnoty z dátového skladu na bunky krížovej tabuľky, čo využívajú aj mnohé komerčné nástroje OLAP.

Generalizované dáta môžu byť prezentované aj ***graficky***, napr. použitím stĺpcových, alebo kruhových grafov.

Ďalšou možnosťou reprezentácie sú ***logické pravidlá***. Každý záznam v generalizovanej relácii reprezentuje jednu alternatívu disjunkcie. Nakoľko jeden záznam obvykle nepokrýva všetky príklady z databázy (základnej relácie), je potrebné pridať kvantitatívnu informáciu o tom, aké percento záznamov spĺňa ľavú a pravú stranu daného pravidla. Logické pravidlo je takto asociované s kvantitatívnou informáciou a preto sa nazýva *kvantitatívne pravidlo* a je definované nasledovne.

Nech trieda objektov, ktorú je potrebné charakterizovať, sa nazýva cieľová trieda *C* a q_a je záznam v úplne generalizovanej relácii popisujúci *C*. Tzv. t-váha t_w označuje percentuálny podiel tých záznamov popisujúcich cieľovú triedu, ktoré sú pokryté q_a. Formálne $t_w = \frac{|q_a|}{\sum_{i=1}^{n} |q_i|}$ kde *n* je celková podpora (počet) záznamov cieľovej triedy, q_1, q_2, ..., q_n sú záznamy v úplne generalizovanej relácii cieľovej triedy a q_a je jeden z q_1, q_2, ..., q_n. $|q_i|$ označuje podporu záznamu q_i.

Kvantitatívne charakteristické pravidlo potom môže byť reprezentované dvojako.

- V podobe *logického pravidla*, kde každý člen disjunkcie pokrývajúci cieľovú triedu má asociovanú t-váhu.
- Vo forme relačnej *tabuľky* (úplne generalizovaná relácia), prípadne krížovej tabuľky, kde sú hodnoty podpory nahradené zodpovedajúcimi t-váhami.

Každý člen disjunkcie v kvantitatívnom charakteristickom pravidle predstavuje určitú podmienku. Tieto podmienky vo všeobecnosti *tvoria nutnú podmienku cieľovej triedy C* (t.j. hodnoty záznamu musia vyhovovať jednému z členov disjunkcie, aby daný záznam popisoval objekt z cieľovej triedy), avšak *nemusia tvoriť aj postačujúcu podmienku*, keďže záznam spĺňujúci podmienku niektorého z členov disjunkcie môže ešte patriť aj do inej než cieľovej triedy.

Preto sa kvantitatívne charakteristické pravidlo vyjadruje vo forme implikácie tvaru $\forall X, X \in C \Rightarrow podmienka_1(X)\lfloor t : t_{w_1} \rfloor \vee \cdots \vee podmienka_m(X)\lfloor t : t_{w_m} \rfloor$.

Takéto pravidlo indikuje, že ak je objekt *X* z cieľovej triedy *C*, potom s pravdepodobnosťou t_{w_i} spĺňa podmienku *podmienka*$_i$, kde t_{w_i} je t-váha podmienky *i* (*i*-teho člena disjunkcie) a *i* je z {*1*, ..., *m*}.

Príklad 3.1

Nech je daná úplne generalizovaná relácia - Tabuľka 3.7. Nakoľko ide o generalizovanú reláciu transakčnej databázy (resp. údajov z databázového skladu vytvoreného nad dátami z transakčnej databázy), podpora (početnosť) jednotlivých záznamov vlastne predstavuje počet predaných kusov jednotlivých typov tovarov. V tomto prípade je preto možné paralelne so spočítavaním podpory spočítavať aj údaje o úhrnných predajoch (tržbách).

Location	Item	Sales (in million dollars)	Count (in thousands)
Asia	TV	15	300
Europe	TV	12	250
North_America	TV	28	450
Asia	computer	120	1000
Europe	computer	150	1200
North_America	computer	200	1800

Tabuľka 3.7 Generalizovaná relácia obsahujúca počty predaných kusov i dosiahnuté tržby.

Uvedenú generalizovanú reláciu je možné prezentovať aj vo forme krížovej tabuľky. Nasleduje príklad dvojrozmernej krížovej tabuľky pre atribúty „Location“ a „Sales“ (Tabuľka 3.8).

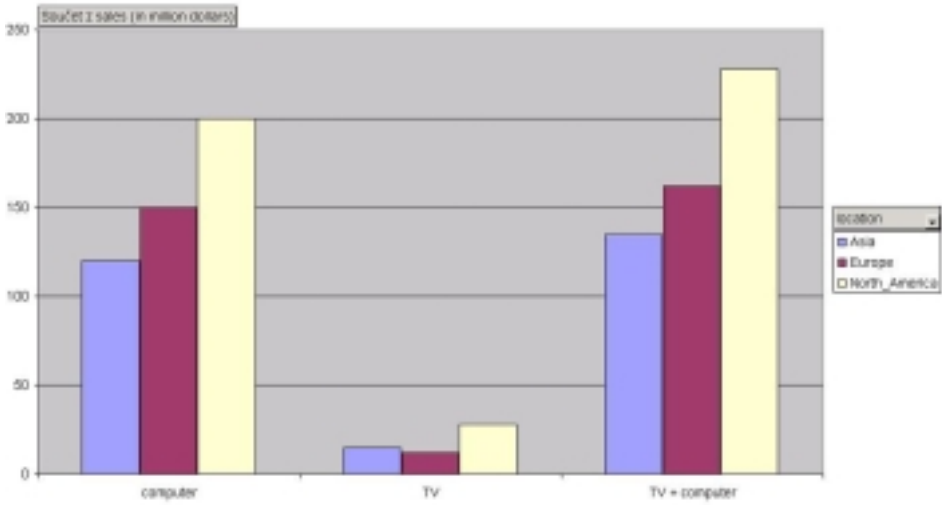
Súčet Sales	Item		
Location	computer	TV	Celkový súčet
Asia	120	15	135
Europe	150	12	162
North_America	200	28	228
Celkový súčet	470	55	525

Tabuľka 3.8 Dvojrozmerná krížová tabuľka.

Podobne je možné vygenerovať trojrozmernú krížovú tabuľku pre atribúty „Location“, „Sales“ a „Count“ (Tabuľka 3.9).

	Item					
	computer		TV		Celkový súčet Sales	Celkový súčet Count
Location	Súčet Sales	Súčet Count	Súčet Sales	Súčet Count		
Asia	120	1000	15	300	135	1300
Europe	150	1200	12	250	162	1450
North_America	200	1800	28	450	228	2250
Celkový súčet	470	4000	55	1000	525	5000

Tabuľka 3.9 Trojrozmerná krížová tabuľka.



Obr. 3.2 Stĺpcový diagram reprezentujúci predaje jednotlivých položiek v jednotlivých častiach sveta.

Ďalšou prehľadnou formou zobrazenia generalizovanej relácie je použitie rôznych typov grafov, napr. stĺpcového (Obr. 3.2) alebo koláčového (Obr. 3.3).

Poslednou vyššie uvedenou formou prezentácie generalizovaných znalostí o určitej triede objektov sú kvantitatívne charakteristické pravidlá. Ak je za cieľovú triedu v tomto príklade zvolená položka „computer“, potom je možné úplne generalizovanú reláciu zapísať vo forme takéhoto kvantitatívneho charakteristického pravidla.

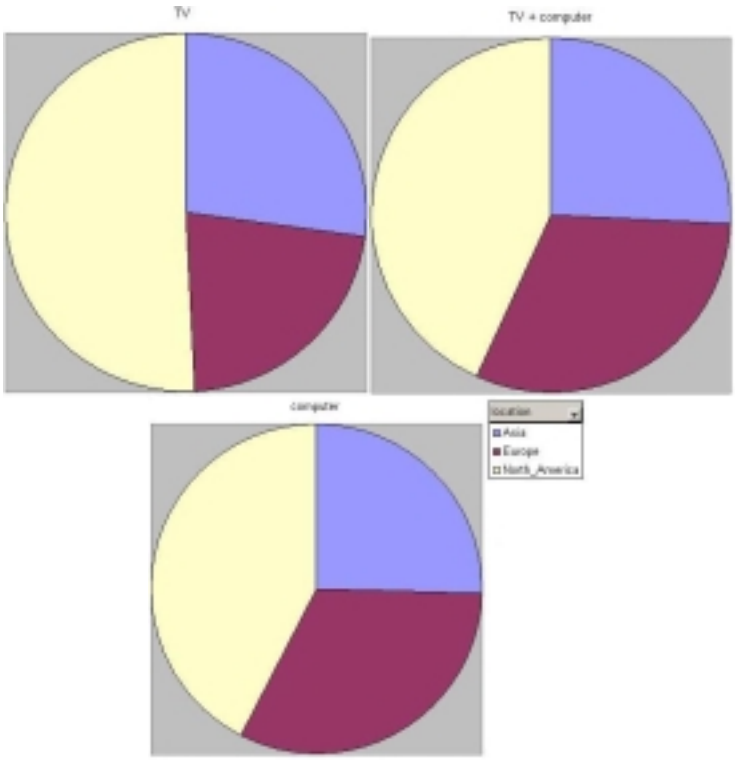
$$\forall X, item(X) = "computer" \Rightarrow$$
$$(location(X) = "Asia")[t : 25\%] \vee (location(X) = "Europa")[t : 30\%] \vee$$
$$(location(X) = "North\_America")[t : 45\%]$$

Hodnotu t-váhy pre záznam generalizovanej relácie obsahujúci dvojicu hodnôt („Asia“, „computer“) možno dostať ak sa predelí počet predaných kusov počítačov v Ázii (1000) celkovým počtom kusov predaných počítačov (4000), viď. Tabuľka 3.9Tabuľka 3.. Podobne pre ostatné dva členy disjunkcie.

3.4.2 Porovnanie

Prvé tri spôsoby prezentácie výsledkov uvedené v predchádzajúcej časti pre charakterizáciu platia rovnako aj pre prípad porovnávania dvoch alebo viacerých skupín objektov. Ide o priame zobrazenie úplne generalizovaných relácií vo forme *tabuliek*, použitie krížových tabuliek a *grafov*.

Čo sa týka logických pravidiel, pre porovnanie sa používajú tzv. ***kvantitatívne diskriminačné pravidlá***, ktoré každému záznamu z úplne generalizovanej relácie priradia ešte tzv. d-váhu d_w. Váha d_w vyjadruje pomer medzi podporou daného záznamu v cieľovej triede k celkovej podpore tohto záznamu vo všetkých triedach (v cieľovej aj v kontrastných triedach).



Obr. 3.3 Kruhové diagramy reprezentujúce predaje pre jednotlivé položky v rôznych častiach sveta.

Formálne je možné d-váhu definovať nasledovne $d_w = \frac{\left|q_a^{C_j}\right|}{\sum_{i=1}^{m}\left|q_a^{C_i}\right|}$ kde *m* je celkový počet tried (cieľová plus kontrastné), $C_j \in \{C_1,\ldots,C_m\}$ a $\left|q_a^{C_i}\right|$ je podpora záznamu q_a v triede C_i.

Vysoká hodnota d-váhy indikuje, že koncept reprezentovaný daným záznamom je primárne odvodený od cieľovej triedy, zatiaľ čo nízke hodnoty d-váhy naznačujú že záznam je odvodený primárne z kontrastných tried.

Kvantitatívne diskriminačné pravidlo pre cieľovú triedu *C* sa zapisuje nasledovne: $\forall X, X \in C \Leftarrow podmienka(X)[d : d_w]$ kde podmienka je tvorená záznamom v úplne generalizovanej relácii *C*.

Ako zo zápisu vidno, šípka implikácie teraz smeruje naopak než v prípade kvantitatívneho charakteristického pravidla. Je to preto, že diskriminačné pravidlo *poskytuje postačujúcu podmienku* pre to, aby objekt patril do cieľovej triedy, *nie však nutnú podmienku*.

To znamená napr. pri d-váhe 30% ak objekt spĺňa podmienku takéhoto pravidla, na 30% patrí do cieľovej triedy. Neplatí to ale naopak, t.j. že ak je objekt z danej triedy, tak s pravdepodobnosťou 30% bude spĺňať aj túto podmienku, keďže pravidlo nepokrýva všetky príklady cieľovej triedy.

Príklad 3.2

Dané sú úplne generalizované relácie dvoch tried z databázy o študentoch univerzity. Cieľovou triedou sú študenti, ktorý získali diplom (graduate) a kontrastnou triedou sú študenti, ktorí diplom zatiaľ nezískali (undergraduate). V oboch reláciách sa vyskytuje záznam popisujúci študentov tradičných univerzitných disciplín (Science), vo veku 21 až 25 rokov s dobrým prospechom. Podpora tohoto záznamu v každej z uvedených tried je zhrnutá v nasledujúcej tabuľke.

skupina	major subject	age_range	gpa	support (count)
graduate	Science	21 ... 25	good	90
undergraduate	Science	21 ... 25	good	210

Tabuľka 3.10 Podpora jedného záznamu v úplne generalizovanej relácii pre cieľovú triedu „graduate“ a kontrastnú triedu „undergraduate“.

Na základe uvedených údajov (Tabuľka 3.10) možno odvodiť takéto kvantitatívne diskriminačné pravidlo:

$$\forall X, status(X) = "graduate" \Leftarrow$$
$$major(X) = "Science" \wedge age\_range(X) = "21...25" \wedge gpa(X) = "good" [d : 30\%]$$

3.4.3 Charakterizácia a porovnanie

Charakterizáciu a porovnanie je možné prezentovať aj súčasne. A to napríklad pomocou spoločnej krížovej tabuľky, kde sa priamo uvedú t-váhy a d-váhy. Nech je daná databáza predajov (Tabuľka 3.7). Za cieľovú reláciu budeme považovať predaje v Európe a kontrastnou reláciou budú predaje v severnej Amerike. Krížová tabuľka, ktorá obsahuje tak charakteristické ako aj porovnávacie informácie o cieľovej relácii vo forme t-váh, resp. d-váh nasleduje (Tabuľka 3.11).

	Item								
	TV			computer			computer + TV		
Location	pod-pora	t-váha	d-váha	pod-pora	t-váha	d-váha	pod-pora	t-váha	d-váha
Europe	12	7,4%	30%	150	92,6%	42,9%	162	100%	41,5%
North_ America	28	12,3%	70%	200	87,7%	57,1%	228	100%	58,5%
Both	40	10,3%	100%	350	89,7%	100%	390	100%	100%

Tabuľka 3.11 Krížová tabuľka pre charakterizáciu aj porovnanie cieľovej triedy predajov v Európe voči predajom v severnej Amerike.

Podobne možno kombinovať prezentáciu charakteristických a diskriminačných kvantitatívnych pravidiel do spoločného zápisu pre popis vybranej cieľovej relácie objektov. Takéto pravidlá sa nazývajú ***kvantitatívne popisné pravidlá*** a možno ich formálne zapísať takto:

$$\forall X, X \in C \Leftrightarrow podmienka_1(X)\left[t : t_{w_1}, d : d_{w_1}\right] \vee \cdots \vee podmienka_m(X)\left[t : t_{w_m}, d : d_{w_m}\right]$$

Takéto pravidlo indikuje, že pre *i* od *1* po *m*, ak *X* je z cieľovej relácie *C*, potom s pravdepodobnosťou t_{w_i} spĺňa podmienku $podmienka_i$ a ak *X* spĺňa podmienku $podmienka_i$, potom s pravdepodobnosťou d_{w_i} patrí do cieľovej relácie *C*.

Napríklad na základe popisnej krížovej tabuľky pre cieľovú reláciu predajov v Európe (Tabuľka 3.11) možno zapísať takéto popisné kvantitatívne pravidlo:

$$\forall X, location(X) = "Europe" \Leftrightarrow$$
$$(item(X) = "TV")\left[t : 7{,}4\%, d : 30\%\right] \vee (item(X) = "computer")\left[t : 92{,}6\%, d : 42{,}9\%\right]$$

Toto pravidlo hovorí, že pre predaje televízorov a počítačov na základe danej databázy platí, že ak sa predaj realizoval v Európe, potom s pravdepodobnosťou 7,4% to bol televízor, zatiaľ čo s pravdepodobnosťou 92,6% to bol počítač. Na druhej strane ak porovnáme predaje v Európe s predajmi v severnej Amerike, potom 30% televízorov a 42,9% počítačov bolo predaných v Európe.

3.5 Použitá literatúra

[1] Ester, M., Sander, J.: Knowledge Discovery in Databases – Techniken und Anwendungen. Springer Verlag, (2000)

[2] Han, J., Kamber, M.: Data Mining – Concepts and Techniques. Morgan Kaufmann Publishers, (2000)

[3] Klösgen, W. and Zytkow, J.M.: The Knowledge Discovery Process. In Handbook of Data Mining and Knowledge Discovery. Oxford University Press, (2002)

[4] Wrobel, S., Morik, K., Joachims, T.: Maschinelles Lernen und Data Mining. Handbuch der Künstlichen Intelligenz, 3. vydanie, Oldenbourg Verlag München, pp. 517 – 598, (2000)

4 Prediktívne dolovanie v dátach

Prediktívne dolovanie v dátach v sebe zahŕňa dve pravdepodobne najčastejšie sa objavujúce úlohy DM, a síce klasifikáciu a predikciu. **Klasifikácia** a **predikcia** sú dve formy DM, ktoré možno použiť na extrakciu modelov popisujúcich dôležité triedy dát alebo na predikciu budúcich hodnôt dát. Hlavný rozdiel medzi klasifikáciou a predikciou spočíva v type cieľového atribútu, ktorého hodnotu je potrebné predpovedať.

Klasifikácia modeluje a predpovedá nominálne atribúty (triedy) a predikcia modeluje a predpovedá numerické hodnoty. Základný prístup je však u oboch typov DM rovnaký, t.j. väčšina prístupov sa *v prvej fáze* snaží vybudovať (naučiť sa) model správania dát na základe nejakej trénovacej množiny. Trénovacia množina obsahuje záznamy o objektoch, u ktorých je už známa hodnota cieľového atribútu.

Zostavený (naučený) model správania sa dát je potom *v druhej fáze* používaný na predpovedanie (predikciu) hodnoty cieľového atribútu u nových objektov (záznamov), na základe hodnôt ostatných atribútov, ktoré sú o objekte k dispozícii.

Typickým príkladom je vytvorenie modelu správania sa žiadateľov o úver. Banka na základe údajov o minulých žiadateľoch o úver a spôsobe ich splácania úveru môže pomocou metód DM vybudovať *klasifikačný model*, ktorý zaradí nových žiadateľov o úver do niektorej z preddefinovaných kategórií (ako napr. prideliť resp. neprideliť úver) na základe údajov, ktoré sú uvedené v žiadosti o úver. Tieto údaje totiž model vyhodnotí na základe podobných údajov minulých žiadateľov o úver.

V prípade predikcie môže ako príklad aplikácie slúžiť predpovedanie spotreby pitnej vody v určitom regióne pre najbližšie obdobie. Takúto informáciu môže spoločnosť zabezpečujúca dodávky pitnej vody využiť na to, aby dodala potrebné množstvo vody v maximálnej kvalite. Takto totiž môže naplniť zásobníkové nádrže primeraným množstvom vody. Ak voda v zásobníku chýba, viaznu dodávky pitnej vody, ak je jej naopak priveľa, jej kvalita státím v zásobníku klesá.

Jednotlivé časti tejto kapitoly uvádzajú najčastejšie používané metódy DM na klasifikáciu, resp. predikciu.

4.1 Klasifikácia

V tejto časti bude najprv úloha klasifikácie formálne definovaná. Potom budú prezentované základné metódy klasifikácie, každá v samostatnej časti, a nakoniec budú uvedené základné metódy na vyhodnotenie kvality klasifikátorov.

Nech je daná množina O objektov $o = (o_1, ..., o_d)$, kde o_i sú známe hodnoty atribútov A_i, $1 \leq i \leq d$, relevantných vzhľadom na klasifikáciu, a tiež trieda c_j, $c_j \in C = \{ c_1, ..., c_n \}$. Počet tried n je obvykle relatívne nízky. Príslušnosť k triede je spravidla vyjadrená hodnotou tzv. cieľového atribútu.

Nech D je *základný súbor objektov*, ktoré je potrebné klasifikovať. Pre každý z objektov v D sú známe hodnoty atribútov A_i, $1 \leq i \leq d$, ale príslušnosť

k triede je známa len u objektov z tzv. *trénovacej množiny* $O \subset D$. Hodnota triedy teda nie je známa u objektov z $D \setminus O$.

Klasifikátor je potom funkcia K, K: $D \rightarrow C$.

Príklad 4.1

Daná je jednoduchá databáza trénovacích dát pre zaradenie klientov poisťovne do rizikových skupín. Rizikové triedy sú pritom dve – „vysoké“, resp. „nízke“ riziko poistnej udalosti (Tabuľka 4.1).

ID	Vek	Typ auta	Riziko
1	23	rodinné	vysoké
2	17	športové	vysoké
3	43	športové	vysoké
4	68	rodinné	nízke
5	32	nákladné	nízke

Tabuľka 4.1 Trénovacia množina dát popisujúcich klientov poisťovne a ich zaradenie do rizikových tried.

V trénovacích dátach je daných päť objektov (množina O), ktoré sú popísané atribútmi A_1 = „ID“, A_2 = „Vek“, A_3 = „Typ auta“ a cieľovým atribútom C = „Riziko“. Existujú dve možné hodnoty triedy C, t.j. cieľový atribút môže nadobúdať dve rôzne hodnoty, a síce „vysoké“ a „nízke“. Z daných dát by mohol byť vytvorený jednoduchý klasifikátor K, K: $D \rightarrow C$ (D môže okrem objektov z O obsahovať ďalšie objekty, pre ktoré poznáme hodnoty atribútov A_1, A_2, A_3, ale nepoznáme hodnotu C) napr. vo forme týchto troch pravidiel:

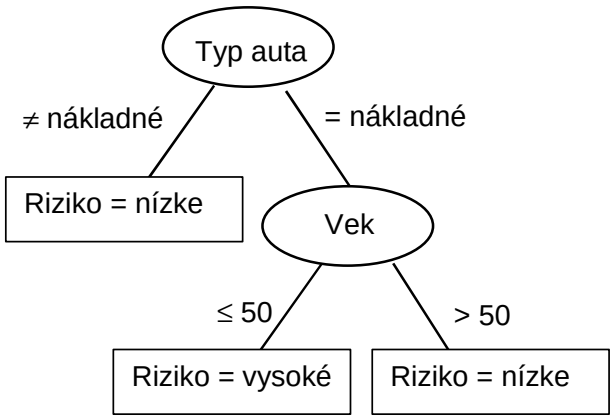
```
if Vek > 50 then Riziko = nízke;
if Vek ≤ 50 and Typ auta ≠ nákladné then Riziko = vysoké;
if Vek ≤ 50 and Typ auta = nákladné then Riziko = nízke;
```

4.1.1 Rozhodovacie stromy

Rozhodovacie stromy ([1], [2], [5], [6], [7], [11]) sú najpopulárnejšou formou reprezentácie klasifikátorov najmä pre svoju ľahko pochopiteľnú reprezentáciu získaných znalostí. Rozhodovací strom je strom s nasledujúcimi vlastnosťami:

- *Medziľahlý uzol* reprezentuje vybraný atribút (prípadne skupinu atribútov)
- *Listový uzol* reprezentuje niektorú z tried
- *Hrana* reprezentuje test na atribút (skupinu atribútov) z nadradeného uzla

Z testov, ktoré sa vzťahujú na daný medziľahlý uzol (t.j. z neho vychádzajúce hrany), môže byť úspešný vždy práve jeden. Rozhodovací strom sa konštruuje na základe objektov trénovacej množiny. Nové objekty, u ktorých nepoznáme zaradenie do triedy, potom prechádzajú týmto stromom počnúc koreňovým uzlom cez jednotlivé hrany podľa výsledkov zodpovedajúcich testov na hodnoty ostatných atribútov a končiac v jednom z listových uzlov, ktorého hodnota vlastne predpovedá novému objektu zaradenie do triedy, ktorú reprezentuje (klasifikuje ho).



Obr. 4.1 Rozhodovací strom pre trénovacie dáta – Tabuľka 4.1.

Pre vyššie uvedený Príklad 4.1 by mohol byť vygenerovaný napríklad takýto rozhodovací strom (Obr. 4.1).

Z rozhodovacieho stromu je možné jednoduchým spôsobom odvodiť aj rozhodovacie pravidlá nasledovným postupom. Pre každý listový uzol je možné zapísať jedno rozhodovacie pravidlo, ktorého podmienkovú časť bude tvoriť konjunkcia testov zodpovedajúcich hranám na ceste vedúcej od koreňového uzla k danému listovému uzlu. Takýmto postupom je možné odvodiť z rozhodovacieho stromu na Obr. 4.1 tri rozhodovacie pravidlá uvedené vyššie (Príklad 4.1).

Všeobecný algoritmus pre generovanie rozhodovacieho stromu z trénovacej množiny *T* a daným minimálnym podielom príkladov z jednej triedy v listovom uzle *min_conf* vyzerá nasledovne.

```
KonštruujRozhodovacíStrom (TrénovaciaMnožina T, Float min_conf)
     if minimálne min_conf objektov z T patrí do triedy C
     then vytvor listový uzol zaradzujúci do C;
          return;
     else
          for each atribút A do
               for each možné rozdelenie hodnôt A do
                    ohodnoť kvalitu rozdelenia, ktoré by
                         takýmto spôsobom vzniklo;
          vykonaj najlepšie zo všetkých možných rozdelení;
          nech T1, T2, …, Tm sú množiny ktoré vzniknú
               týmto rozdelením;
          KonštruujRozhodovacíStrom(T1, min_conf);
          ...
          KonštruujRozhodovacíStrom(Tm, min_conf);
          return;
```

Jednotlivé algoritmy pre konštrukciu rozhodovacích stromov sa líšia jednak stratégiou rozdeľovania aktuálnej množiny objektov, spôsobom orezávania stromov, ale aj ďalšími vlastnosťami. Orezávanie stromov má za cieľ dosiahnuť stručnejšiu reprezentáciu výsledného stromu pri zachovaní primeranej presnosti klasifikácie. Dôvodom je jav označovaný ako *preučenie*,

kedy skonštruovaný rozhodovací strom príliš podrobne kopíruje charakteristiky príkladov z trénovacej množiny, čo ale môže znížiť jeho presnosť pre nové, neznáme príklady v budúcnosti. Okrem toho menšie stromy sú prehľadnejšie a poskytujú tak ľahšie pochopiteľné znalosti.

Jeden z najčastejšie používaných algoritmov je algoritmus využívajúci teóriu informácie na výber atribútu. Tento algoritmus sa nazýva C4.5 [7] a poskytuje v mnohých aplikáciách veľmi dobré výsledky. Jeho výhodou je aj jeho schopnosť pracovať tak s nominálnymi (diskrétnymi), ako aj spojitými atribútmi (na rozdiel napr. od jeho predchodcu, algoritmu ID3 [5], ktorý pracuje len s diskrétnymi atribútmi). Tento algoritmus je implementovaný napr. aj v systémoch Weka [10] a KDD Package [3].

Algoritmus C4.5 a jeho predchodca ID3 využívajú teóriu informácie pre výber atribútu, na ktorom je založená testovacia podmienka. Teória informácií vyžíva pre meranie množstva informácie entrópiu. Ak jednotlivé správy $x_1, x_2, \ldots, x_n$ sú možné s pravdepodobnosťami $p(x_1), p(x_2), \ldots, p(x_n)$ pričom pravdepodobnosti vytvárajú úplný súbor pravdepodobností $\sum_{j=1}^{n} p(x_j) = 1$ potom entrópiu (neurčitosť) súboru správ $x_1, x_2, \ldots, x_n$ možno vyjadriť ako $H = -\sum_{j=1}^{n} p(x_j)\log_2(p(x_j))\ [bit]$ čo je vzťah známy ako Shannonova entrópia. Čím je entrópia súboru správ vyššia, tým neurčitejší je obsah správy.

Pri klasifikácii príkladov je potrebné prijať určité množstvo informácie, aby bolo možné rozhodnúť o tom, do ktorej triedy daný objekt patrí. Pred klasifikovaním príkladov je neurčitosť (entrópia) najväčšia, po ich klasifikácii sú už príkladom priradené triedy, a teda entrópia klesne na nulu. Podobne v rozhodovacom strome v koreňovom uzle je entrópia maximálna a v listových uzloch minimálna, prípadne nulová. Algoritmus využíva túto znalosť pre výber rozhodovacieho atribútu. Snahou je, aby entrópia poklesla čo najskôr, preto sa na rozdelenie množiny príkladov vyberá vždy ten atribút, ktorý minimalizuje entrópiu vzniknutého rozdelenia.

Ak klasifikačné triedy príkladov sú $c_1, c_2, \ldots, c_n$, potom *entrópia v uzle* S je $H(S) = -\sum_{j=1}^{n} p(c_j)\log_2(p(c_j))$ kde $p(c_j)$ je pravdepodobnosť, že nejaký príklad patriaci uzlu S, bude klasifikovaný do triedy c_j. Ak použitím atribútu A_i sa uzol rozvetví na m vetiev $s_1, s_2, \ldots, s_m$, tak *celková entrópia v uzle S použitím atribútu* A_i na jeho rozdelenie bude $H(S, A_i) = \sum_{j=1}^{m} p(s_j)H(S_j)$ kde $p(s_j)$ je pravdepodobnosť, že nejaký príklad, patriaci uzlu S, prešiel vetvou s_j do poduzla S_j. Odčítaním týchto dvoch entrópií sa dosiahne rozhodovacie kritérium označované ako *informačný zisk* $I(S, A_i) = H(S) - H(S, A_i)$.

Kritérium informačný zisk, používané v algoritme ID3 má závažný nedostatok v uprednostňovaní výberu testovacích podmienok s mnohými výstupmi. Ak sa v trénovacej množine nachádza diskrétny atribút A_i s rozličnými hodnotami pre každý príklad (napr. atribút ID – Príklad 4.1), tento atribút má najväčšiu šancu na výber do testovacej podmienky. Na základe atribútu A_i je rozdelená trénovacia množina na podmnožiny s práve jedným príkladom, lebo vtedy je entrópia v daných poduzloch nulová a tým aj celková entrópia

koreňového uzla použitím testovacieho atribútu A_i. Preto informačný zisk je v tomto prípade maximálny. Avšak z pohľadu klasifikácie neznámych príkladov je takýto rozhodovací strom zbytočný. Uvedený nedostatok je možné odstrániť normalizovaním informačného zisku, ako to robí algoritmus C4.5 [7]. Nech sa uzol S použitím atribútu A_i rozvetví na m vetiev, tak výsledná *pomerová entrópia* bude $H_p(S,A_i) = -\sum_{j=1}^{m} p(s_j)\log_2(p(s_j))$ a $I_p(S,A_i) = \frac{I(S,A_i)}{H_p(S,A_i)}$ je ***pomerový informačný zisk*** (používaný v algoritme C4.5).

Testovacia podmienka pre spojité atribúty pozostáva z prahovej hodnoty h, ktorá rozdeľuje usporiadanú množinu čísel na dve podmnožiny. Zostáva určiť túto prahovú hodnotu, čo algoritmus C4.5 vykonáva nasledovne. Trénovacie príklady sú najprv usporiadané vzostupne podľa daného spojitého atribútu. Keďže príkladov je konečný počet, usporiadanú množinu hodnôt spojitého atribútu je možné označiť $\{v_1, v_2, \ldots, v_k\}$. Prahová hodnota ležiaca medzi dvoma hodnotami v_i a v_{i+1}, rozdelí množinu príkladov na množiny $\{v_1, v_2, \ldots, v_i\}$ a $\{v_{i+1}, v_{i+2}, \ldots, v_k\}$. Takýchto možných rozdelení je $k-1$. Pre všetky rozdelenia sa vypočíta hodnotiaca funkcia a vyberie sa rozdelenie s maximálnym ohodnotením. Hodnotiacou funkciou môže byť informačný zisk alebo pomerový informačný zisk. Prahová hodnota medzi dvoma hodnotami spojitého atribútu v_i a v_{i+1} sa určí ako ich priemerná hodnota, t.j. $h = \frac{v_i + v_{i+1}}{2}$.

Iným kritériom pre výber testovacej podmienky je tzv. ***Gini-index*** (používaný napr. v systéme IBM Intelligent Miner). Gini-index pre množinu trénovacích príkladov T označovaný *gini(T)* možno vypočítať nasledovne $gini(T) = 1 - \sum_{i=1}^{n} p_i^2$ kde p_i je relatívna početnosť triedy c_i. Gini-index pre rozdelenie množiny T na podmnožiny T_1, T_2, ..., T_k označovaný $gini(T_1, T_2, ..., T_k)$ sa vypočíta nasledovne $gini(T_1, T_2, \ldots, T_k) = \sum_{i=1}^{k} \frac{|T_i|}{|T|} \cdot gini(T_i)$.

Príklad 4.2

V nasledujúcej tabuľke je uvedených 14 príkladov trénovacej množiny popisujúcej podmienky pre hru golfu (Tabuľka 4.2). Príklady sú popísané dvoma spojitými (A_2 = „Teplota“, A_3 = „Vlhkosť“) a dvoma diskrétnymi atribútmi (A_1 = „Počasie“, A_4 = „Vietor“) a sú rozdelené do dvoch tried (c_1 = „Hrá sa“, c_2 = „Nehrá sa“). Informácia o tom je obsiahnutá v cieľovom atribúte C = „Trieda“. Na základe tejto trénovacej množiny je potrebné vygenerovať rozhodovací strom, pričom ako kritérium pre výber testovacieho atribútu pre medziľahlé uzly má byť použitý pomerový informačný zisk. Koncová podmienka bude splnená, ak uzol obsahuje len príklady patriace do jednej triedy (t.j. min_conf = 1, teda 100%).

Po inicializácii sa v koreňovom uzle nachádzajú všetky príklady, takže tento uzol sa nemôže stať listovým a je potrebné pre neho vybrať najvhodnejší testovací atribút v zmysle kritéria pomerového informačného zisku. Preto je nutné vypočítať pomerový informačný zisk v koreňovom uzle, ktorý by sa dosiahol najlepším možným rozdelením pre každý zo štyroch existujúcich atribútov a vybrať ten s najväčšou hodnotou pomerového informačného zisku.

Počasie	Teplota [°C]	Vlhkosť [%]	Vietor?	Trieda
Slnečno	24	70	Áno	Hrá sa
Slnečno	27	90	Áno	Nehrá sa
Slnečno	29	85	Nie	Nehrá sa
Slnečno	22	95	Nie	Nehrá sa
Slnečno	21	70	Nie	Hrá sa
Zamračené	22	90	Áno	Hrá sa
Zamračené	28	78	Nie	Hrá sa
Zamračené	18	65	Áno	Hrá sa
Zamračené	27	75	Nie	Hrá sa
Dážď	22	80	Áno	Nehrá sa
Dážď	18	70	Áno	Nehrá sa
Dážď	24	80	Nie	Hrá sa
Dážď	20	80	Nie	Hrá sa
Dážď	21	96	Nie	Hrá sa

Tabuľka 4.2 Trénovacia množina príkladov (golf).

Entrópia v koreňovom uzle *je* $H(S_0) = -p(\text{Hrá sa})*\log_2(p(\text{Hrá sa})) – p(\text{Nehrá sa})*\log_2(p(\text{Nehrá sa})) = -9/14*\log_2(9/14) – 5/14*\log_2(5/14) = 0.940.$

- V prípade výberu atribútu A_1 = „Počasie“ by bolo výsledné delenie jednoznačné na 3 vetvy pre tri existujúce hodnoty tohto atribútu.

Pomerový informačný zisk za predpokladu použitia diskrétneho atribútu *Počasie* sa vypočíta nasledovným postupom.

$H(\text{Slnečno}) = -2/5*\log_2(2/5) - 3/5*\log_2(3/5) = 0.971$
$H(\text{Zamračené}) = -4/4*\log_2(4/4) = 0$
$H(\text{Dážď}) = -3/5*\log_2(3/5) - 2/5*\log_2(2/5) = 0.971$

Celková entrópia v koreňovom uzle za predpokladu, ak sa na jeho rozdelenie použije atribút Počasie $H(S_0, \text{Počasie}) = p(\text{Slnečno})*H(\text{Slnečno}) + p(\text{Zamračené})*H(\text{Zamračené}) + p(\text{Dážď})*H(\text{Dážď}) = 5/14*0.971 + 4/14*0 + 5/14*0.971 = 0.694$. Teda zodpovedajúci informačný zisk bude $I(S_0, \text{Počasie}) = H(S_0) – H(S_0, \text{Počasie}) = 0.249$.

Pre výpočet pomerového informačného zisku musíme ešte najprv určiť pomerovú entrópiu $H_p(S_0, \text{Počasie}) = -p(\text{Slnečno})*\log_2(p(\text{Slnečno})) – p(\text{Zamračené})*\log_2(p(\text{Zamračené})) – p(\text{Dážď})\log_2(p(\text{Dážď}))= -5/14*\log_2(5/14) – 4/14*\log_2(4/14) – 5/14*\log_2(5/14) = 1.577$. Takže nakoniec pomerový informačný zisk v koreňovom uzle pre prípad výberu testovacieho atribútu Počasie na rozdelenie príkladov bude $I_p(S_0, \text{Počasie}) = I(S_0, \text{Počasie}) / H_p(S_0, \text{Počasie}) =$ ***0.157***.

- V prípade výberu atribútu A_2 = „Teplota“ je možných niekoľko rozdelení, vždy však na dve vetvy podľa zvolenej hranice h, pričom jednou vetvou pôjdu príklady s hodnotou $A_2 \leq h$ a druhou potom ostatné, t.j. u ktorých $A_2 > h$.

Pre spojitý atribút *Teplota* je potrebné najprv zoradiť jeho vyskytujúce sa hodnoty a určiť možné alternatívne prahové hodnoty. Usporiadané hodnoty

tohto atribútu sú {18, 20, 21, 22, 24, 27, 28, 29} a zodpovedajúce prahové hodnoty {19, 20.5, 21.5, 23, 25.5, 27.5, 28.5}. Pre prvú prahovú hodnotu sa vypočíta pomerový informačný zisk podobne ako v predchádzajúcich výpočtoch.

$H(Teplota < 19) = -1/2*log_2(1/2) -1/2*log_2(1/2) = 1$

$H(Teplota > 19) = -4/12*log_2(4/12) - 8/12*log_2(8/12) = 0.918$

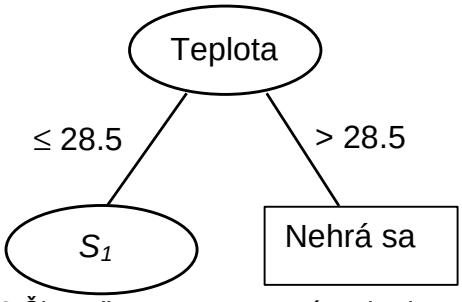
$H(S_0, Teplota(19)) = 2/14*1 + 12/14*0.918 = 0.930$

$I(S_0, Teplota(19)) = H(S_0) - H(S_0, Teplota(19)) = 0.010$

$H_p(S_0, Teplota(19)) = - 2/14*log_2(2/14) – 12/14*log_2(12/14) = 0.592$

$I_p(S_0, Teplota(19)) = I(S_0, Teplota(19)) / H_p(S_0, Teplota(19)) = 0.017$

Takýmto postupom sa vypočítajú pomerové informačné zisky pre všetky prahové hodnoty (Tabuľka 4.3). Maximálny pomerový informačný zisk ***0.304*** pre



Obr. 4.2 Čiastočne vygenerovaný rozhodovací strom

atribút *Teplota* sa dosiahne použitím prahovej hodnoty *28.5*.

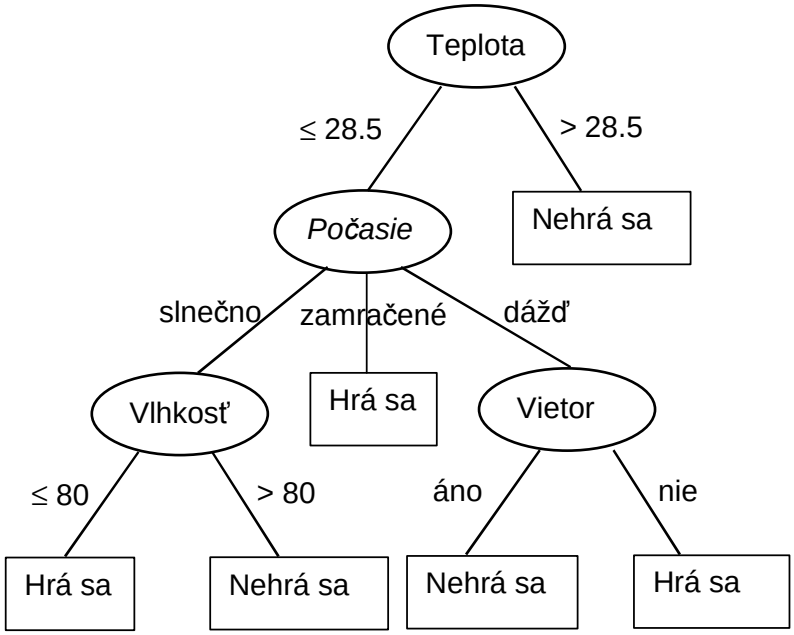
	19	20.5	21.5	23	25.5	27.5	28.5
I_p	0.027	0	0.048	0.001	0.029	0.017	0.304

Tabuľka 4.3 Pomerové zisky vypočítané pre rozdelenia zodpovedajúce jednotlivým prahovým hodnotám atribútu *Teplota*.

Informačné zisky pre rozdelenia koreňového uzla pomocou atribútu *Vlhkosť* resp. *Vietor* sa vypočítajú podobným postupom s ohľadom na typ atribútu. Výsledné hodnoty sú nasledovné: $I_p(S_0, Vlhkosť (95.5)) =$ ***0.129*** a $I_p(S_0, Vietor) =$ ***0.049***.

Maximálny pomerový informačný zisk v koreňovom uzle sa teda dosiahne použitím atribútu *Teplota* s prahovou hodnotou *28.5*. Čiastočne vygenerovaný rozhodovací strom je znázornený na Obr. 4.2. V ľavej vetve je nesplnená koncová podmienka, naopak pravá vetva je ukončená listovým uzlom s triedou *Nehrá sa* obsahujúcim jeden trénovací príklad.

Ďalší postup v uzle S_1 naznačuje Tabuľka 4.4. Po následnej voľbe atribútov s maximálnym informačným ziskom už vznikne rozhodovací strom, u ktorého každý listový uzol obsahuje už len príklady klasifikované do jednej triedy (Obr. 4.3).



Obr. 4.3 Výsledný rozhodovací strom.

Atribút	**Počasie**	**Teplota** (prah)	**Vlhkosť** (prah)	**Vietor**
$I_p(S_1, A_i)$	**0.133**	0.11 (27.5)	0.11 (95.5)	0.11
Atribút *Počasie* rozvetví uzol S_1 na $S_{1Slnečno}$, $S_{1Zamračené}$ (listový uzol) a $S_{1Dážď}$				
$I_p(S_{1Slnečno}, A_i)$	-	0.383 (25.5)	**1 (80)**	0
$I_p(S_{1Dážď}, Atr)$	-	0.446 (19)	0.446 (75)	**1**

Tabuľka 4.4 Postup pri generovaní rozhodovacieho stromu.

Už spomínané orezávanie rozhodovacích stromov môže prebiehať dvojako:

- Počas generovania rozhodovacieho stromu (tzv. pre-prunning). Ak hodnota zvolenej štatistickej mierky významnosti (χ^2, informačný zisk a pod.) pre vybraný testovací atribút nepresiahne stanovený prah, ďalšie vetvenie sa zastaví.
- Po vygenerovaní rozhodovacieho stromu (tzv. post-prunning). Toto orezávanie odstraňuje vetvy z už vygenerovaného stromu takým spôsobom, že porovnáva veľkosť očakávanej chyby klasifikácie pre daný podstrom a jeho náhradu listovým uzlom. Ak sa chyba po náhrade listovým uzlom zmenší, daný podstrom je možné odrezať.

Podrobnejší popis niektorých techník orezávania, ako aj ďalších vlastností rozhodovacích stromov je možné nájsť v [5], [6], a [7].

4.1.2 Bayesovská klasifikácia

Bayesovské klasifikátory sú štatistické klasifikátory, ktoré predikujú pravdepodobnosti, s ktorými daný príklad patrí do tej – ktorej triedy. Pritom vychádzajú z určenia podmienených pravdepodobností jednotlivých hodnôt atribútov pre rôzne triedy.

Bayesovská teoréma hovorí ako možno vypočítať podmienenú pravdepodobnosť *P(H|X)*: $P(H \mid X) = \frac{P(X \mid H) \cdot P(H)}{P(X)}$.

Nech *X* je objekt (povedzme červený a guľatý) s neznámym zaradením do triedy *C* (typ ovocia). *H* je hypotéza, že *X* patrí do triedy c_i (jablká). Cieľom klasifikácie Bayesovským klasifikátorom je vlastne určiť podmienenú pravdepodobnosť *P(H|X)*, t.j. v spomenutom prípade pravdepodobnosť toho, že guľatý a červený objekt je jablko. Ide o tzv. *posteriórnu pravdepodobnosť H* za podmienky *X*. Na rozdiel od tejto pravdepodobnosti je *P(H) apriórna pravdepodobnosť H*. V uvedenom príklade ide o pravdepodobnosť toho, že ľubovoľný objekt z databázy je jablko, bez ohľadu na to, ako ten objekt vyzerá. Je zjavné, že na určenie posteriórnej pravdepodobnosti *P(H|X)* je potrebných viac informácií (tzv. background knowledge) než na určenie apriórnej pravdepodobnosti *P(H)*, ktorá je nezávislá od *X*.

Podobne *P(X|H)* je posteriórna pravdepodobnosť *X* za podmienky *H*. T.j. ak daný objekt je jablko, na koľko je pravdepodobné, že je guľaté a červené. *P(X)* je apriórna pravdepodobnosť toho, že objekt z danej databázy je červený a guľatý.

Keďže hodnoty *P(H)*, *P(X)* a *P(X|H)* je možné odhadnúť na základe dát z danej databázy, Bayesovská teoréma poskytuje nástroj, ako vypočítať posteriórnu pravdepodobnosť *P(H|X)* z *P(H)*, *P(X)* a *P(X|H)*.

Naivný Bayesovský klasifikátor vychádza z predpokladu, že efekt, ktorý má hodnota (každého) atribútu na danú triedu, nie je ovplyvnený hodnotami ostatných atribútov. Jeho implementáciu možno nájsť napr. v systéme Weka [10].

Nech je daná množina *O* = objektov *o* = (o_1, ..., o_d). Pre každý objekt *o* sú známe jeho hodnoty atribútov A_i, $1 \le i \le d$ relevantných vzhľadom na klasifikáciu, ako aj trieda c_j, $c_j \in C = \{c_1, ..., c_n\}$. Neznámy príklad $X = (x_1, ..., x_d)$ bude klasifikovaný do triedy c_i s najväčšou posteriórnou pravdepodobnosťou $P(c_i|X) > P(c_j|X)$ $\forall 1 \le j \le n$, $i \ne j$.

Keďže $P(c_i \mid X) = \frac{P(X \mid c_i) \cdot P(c_i)}{P(X)}$ a *P(X)* je konštantná pre všetky triedy c_i, stačí nájsť minimálnu hodnotu výrazu $P(X \mid c_i) \cdot P(c_i)$ spomedzi všetkých tried c_i. Pravdepodobnosť zaradenia ľubovoľného objektu do triedy c_i je $P(c_i) = \frac{N_O^i}{N_O}$, kde N_O je počet všetkých príkladov z trénovacej množiny *O* a N_O^i je počet tých príkladov z *O*, ktoré patria do triedy c_i.

Ostáva teda určiť pravdepodobnosti $P(X|c_i)$. Tieto pri predpoklade nezávislosti jednotlivých atribútov A_i možno vypočítať nasledovne:

- Pre *kategorické atribúty* $P(X \mid c_i) = \prod_{k=1}^{d} P(x_k \mid c_i)$, kde $P(x_k \mid c_i) = \frac{N_O^{i,k}}{N_O^i}$, pričom N_O^i je počet tých príkladov z *O*, ktoré patria do triedy c_i a $N_O^{i,k}$ je počet tých z nich, pre ktoré hodnota atribútu $A_k = x_k$.
- Pre *spojité atribúty* A_k sa obvykle predpokladá Gaussovo normálne rozdelenie hodnôt, a potom $P(X \mid c_i) = \frac{1}{\sqrt{2\pi}\sigma_{c_i}} \cdot e^{-\frac{x_k - \mu_{c_i}}{2\sigma^2_{c_i}}}$, kde μ_{c_i} je stredná hodnota a σ_{c_i} je rozptyl hodnôt atribútu A_k z tých príkladov trénovacej množiny, ktoré patria do triedy c_i.

Príklad 4.3

Daná je nasledovná trénovacia množina *O* (Tabuľka 4.5). Objekty sú popísané štyrmi významovými atribútmi (*vek*, príjem, *študent* a *kreditné ohodnotenie*). Každý z objektov (zákazníkov obchodu s elektronikou) je zaradený do jednej z dvoch tried – kúpil si počítač, resp. nekúpil si počítač. Je potrebné navrhnúť naivný Bayesovský klasifikátor a klasifikovať ním nového zákazníka s nasledovnými hodnotami atribútov X = (vek = „≤ 30“, príjem = „stredný“, študent = „áno“, kreditné ohodnotenie = „priemerné“).

ID	vek	príjem	študent	kreditné ohodnotenie	trieda (kúpi si počítač)
1	≤ 30	vysoký	nie	priemerné	nie
2	≤ 30	vysoký	nie	výborné	nie
3	31 .. 40	vysoký	nie	priemerné	áno
4	> 40	stredný	nie	priemerné	áno
5	> 40	nízky	áno	priemerné	áno
6	> 40	nízky	áno	výborné	nie
7	31 .. 40	nízky	áno	výborné	áno
8	≤ 30	stredný	nie	priemerné	nie
9	≤ 30	nízky	áno	priemerné	áno
10	> 40	stredný	áno	priemerné	áno
11	≤ 30	stredný	áno	výborné	áno
12	31 .. 40	stredný	nie	výborné	áno
13	31 .. 40	vysoký	áno	priemerné	áno
14	> 40	stredný	nie	výborné	nie

Tabuľka 4.5 Trénovacia množina príkladov z databázy zákazníkov obchodu s elektronikou.

Je potrebné teda zistiť, pre ktorú z dvoch daných tried je hodnota súčinu $P(X \mid c_i) \cdot P(c_i)$ maximálna. Apriórne pravdepodobnosti jednotlivých tried $P(c_i)$ možno jednoducho určiť z trénovacej množiny.

P(kúpi si počítač) = 9/14 = 0.643 a *P(nekúpi si počítač)* = 5/14 = 0.357.

Pre výpočet posteriórnych pravdepodobností $P(X|c_i)$ je potrebné najskôr vypočítať nasledovné podmienené pravdepodobnosti je jednotlivé hodnoty atribútov A_i.

P(vek = „≤ 30“ | kúpi si počítač) = 2/9 = 0.222
P(vek = „≤ 30“ | nekúpi si počítač) = 3/5 = 0.600
P(príjem = „stedný“ | kúpi si počítač) = 4/9 = 0.444
P(príjem = „stedný“ | nekúpi si počítač) = 2/5 = 0.400
P(študent = „áno“ | kúpi si počítač) = 6/9 = 0.667
P(študent = „áno“ | nekúpi si počítač) = 1/5 = 0.200
P(kreditné ohodnotenie = „priemerné“ | kúpi si počítač) = 6/9 = 0.667
P(kreditné ohodnotenie = „priemerné“ | nekúpi si počítač) = 2/5 = 0.400

Použitím týchto pravdepodobností možno vypočítať:

P(X | kúpi si počítač) = 0.222 x 0.444 x 0.667 x 0.667 = 0.044
P(X | nekúpi si počítač) = 0.600 x 0.400 x 0.200 x 0.400 = 0.019

a teda

P(X | kúpi si počítač) x P(kúpi si počítač) = 0.044 x 0.643 = 0.028
P(X | nekúpi si počítač) x P(nekúpi si počítač) = 0.019 x 0.357 = 0.007

To znamená, že naivný Bayesovský klasifikátor bude klasifikovať daný objekt do triedy c_1 = „kúpi si počítač“.

4.1.3 Klasifikátory na princípe k-najbližších susedov

Klasifikátory z tejto skupiny sú založené na princípe učenia sa na základe analógie. Spravidla sa predpokladá, že trénovacie príklady (množina *O*) sú popísané *n* numerickými atribútmi, a teda predstavujú vlastne body v *n*-dimenzionálnom priestore príkladov.

Ak príde nový príklad *q* s ešte neznámou hodnotou cieľového atribútu (triedy), klasifikátor na princípe *k*-najbližších susedov hľadá v priestore príkladov takých *k* trénovacích príkladov, ktoré sú k novému príkladu najbližšie. Pritom vzdialenosť príkladov môže byť definovaná napr. ako euklidovská vzdialenosť dvoch bodov $X = (x_1, x_2, ..., x_n)$ a $Y = (y_1, y_2, ..., y_n)$, t.j. $d(X,Y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$. Jedna z implementácií tohto postupu je aj v systéme Weka [10].

Neznámy príklad je potom klasifikovaný do tej triedy, ktorá sa najčastejšie vyskytuje medzi *k* jemu najbližšími trénovacími príkladmi. Z uvedeného vyplýva, že tento typ klasifikátorov nekonštruuje žiaden klasifikačný model, t.j. odpadá vlastne prvá fáza klasifikácie, konštrukcia modelu. Ku klasifikácii tu dochádza až v momente, keď je potrebné klasifikovať nejaký nový, dovtedy neznámy príklad. Takéto klasifikátory sa zvyknú v literatúre nazývať aj klasifikátory založené na inštanciách (instance-based), resp. sa hovorí o „lenivom učení“ (lazy learners). Nevýhodou takýchto prístupov je skutočnosť, že potrebujú dlhší čas na klasifikáciu, keďže celá činnosť začína až v momente príchodu nového príkladu a spravidla je potrebné prehľadať veľkú časť priestoru trénovacích príkladov. Tu sa zvyknú používať rôzne efektívne indexovacie techniky, aby bolo možné v trénovacom priestore rýchle nájsť najbližších susedov klasifikovaného príkladu.

Jednotlivé varianty klasifikátorov na princípe k-najbližších susedov sa môžu líšiť v tom, akým spôsobom vyberajú najbližších susedov, koľko ich vyberú pre klasifikáciu nového príkladu a ako vplývajú jednotliví susedia na rozhodnutie o výslednej triede nového príkladu. Niektoré z používaných klasifikátorov na princípe *k*-najbližších susedov pracujú nasledovne.

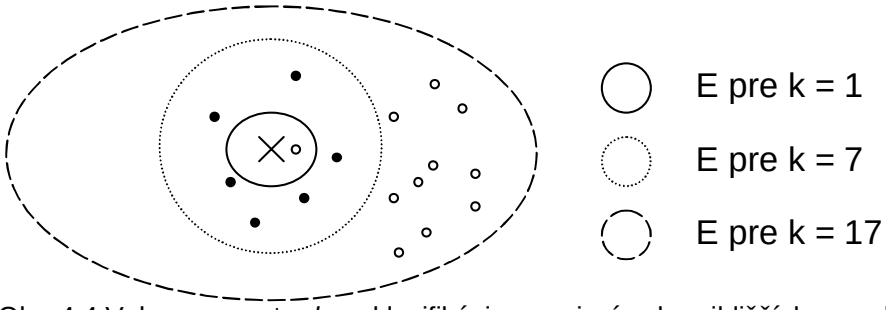
- Ako trénovacie príklady sa používajú len vektory stredných hodnôt pre jednotlivé triedy, t.j. trénovacie príklady z jednej triedy sa akoby nahradia len jedným príkladom, ktorého poloha bude vlastne aritmetickým stredom hodnôt všetkých pôvodných trénovacích príkladov z tejto triedy.
- Pre rozhodnutie o zaradení nového príkladu do triedy sa berie do úvahy len jeho najbližší sused v priestore trénovacích príkladov (t.j. $k = 1$).
- Vplyv jednotlivých susedov na rozhodnutie o klasifikačnej triede nového príkladu nie je rovnaký, ale závisí od ich vzdialenosti. To znamená, že trieda bližších susedov zaváži vo výslednom rozhodnutí viac, ako trieda vzdialenejších susedov.

Všeobecný algoritmus pre klasifikátor na princípe k-najbližších susedov môže vyzerať nasledovne.

```
Klasifikátor_k-NajbližšíchSusedov (TrénovaciePríklady O,
                                   Objekt q, Integer k)
    Vyber ako rozhodovaciu množinu E k-najbližších susedov
       objektu q z množiny O;
    Trieda = arg max_{c_j ∈ C} Σ_{o ∈ E} w(d(o,q)) · δ(c_j, Trieda(o)) ;
    return Trieda;
```

Pričom *Trieda(o)* označuje triedu trénovacieho príkladu *o*, *w* je váhová funkcia (napr. $w(x) = \frac{1}{x^2}$), δ je funkcia $\delta(x, y) = \begin{cases} 1 \Leftrightarrow x = y \\ 0 \Leftrightarrow x \neq y \end{cases}$ a *dist(x,y)* je funkcia vzdialenosti, napr. euklidovská.

Voľba parametra *k* má významný vplyv na kvalitu výsledného klasifikátora na princípe *k*-najbližších susedov (viď. Obr. 4.4).



Obr. 4.4 Vplyv parametra *k* na klasifikáciu na princípe k-najbližších susedov.

- Príliš malé *k* vedie k prílišnej citlivosti na príklady, ktoré predstavujú šumy.
- Príliš veľké *k* zase zvyšuje riziko prekročenia hraníc zhluku príkladov reprezentujúcich určitú triedu a zahrnutie mnohých príkladov z inej triedy.
- Stredná hodnota *k* preto vo všeobecnosti poskytuje najlepšie výsledky klasifikácie. Často platí pre hodnotu najlepšieho *k*: *1 << k < 10*.

4.1.4 Vyhodnotenie kvality klasifikátorov

Je zrejmé, že na danej trénovacej množine možno v závislosti od použitého algoritmu, resp. v závislosti od nastavenia jeho parametrov, získať mnohé navzájom rôzne klasifikátory. Je preto potrebné vedieť ich navzájom porovnať a zistiť, ktorý z nich je najlepší. Ako najdôležitejšie kritérium porovnania sa používa ***chyba klasifikácie***, t.j. podiel chybne klasifikovaných objektov. Vzniká ale otázka, ktoré dáta sa majú použiť pre odhad chyby klasifikácie.

Ak by sa na tento účel použili trénovacie dáta, obvykle by sa získali veľmi nízke hodnoty klasifikačnej chyby, keďže získaný klasifikátor je naučený, a teda vlastne optimalizovaný práve na trénovacie dáta. Tento efekt sa zvykne nazývať preučenie (overfitting).

Namiesto toho je možné množinu všetkých známych objektov *O* rozdeliť na dve podmnožiny.

- ***Trénovaciu množinu***, ktorá sa použije v procese budovania klasifikátora (t.j. pre učenie). Často sa zvykne na tento účel vybrať zhruba dve tretiny príkladov z *O*.
- ***Testovaciu množinu***, ktorá sa používa len na odhad chyby klasifikácie pre získaný klasifikátor. Spravidla ide o jednu tretinu príkladov z *O*.

Táto metóda sa nazýva ***trénovanie a testovanie***. Nemožno ju ale napr. dobre použiť v prípade, keď počet objektov so známou hodnotou triedy je malý a nestačí na naučenie kvalitného klasifikátora. Vtedy, ale aj vo všeobecnosti pre získanie spoľahlivejšieho odhadu chyby klasifikátora sa zvykne používať tzv. *m-násobná krížová validácia* (*m-fold cross validation*).

Pri ***m-násobnej krížovej validácii*** sa množina *O* rozdelí na *m* rovnako veľkých podmnožín, z ktorých sa zakaždým použije *m–1* podmnožín na trénovanie klasifikátora a zvyšná podmnožina potom následne na jeho testovanie. Takto sa získa *m* rôznych chýb klasifikátora, ktoré sa nakoniec skombinujú pre získanie výsledného odhadu chyby klasifikácie pri použití daného algoritmu a daného nastavenia jeho parametrov. Formálne možno tento postup zapísať nasledovne.

Ak rozdelenie množiny *O* na podmnožiny nie je náhodné, ale také, aby jednotlivé podmnožiny zachovávali distribúciu jednotlivých tried v každej z podmnožín, ide o tzv. ***rozvrstvenú násobnú krížovú validáciu*** (stratified cross validation). To znamená, že podiel príkladov z jednotlivých tried je rovnaký tak v pôvodnej množine *O*, ako aj v každej z jej podmnožín $O_1, ..., O_m$.

Vo všeobecnosti sa najčastejšie doporučuje 10-násobná krížová validácia na odhad kvality klasifikátorov.

```
KrížováValidácia (Databáza O, Integer m, Procedúra klasifikácia)
     Rozdeľ O na m podmnožín O1, ..., Om pokiaľ možno rovnakej
        Veľkosti;
     chyba = 0;
     for i from 1 to m
          klasifikátori := klasifikácia(O1 ∪..∪ Oi-1 ∪ Oi+1 ∪ Om)
          Nech chybai je chyba klasifikácie klasifikátori na Oi;
          chyba = chyba + chybai;
     return chyba/m;
```

4.1.5 Zvyšovanie presnosti klasifikátorov

Pri popisovaní klasifikácie pomocou rozhodovacích stromov bolo spomenuté, že na zvýšenie presnosti výsledného rozhodovacieho stromu sa zvyknú používať rôzne techniky orezávania ([5], [6]). Ale existujú aj všeobecné techniky na zvyšovanie presnosti klasifikátorov. Takýmito technikami sú napríklad *bagging* a *boosting*. Obe techniky sa snažia využiť množinu T klasifikátorov C_1, C_2, ..., C_T v snahe vytvoriť lepší, zložený klasifikátor C_*.

Bagging pracuje pre danú množinu O objektov o nasledovne: v každej iterácii t (t = 1, 2, ..., T) sa najprv z množiny O vytvorí vzorka O_t s nahradením, t.j. niektoré príklady sa môžu vyskytnúť vo vzorke O_t aj viackrát a iné vôbec nie. Následne sa množina O_t použije ako trénovacia množina pre získanie klasifikátora C_t.

Pre klasifikáciu neznámeho príkladu X zložený klasifikátor C_* najprv zistí klasifikácie všetkých čiastkových klasifikátorov C_t (t = 1, 2, ..., T), spočíta hlasy pre jednotlivé triedy a príklad X nakoniec klasifikuje do tej triedy, ktorá získala najväčší počet hlasov.

Pri ***boostingu*** je každému príkladu v trénovacej množine priradená určitá váha. Potom sa trénuje séria klasifikátorov C_t (t = 1, 2, ..., T) takým spôsobom, že po vygenerovaní každého klasifikátora C_t sa upravia váhy príkladov v trénovacej množine tak, aby pri sa pri učení nasledujúceho klasifikátora C_{t+1} venovala zvýšená pozornosť predtým chybne klasifikovaným príkladom. Výsledný zložený klasifikátor C_* kombinuje hlasy jednotlivých klasifikátorov tak, že váha hlasu každého z klasifikátorov je priamo úmerná jeho presnosti.

4.2 Predikcia

Ak je potrebné predpovedať hodnotu spojitého (numerického) cieľového atribútu, hovorí sa o predikcii. V tejto podkapitole sú uvedené niektoré vybrané metódy predikcie.

4.2.1 Regresia

Mnoho predikčných úloh je možné riešiť pomocou lineárnej regresie, či už priamo alebo po určitej transformácii premenných, ktorou sa nelineárny problém zmení na lineárny.

V lineárnej regresii sú dáta aproximované (modelované) pomocou priamky. ***Lineárna regresia*** je najjednoduchšou formou regresie. *Dvojrozmerná*

lineárna regresia modeluje cieľový atribút *Y* (predikovaný atribút) ako lineárnu funkciu iného, známeho atribútu *X* (tzv. predikujúci atribút), teda $Y = \alpha + \beta \cdot X$. Ide vlastne o rovnicu priamky, pričom α je posun jej priesečníka s *y*-ovou osou oproti počiatku súradnicovej sústavy a β je sklon priamky vzhľadom k *x*-ovej osi. Regresné koeficienty α, β možno vypočítať metódou najmenších štvorcov, ktorá minimalizuje chybu medzi skutočnými dátami a aproximačnou priamkou.

Ak sú dané trénovacie dáta vo forme bodov (x_1, y_1), (x_2, y_2), ..., (x_n, y_n), potom regresné koeficienty α, β možno odhadnúť uvedenou metódou pomocou nasledovných vzťahov: $\beta = \frac{\sum_{i=1}^{n}(x_i - \bar{x}) \cdot (y - \bar{y})}{\sum_{i=1}^{n}(x_i - \bar{x})^2}$, $\alpha = \bar{y} - \beta \cdot \bar{x}$, kde $\bar{x}$ je priemer hodnôt x_1, x_2, ..., x_n a $\bar{y}$ je priemer hodnôt y_1, y_2, ..., y_n.

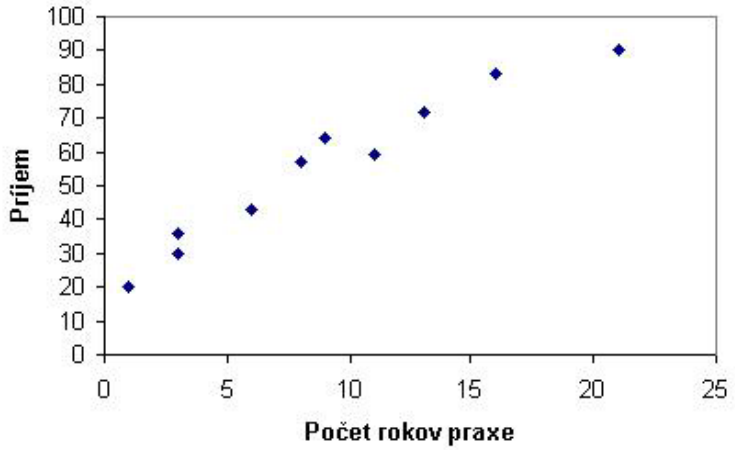
Príklad 4.4

Daná je tabuľka dvojrozmerných hodnôt (Tabuľka 4.6). Premenná *X* označuje počet rokov praxe daného absolventa vysokej školy a *Y* vyjadruje výšku jeho zárobku v tisícoch dolárov ročne.

Počet rokov praxe	3	8	9	13	3	6	11	21	1	16
Príjem	30	57	64	72	36	43	59	90	20	83

Tabuľka 4.6 Vstupné dáta o príjmoch absolventov vysokej školy.

Po znázornení daných dát vo forme grafu (Obr. 4.5) vidno, že závislosť medzi premennými *X* a *Y* je približne lineárna, a tak na predikciu výšky zárobku



Obr. 4.5 Graf nakreslený na základe dát z predchádzajúcej tabuľky

na základe informácie o počte rokov praxe je možné použiť lineárny model. Na nájdenie optimálne aproximujúcej priamky možno použiť lineárnu regresiu, a teda priamku v tvare $Y = \alpha + \beta \cdot X$.

Z daných dát (Tabuľka 4.6) možno vypočítať $\bar{x}$ = 9.1 a $\bar{y}$ = 55.4. Po dosadení do vzorcov pre výpočet koeficientov regresnej priamky bude:

$$\beta = \frac{(3-9.1)\cdot(30-55.4)+(8-9.1)\cdot(57-55.4)+\cdots+(16-9.1)\cdot(83-55.4)}{(3-9.1)^2+(8-9.1)^2+\cdots+(16-9.1)^2} = 3.5$$

a $\alpha = 55.4-(3.5)\cdot(9.1) = 23.6$, takže rovnica priamky podľa metódy najmenších štvorcov bude $Y = 23.6 + 3.5X$.

Použitím tejto rovnice možno napr. predikovať, že absolvent danej vysokej školy s 10-ročnou praxou by mal zarábať okolo 58.600,- dolárov ročne.

Viacnásobná regresia je rozšírením lineárnej regresie na viac ako jeden predikujúci atribút. Ak je predikovaný atribút *Y* lineárne závislý na atribútoch X_1, ..., X_n, potom je možné tento vzťah popísať polynómom prvého stupňa, t.j. priamkou v *n*-rozmernom priestore. Tento vzťah pre *i*-ty príklad má tvar $Y_i = a_0 + a_1 x_{1i} + a_2 x_{2i} + ... + a_n x_{ni}$ kde Y_i je predikovaná hodnota cieľového atribútu *Y* pre *i*-ty príklad v trénovacej množine (*i* = 1, ..., *k*) dát a x_{ji} je číselná hodnota atribútu X_j pre *i*-tý príklad z trénovacej množiny.

Metóda najmenších štvorcov minimalizuje súčet štvorcov odchýliek skutočných hodnôt atribútu *Y* od regresnou priamkou predikovaných hodnôt. Teda chybu *Z* možno vyjadriť ako $Z = \sum_i (Y_i - y_i)^2$, kde y_i sú skutočné hodnoty cieľového atribútu *Y* pre všetky príklady *i* = 1 .. *n* z trénovacej množiny. Cieľom metódy je minimalizovať kvadratickú chybu *Z*, teda dosiahnuť aby jej derivácia bola nulová, t.j. $Z \to min$, resp. $\frac{dZ}{dX} = 0$.

Z tejto požiadavky potom vyplynie sústava rovníc $(X^T X)a^T = X^T y$ kde *X* je dátová matica, *a* je vektor parametrov a *y* je dátový vektor (hodnoty cieľového atribútu), t.j. $X = \begin{bmatrix} 1 & x_{11} & x_{12} & ... & x_{1k} \\ 1 & x_{21} & x_{22} & ... & x_{2k} \\ ... & ... & ... & ... & ... \\ 1 & x_{n1} & x_{n2} & ... & x_{nk} \end{bmatrix}$, $y = \begin{bmatrix} y_1 \\ y_2 \\ ... \\ y_n \end{bmatrix}$, $a^T = [a_0 \quad a_1 \quad a_2 \quad ... \quad a_k]$.

Hodnoty hľadaných parametrov a_0, ..., a_k možno získať riešením uvedenej sústavy rovníc napr. pomocou LU rozkladu nasledovným postupom.

Nech *A p* = *b*, pričom $A = X^T X$, $p = a^T$ a $b = X^T y$. LU rozklad vyjadrí maticu *A* ako súčin dolnej trojuholníkovej matice *L* a hornej trojuholníkovej matice *U*, t.j. *A* = *L U*. Pritom dolná trojuholníková matica má nenulové prvky len na hlavnej diagonále a pod ňou, t.j. $L = \begin{bmatrix} l_{11} & 0 & ... & 0 \\ l_{21} & l_{22} & ... & 0 \\ ... & ... & ... & ... \\ l_{n1} & l_{n2} & ... & l_{nn} \end{bmatrix}$ a naopak horná trojuholníková matica má nenulové prvky len na hlavnej diagonále a nad ňou,

t.j. $U = \begin{bmatrix} u_{11} & u_{12} & ... & u_{1n} \\ 0 & u_{22} & ... & u_{2n} \\ ... & ... & ... & ... \\ 0 & 0 & ... & u_{nn} \end{bmatrix}$. Takže po úspešnom LU rozklade matice *A* možno sústavu rovníc *L U p = b* riešiť v dvoch krokoch.

- Najprv sa rieši systém *L q = b* tak, že z prvej rovnice sa vypočíta neznáma q_1, dosadí sa do druhej rovnice, odtiaľ sa vypočíta q_2, atď. až po poslednú z rovníc, z ktorej po dosadení už vtedy známych hodnôt q_1, q_2,...,q_{n-1} vyjde q_n.
- Analogicky sa rieši systém *U p = q*, ale tento krát sa postupuje od poslednej rovnice, z ktorej sa najprv vypočíta p_n, až k prvej, kde sa určí p_1.

Polynomiálna regresia vznikne pridaním polynomiálnych termov k základnému lineárnemu regresnému modelu. Aplikovaním transformácií na jednotlivé nelineárne členy polynómu je možné konvertovať nelineárny model na lineárny a tento potom riešiť lineárnou, resp. viacnásobnou regresiou (viď. vyššie).

Príklad 4.5

Cieľom je transformovať polynomiálny regresný model $Y = \alpha + \beta_1 X + \beta_2 X^2 + \beta_3 X^3$ na lineárny.

Ak sa použije nasledovná substitúcia: $X_1 = X$, $X_2 = X^2$, $X_3 = X^3$, potom je možné daný polynomiálny regresný model transformovať na viacnásobný lineárny model tvaru $Y = \alpha + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_3$, ktorý je riešiteľný metódou najmenších štvorcov.

4.2.2 Modelové stromy

Pre riešenie predikcie v prípade úloh, kde nie je možná transformácia na lineárny model je vhodné použiť napríklad algoritmus M5' [9], ktorý vytvára predikčné modely vo forme stromovej štruktúry (tzv. *modelové stromy*), veľmi podobnej rozhodovacím stromom (viď, predchádzajúca podkapitola venovaná klasifikácii). Medziľahlé uzly opäť zodpovedajú testom na numerické atribúty s dvoma vetvami. Listové uzly ale obsahujú lineárne regresné modely.

Vetvením v medziľahlých uzloch pre jednotlivé intervaly hodnôt predikujúcich atribútov sa dosahuje vlastne rozdelenie priestoru na disjunktné oblasti, v ktorých je možné závislosť medzi predikujúcimi atribútmi a cieľovým, predikovaným atribútom vyjadriť pomocou lineárneho modelu. Výsledok je vlastne aproximácia priebehu predikovaného atribútu pomocou lineárnej lomenej funkcie.

Algoritmus M5' popísaný v [9] vychádza z pôvodného Quinlanovho prístupu [8] a je implementovaný napr. v systéme Weka [10] a v systéme KDD Package [3], kde príslušný modul umožňuje realizovať aj jednoduchú lineárnu resp. viacnásobnú lineárnu regresiu. Algoritmus M5' po nevyhnutnej *príprave dát* (výber najvhodnejších atribútov a ich následná normalizácia), pozostáva z troch základných krokov.

1. *Vytvorenie modelového stromu*
2. *Orezávanie* modelového stromu
3. *Vyhladzovanie* modelového stromu

Nakoniec sa ako v prípade klasifikácie ešte *vyhodnotí presnosť* predikcie na nejakej testovacej množine. V etape ***vytvárania modelového stromu*** sa ako kritérium pre výber testovacieho atribútu používa tzv. *miera redukcie štandardnej odchýlky* (*standard deviation reduction* - *SDR*) daná vzťahom $SDR = \sigma(T) - \sum_{i=1}^{2} \frac{|T_i|}{|T|} \cdot \sigma(T_i)$ kde T je množina príkladov prislúchajúca danému vrcholu a T_1, T_2 sú množiny príkladov, ktoré vzniknú štiepením množiny T podľa zvoleného atribútu. $\sigma(T)$ je štandardná odchýlka predikovaného atribútu vypočítaná z dát množiny T.

Samotný algoritmus vytvárania modelového stromu vyzerá nasledovne.

```
VytvorModelovýStrom (Databáza T, Float StdOdchylkaPreCelySubor)
     if |T| < 4 or σ(T) < 0.05*StdOdchylkaPreCelySubor then
          return VytvorLinearnyModel(T);
     else
          Vypočítaj pre všetky atribúty a všetky ich možné
               pozície štiepenia výslednú SDR;
          Podľa najlepšieho nájdeného štiepenia rozdeľ T na
               T1 a T2, vytvor medziľahlý uzol Uzol s príslušným
               testovacím atribútom a dvoma poduzlami V1 a V2,
               podľa najlepšej pozície štiepenia;
          V1=VytvorModelovýStrom(T1, StdOdchylkaPreCelySubor);
          V2=VytvorModelovýStrom(T2, StdOdchylkaPreCelySubor);
     return Uzol;
```

Aby sa eliminoval efekt preučenia, výsledný modelový strom je potrebné ***orezať***. Na to slúži nasledujúca procedúra.

```
OrežModelovýStrom (Strom S)
     if Strom je listový uzol then
          return;
     else // Strom začína medziľahlým uzlom s poduzlami V1 a V2
          OrežModelovýStrom(V1);
          OrežModelovýStrom(V2);
     Nech T sú všetky príklady z S;
     LM = VytvorLinearnyModel(T);
     if ChybaPredikcie(LM) < ChybaPredikcie(S) then
          return LM;
     else
          return S;
```

Výpočet chyby predikcie modelového stromu je uvedený v algoritme **ChybaPredikcie** (Strom S) nižšie.

Modelový strom, ako bolo spomínané vyššie, v podstate tvorí po častiach lomené čiary v *n*-rozmernom priestore, pričom jednotlivé body zlomu sú definované v testoch medziľahlých uzlov stromu a priame úseky zase lineárnymi modelmi v konkrétnych listových uzloch. Vytvorená čiara však

nemusí byť nutne spojitá, závisí to od rozloženia bodov trénovacej množiny v spomínanom priestore. Aby predikčný model pokrýval celý rozsah hodnôt, je potrebné ho ešte ***vyhladiť***.

```
ChybaPredikcie (Strom S)
     if Strom je listový uzol then
          z = 0;
          n = PočetPríkladov(S);
          for each príklad oi = (x1i,x2i, ..., xni, yi) z S do
               zi = ¦yi – Predikuj(oi,S)¦;
               z = z + zi;
          return z/n;
     else // Strom začína medziľahlým uzlom s poduzlami V1 a V2
          z1 = ChybaPredikcie(V1);
          p1 = PočetPríkladov(V1);
          z2 = ChybaPredikcie(V2);
          p2 = PočetPríkladov(V2);
          z = z1 p1 + z2 p2;
     return z/n;
```

Pri vyhladzovaní modelových stromov sa najprv využívajú lineárne modely v listových uzloch na výpočet predikovanej hodnoty, potom sa filtrujú tieto hodnoty cestou späť ku koreňovému uzlu, pričom sa zakaždým vyhladzuje aktuálny uzol modelového stromu kombinovaním predikovanej hodnoty (vypočítanej z lineárnych modelov v listových uzloch) a lineárneho modelu, ktorý by prislúchal priamo danému uzlu.

Vzťah pre vyhladzovanie je $p' = \frac{np + kq}{n + k}$, kde *p'* je predikcia pre nasledujúci rodičovský uzol, *p* je predikcia daného uzla určená z jeho listov, *q* je predikovaná hodnota pre daný uzol, *n* je počet príkladov daného uzla a *k* je konštanta (napr. v systéme Weka štandardne nastavená na empiricky určenú hodnotu 15).

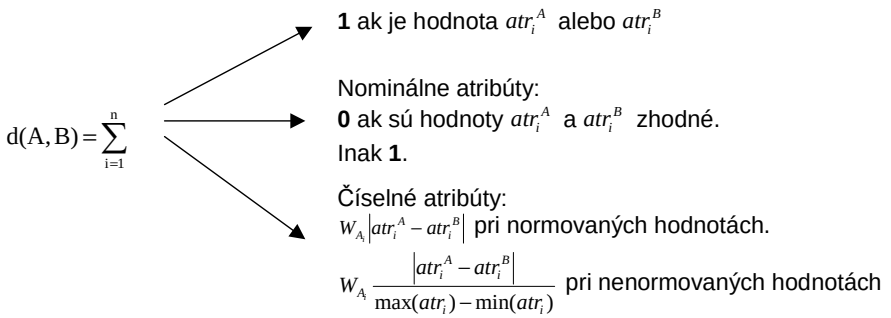
4.2.3 Prediktory na princípe k-najbližších susedov

Analogicky ako v prípade klasifikácie (viď. predchádzajúca podkapitola), aj predikciu je možné robiť na základe prípadového uvažovania, kedy sa nekonštruuje špeciálny klasifikátor, ale výsledná predikcia sa robí na základe hodnôt cieľového (predikovaného atribútu) *k*-najbližších príkladov z trénovacej množiny k práve predikovanému príkladu.

V podstate platí všetko to, čo bolo uvedené v prípade klasifikácie s tým rozdielom, že výsledná predikovaná hodnota sa vypočíta ako priemer (resp. vážený priemer) hodnôt *k*-najbližších susedov z trénovacej množiny.

Problémom zostáva, akú hodnotu *k* zvoliť a ako nastaviť prípadné váhy jednotlivých susedov, aby výsledok predikcie bol čo najlepší. Implementácia tohto prístupu v systéme KDD Package [3] využíva na optimalizáciu váh evolučný algoritmus. Postup je v tomto prípade zhruba nasledovný.

Pre vzdialenosť dvoch príkladov *A* a *B* sa používa nasledovný vzťah: $D_{A,B} = \sum_{i=1}^{n} W_{A_i} \left| atr_i^A - atr_i^B \right|$, kde *A* je trénovací príklad, *B* je testovací (neznámy)



Obr. 4.6 Výpočet vzdialenosti dvoch príkladov *A* a *B* pre rôzne typy atribútov.

príklad, *n* je počet atribútov, W_{A_i} je váha atribútu *i*, atr_i^A je hodnota atribútu *i*-teho trénovacieho príkladu a atr_i^B je hodnota atribútu *i*-teho testovacieho príkladu. Tento vzťah predpokladá, že všetky atribúty sú numerické a normalizované. Ale v spomínanom systéme si príslušný modul vie poradiť aj s nenormovanými a dokonca aj nenumerickými atribútmi (viď. Obr. 4.6).

Hodnota cieľového (predikovaného) atribútu *Y* nového príkladu sa potom počíta ako vážený priemer $Y = \sum_{i=1}^{k} W_{N_i}.Corr_i.Y_i$, kde W_{N_i} je váha *i*-teho suseda, Y_i je hodnota cieľového atribútu *i*-teho suseda, $Corr_i$ je hodnota korekčného koeficienta *i*-teho suseda a *k* je počet susedov.

Cieľom zavedenia korekcie hodnoty cieľového atribútu pre *i*-teho suseda je vysporiadať sa s prípadným trendom predikovaného atribútu, ktorý sa môže progresívne meniť a obyčajný vážený priemer by tak neumožnil pružne reagovať na takéto zmeny. Hodnoty korekčných koeficientov sú určované nasledovne: $Corr_i = 1 + \sum_{j=1}^{n} (1 - \frac{atr_j^A}{atr_j^B}).W_{C_j}$, kde W_{C_i} je váha *i*-teho atribútu pri korekcii. Pre určenie presnosti generovaného prediktora je potrebné určiť chybu predikcie, akej sa dopúšťa na testovacej množine. KDD Package umožňuje vybrať si z niekoľkých najznámejších typov chýb, ako je stredná absolútna chyba (MAD), stredná kvadratická odchýlka (MSE) alebo relatívna chyba (E).

Na výpočet váh W_{A_i} , W_{N_i} a W_{C_i} pri najmenšej chybe predikcie (zvoleného typu) je v spomínanom systéme do metódy zabudovaný evolučný algoritmus, ktorý slúži na optimalizáciu váh pri používateľom zadanom počte najbližších susedov.

Evolučný algoritmus prehľadáva priestor možných riešení a hľadá čo najlepšie riešenie vzhľadom na zvolenú kriteriálnu funkciu (v tomto prípade ide o minimalizáciu zvoleného typu chyby predikcie). Tento priestor riešení je viacrozmerný a jeho rozmermi sú práve hľadané váhy. Algoritmus pracuje naraz s viacerými potencionálnymi riešeniami. Tieto riešenia sa nazývajú *jedince*. Súbor týchto jedincov tvorí *populáciu*. Z tejto populácie sa snaží evolučný algoritmus použitím tzv. *genetických operátorov* vytvoriť nové jedince. Jedince, na ktoré sú aplikované genetické operátory sa nazývajú *rodičia*. Z už existujúcej

populácie a novovytvorených jedincov – *potomkov* je na základe určitých výberových kritérií vytvorená nová populácia. Jeden takýto cyklus vzniku novej populácie sa nazýva *generácia*. Algoritmus končí prehľadávanie priestoru riešenia po dosiahnutí požadovanej presnosti predikcie alebo po dosiahnutí zadaného počtu generácií. Prehľadávanie samozrejme nie je náhodné ale je riadené určitou stratégiou, nazývanou tiež *selekčným tlakom.*

Keďže evolučný algoritmus má nájsť najlepšie hodnoty váh, má každý jedinec v populácii taký tvar, ako to zobrazuje Tabuľka 4.7.

W_{a1}	...	W_{an}	W_{N1}	...	W_{Nnn}	W_{c1}	...	W_{cn}

Tabuľka 4.7 Reprezentácia jedinca v použitom evolučnom algoritme.

Ide o jednorozmerné pole reálnych čísel. Každé číslo reprezentuje jednu váhu. Každému jedincovi je priradená tzv. *vhodnosť (*angl. *fitness)*, ktorá určuje kvalitu daného jedinca. Keďže cieľom je nájsť také váhy, aby predikcia bola čo najpresnejšia, je táto vhodnosť jedincov nepriamo závislá na chybe predikcie. To vlastne znamená, že jedinec je najvhodnejší vtedy, keď je chyba predikčnej metódy pri použití váh, ktoré jedinec obsahuje, najmenšia. Ďalší postup je uvedený v podobe algoritmu.

```
EvolučneOptimalizujVáhy (Populácia P, Integer generácia, Integer
                         MaxGenerácií)
     if P je prázdne then P = VytvorPočiatočnúPopuláciu();
     if generácia = MaxGenerácií then
          return P;
     if SpĺňaPožadovanúPresnosť(P) then
          return P;
     NP = VytvorNovýchPotomkov(P);
     NováPopulácia = VýberNovejPopulácie(P, NP);
     return EvolučneOptimalizujVáhy(NováPopulácia);
```

4.3 Použitá literatúra

[1] Ester, M., Sander, J.: Knowledge Discovery in Databases – Techniken und Anwendungen. Springer Verlag, (2000)

[2] Han, J., Kamber, M.: Data Mining – Concepts and Techniques. Morgan Kaufmann Publishers, (2000)

[3] KDD Package with documentation: http://neuron.tuke.sk/~paralic/KDD/

[4] Klösgen, W. and Zytkow, J.M.: The Knowledge Discovery Process. In Handbook of Data Mining and Knowledge Discovery. Oxford University Press, (2002)

[5] Mach, M.: Získavanie znalostí pre znalostné systémy. Elfa, Košice, 104 s. ISBN 80-88786-67-3 (1997)

[6] Machová, K.: Strojové učenie – Princípy a algoritmy. Elfa, Košice, 117 s. ISBN 80 89066-51-8 (2002)

[7] Quinlan, J. R.: C4.5 Programs for machine learning. Morgan Kaufmann Publishers, San Mateo, (1993)

[8] Quinlan J.R.: *Learning with continuous classes,* Proceedings 5th Australian Joint Conference on Artificial Intelligence, Singapure, pp. 343-348, (1992)

[9] Wang Y., Witten I. H.: *Inducing Model Trees for Continuous Classes*, University of Waikato, New Zealand

[10] Witten, I.W., Frank, E.: Data Mining – Practical Machine Learning Tools and Techniques with JAVA Implementations. Morgan Kaufman, San Francisco, (2000)

[11] Wrobel, S., Morik, K., Joachims, T.: Maschinelles Lernen und Data Mining. Handbuch der Künstlichen Intelligenz, 3. vydanie, Oldenbourg Verlag München, pp. 517 – 598, (2000)

5 Ďalšie úlohy dolovania v dátach

Táto, posledná kapitola pojednáva o jednej z ďalších často používaných úloh dolovania v dátach, a síce dolovanie tzv. asociačných pravidiel.

5.1 Asociačné pravidlá

Asociačné pravidlá ([1], [2]) sú okrem iného nástrojom na tzv. analýzu nákupného košíka. Daná je pritom databáza určitých zákazníckych transakcií, t.j. v najjednoduchšom prípade ide naozaj o množinu spolu nakúpených artiklov jednotlivých zákazníkov obchodu, ale vo všeobecnosti to môžu byť ľubovoľné služby, alebo informácie nejakého predajcu.

Asociácie potom vyjadrujú také súvislosti v rámci transakcií, ktoré sa v danej databáze často vyskytujú. V prípade supermarketu môže teda ísť o súvislosti medzi často spolu nakupovanými tovarmi. Takéto pravidlá môžu byť vyjadrené nasledovne: „múka, vajíčka $\Rightarrow$ maslo“. Toto pravidlo platí vtedy, ak v nákupných košíkoch, kde sa nachádza múka a vajíčka, sa často okrem toho nachádza aj maslo. Takéto pravidlo je samozrejme zaujímavé len vtedy, ak platí pre dostatočné množstvo nákupných košíkov.

Znalosti vo forme asociačných pravidiel je možné v praxi využiť mnohými spôsobmi napr. efektívny krížový marketing, cielenú reklamu, zlepšený návrh katalógov a štruktúry obchodu, ale aj na segmentáciu zákazníkov s rovnakým správaním sa pri nakupovaní určenom asociačnými pravidlami.

5.1.1 Jednoduché asociačné pravidlá – Algoritmus Apriori

Nech $I = \{i_1, ..., i_m\}$ je množina literálov, nazývaných *položky*. $X \subseteq I$ je tzv. *množina položiek*, alebo *položková množina* (napr. tovarov v supermarkete).

D je množina transakcií T, pričom každá transakcia T vlastne predstavuje množinu položiek, teda $T \subseteq I$. Ak pre určitú množinu položiek X platí, že $X \subseteq T$, potom T *obsahuje* X. T typicky reprezentuje jeden nákup (obsah nákupného košíka) a D potom celú databázu, t.j. všetky nákupy za určité časové obdobie.

Jednotlivé položky v transakciách majú byť lexikograficky usporiadané. Teda položkovú množinu X pozostávajúcu z položiek $x_1, ..., x_k$ možno potom zapísať ako $X = (x_1, ..., x_k)$, pričom $x_1 \leq ... \leq x_k$. Počet položiek v položkovej množine vyjadruje jej *dĺžku* a množina položiek dĺžky k sa nazýva *k-položková množina*.

Pre množinu položiek $X \subseteq I$ je *podpora množiny X v D* definovaná ako podiel tých transakcií v D, ktoré obsahujú X. Ide teda o relatívnu početnosť tých transakcií, v ktorých boli spoločne nakúpené položky z X.

Asociačné pravidlo je implikácia vo forme $X \Rightarrow Y$, pričom X a Y sú dve disjunktné položkové množiny, t.j. $X \subseteq I$, $Y \subseteq I$ a $X \cap Y = \emptyset$. *Podpora* asociačného pravidla $X \Rightarrow Y$ v D (označovaná s) je vlastne podpora zjednotenia $X \cup Y$, t.j. relatívna početnosť spoločného výskytu všetkých položiek, ktoré sa vyskytujú v asociačnom pravidle.

Spoľahlivosť c asociačného pravidla $X \Rightarrow Y$ v *D* je definovaná ako podiel transakcií obsahujúcich množinu položiek *Y* v rámci množiny tých transakcií z *D*, ktoré obsahujú množinu položiek *X*. T.j. asociačné pravidlo $X \Rightarrow Y$ platí so spoľahlivosťou *c* v *D*, ak *c*% všetkých takých transakcií z *D*, ktoré obsahujú *X*, obsahujú súčasne aj *Y*. Ešte ináč to možno vyjadriť tak, že spoľahlivosť *c* asociačného pravidla $X \Rightarrow Y$ v *D* je podmienená pravdepodobnosť toho, že transakcia *T* obsahuje všetky položky množiny *Y* za podmienky, že obsahuje aj všetky položky množiny *X*.

Príklad 5.1

Význam podpory *s* a spoľahlivosti *c* asociačného pravidla je ukázaný na príklade z nasledovnej databázy *D* (Tabuľka 5.1).

Transakcia	Nakúpené položky
1	chlieb, káva, mlieko, koláč
2	káva, mlieko, koláč
3	chlieb, maslo, káva, mlieko
4	mlieko, koláč
5	chlieb, koláč
6	Chlieb

Tabuľka 5.1 Daná transakčná databáza *D*.

Množina položiek X_1 = {káva, mlieko} a X_2 = {káva, koláč, mlieko}. Podpora s_1 množiny položiek X_1 je 3 transakcie z celkového počtu 6 transakcií v *D*, t.j. 50%. Podpora s_2 množiny položiek X_2 je 2 transakcie z celkového počtu 6 transakcií v *D*, t.j. 33%.

Asociačné pravidlo „káva, mlieko $\Rightarrow$ koláč“ má teda podporu $s = s_2$ zhodnú s podporou množiny položiek X_2. t.j. 33%. Spoľahlivosť tohto pravidla je podiel 2 transakcií (obsahujúcich X_2) ku 3 (obsahujúcim X_1), t.j. 67%.

Úlohou algoritmu Apriori je vzhľadom na danú množinu transakcií *D* nájsť všetky také asociačné pravidlá, ktorých podpora je väčšia ako zadaná minimálna podpora *min_s* a spoľahlivosť väčšia ako minimálna zadaná spoľahlivosť *min_c*. Táto úloha sa dá rozdeliť na dve podúlohy.

- Nájdenie všetkých takých množín položiek, ktorých podpora je väčšia ako *min_s*. Takéto množiny položiek sa nazývajú *frekventované*. Naivný algoritmus by za týmto účelom musel preskúmať všetkých 2^m možných podmnožín *I*, pričom *m* je obvykle veľké číslo. Apriori algoritmus ale využíva určitú vlastnosť monotónnosti (viď nižšie), čo mu umožňuje výrazne obmedziť množstvo alternatívnych podmnožín, ktoré je nutné preskúmať.
- Použitie nájdených frekventovaných množín položiek na vytvorenie asociačných pravidiel spĺňajúcich podmienku minimálnej zadanej spoľahlivosti *min_c*. Ak totiž nejaká množina položiek *X* je frekventovaná (má aspoň minimálnu požadovanú spoľahlivosť *min_s*), potom každá podmnožina A z X vytvára pravidlo v tvare $A \Rightarrow$

(X – A), ktoré spĺňa minimálnu podporu *min_s*. Preto už je potrebné len zistiť, či má aj potrebnú spoľahlivosť (aspoň *min_c*).

Apriori algoritmus využíva pre nájdenie frekventovaných podmnožín nasledovnú vlastnosť monotónnosti, ktorú majú frekventované množiny – každá podmnožina frekventovanej množiny položiek musí byť tiež frekventovaná.

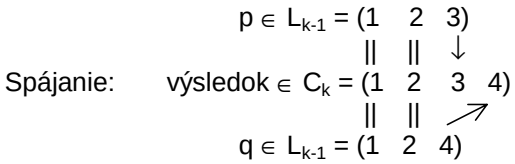
Aby bolo možné využiť túto vlastnosť, musia byť frekventované množiny určované podľa veľkosti, t.j. najprv jednoprvkové frekventované množiny (na ich určenie stačí jednoduché spočítanie výskytov jednotlivých položiek v transakčnej databáze). V ďalšom kroku už potom nie je nutné skúmať všetky dvojprvkové množiny, ale iba tie, ktoré je možné vytvoriť kombináciou jednoprvkových frekventovaných množín. Vo všeobecnosti kandidáti na frekventované (*k*+1)-prvkové množiny sú len tí, ktorých je možné získať „spojením“ frekventovaných *k*-položkových množín. Algoritmus prechádza viackrát databázou až dovtedy, kým pri určitej veľkosti *k* už nemožno nájsť žiadne ďalšie frekventované množiny.

```
Apriori (Položky I, Databáza D, Float min_s)
     L1 := {frekventované jednopoložkové množiny z I};
     k := 2;
     while Lk-1 ≠ ∅ do
          Ck := GenerovanieKandidátovApriori(Lk-1);
          for each Transakciu T ∈ D do
               CT := Podmnožina(Ck, T); // všetci kandidáti
               // z Ck, ktorí sú obsiahnutí v transakcii T
          for each Kandidát c ∈ CT do c.count++;
          Lk := {c ∈ Ck | c.count / |D| ≥ min_s};
          k++;
     return ⋃k Lk ;

GenerovanieKandidátovApriori (Lk-1)
     insert into Ck // 1. spájanie
     select p.item1, p.item2, ... p.itemk-1, q.itemk-1
     from Lk-1 p, Lk-1 q
     where (p.item1 = q.item1), ..., (p.itemk-2 = q.itemk-2),
           (p.itemk-1 < q.itemk-1);
     for each položková_množina c ∈ Ck do
          for each (k-1)-položková podmnožina s z c do
               if s ∉ Lk-1 then vymaž c z Ck; // 2. orezávanie
```

Generovanie kandidátov na frekventované množiny veľkosti *k* tvorí jadro Apriori algoritmu. Množina vygenerovaných kandidátov musí obsahovať všetky frekventované množiny veľkosti *k*, ale zároveň by mala byť podstatne menšia ako množina všetkých *k*-prvkových podmnožín *I*.

Kandidáti veľkosti *k*+1 sú generovaní v dvoch krokoch, pričom sa predpokladá, že položky sú v rámci položkových množín usporiadané lexikograficky. V prvom kroku (spájanie) sa spájajú medzi sebou frekventované množiny veľkosti *k*. Pritom sa *k*-prvková frekventovaná množina *p* predĺži o *k*-ty prvok takej *k*-prvkovej frekventovanej množiny *q*, ktorá sa s množinou *p* zhoduje v prvých *k*-1 prvkoch. Tento postup je znázornený na Obr. 5.1.



Obr. 5.1 Generovanie kandidátov veľkosti *k* algoritmom Apriori

V druhom kroku (orezávanie) sa potom z množiny kandidátov veľkosti *k*, t.j. C_k vygenerovanej v prvom kroku spájania odstránia tie množiny, ktoré obsahujú takú podmnožinu veľkosti *k*-1, ktorá nie je frekventovaná (t.j. nenachádza sa v L_{k-1}).

Nech je daná množina trojprvkových frekventovaných množín L_3 = {(1 2 3), (1 2 4), (1 3 4), (1 3 5), (2 3 4)}. Aplikovaním prvého kroku spájania z nej možno získať množiny kandidátov veľkosti 4, a síce C_4 = {(1 2 3 4), (1 3 4 5)}. Z tejto množiny však potom druhý krok orezávania odstráni množinu (1 3 4 5), nakoľko jej podmnožina (1 4 5) sa nenachádza v L_3, a teda nie je frekventovaná. Množina štvorprvkových frekventovaných množín bude potom L_4 = {(1 2 3 4)}.

Príklad 5.2

Daná je transakčná databáza D (Tabuľka 5.2). Úlohou je nájsť všetky frekventované množiny položiek za predpokladu, že hranica minimálnej podpory je stanovená na 40% (t.j. aspoň výskyt v aspoň dvoch transakciách z celkového počtu 4 transakcie).

ID transakcie	Položky
100	A C D
200	B C E
300	A B C E
400	B E

Tabuľka 5.2 Daná transakčná databáza *D*

Najskôr sa prechodom databázy spočítajú výskyty jednotlivých položiek s cieľom určiť frekventované jednoprvkové množiny. Počty výskytov jednotlivých položiek uvádza nasledovná Tabuľka 5.3.

Položková množina	Podpora
(A)	2
(B)	3
(C)	3
(D)	1
(E)	3

Tabuľka 5.3 Podpora jednotlivých položiek v *D* (t.j. C_1, kandidáti veľkosti 1)

Z kandidátov frekventovaných jednoprvkových množín nespĺňa stanovený prah podpory množina (D), ostatné ju spĺňajú, takže množina

frekventovaných jednoprvkových množín L_1 = {(A), (B), (C), (E)}. Z tejto množiny je možné krokom spájania vygenerovať nasledovných kandidátov veľkosti 2 – C_2 = {(A B), (A C), (A E), (B C), (B E), (C E)}. Prechodom databázy možno získať hodnoty podpory jednotlivých kandidátov veľkosti 2 (Tabuľka 5.4Tabuľka).

Položková množina	Podpora
(A B)	1
(A C)	2
(A E)	1
(B C)	2
(B E)	3
(C E)	2

Tabuľka 5.4 Podpora kandidátov na dvojprvkové frekventované množiny v *D*.

Minimálnu stanovenú hodnotu podpory spĺňajú štyri z uvedených dvojprvkových množín, a síce L_2 = {(A C), (B C), (B E), (C E)}. Z uvedenej množiny dvojprvkových frekventovaných množín položiek možno spájaním vygenerovať jedného trojprvkového kandidáta C_3 = {(B C E)}. Keďže podpora tejto množiny je 2, čo vyhovuje podmienke minimálnej podpory, je táto množina skutočne aj frekventovanou, t.j. L_3 = {(B C E)}. Z tejto množiny trojprvkových frekventovaných množín už ale nie je možné vygenerovať žiadneho štvorprvkového kadidáta, a tak algoritmus Apriori končí. Všetky frekventované množiny v *D* boli nájdené (ide o množiny z L_1, L_2 a L_3).

Po nájdení všetkých frekventovaných množín položiek $L = \bigcup_k L_k$ je ešte potrebné určiť na základe nich asociačné pravidlá spĺňajúce podmienku minimálnej spoľahlivosti. Pre každé asociačné pravidlo $A \Rightarrow B$ musí zodpovedajúca množina položiek ($A \cup B$) spĺňať podmienku minimálnej podpory, a musí sa preto nachádzať medzi nájdenými frekventovanými množinami. To znamená, že teraz stačí pre každú podmnožinu A frekventovanej množiny X preskúmať spoľahlivosť potenciálneho asociačného pravidla $A \Rightarrow (X - A)$. K tomu stačí poznať podporu všetkých frekventovaných množín, lebo

$$spoľahlivosť(A \Rightarrow (X - A)) = \frac{podpora(X)}{podpora(A)}.$$

Na zisťovanie asociačných pravidiel teda už nie je potrebná žiadna dodatočná informácia. Podpora jednotlivých frekventovaných množín môže byť uložená už pri ich generovaní (napr. pomocou špeciálneho hashovacieho stromu pre efektívny prístup [1]).

Ale aj generovanie všetkých možných pravidiel typu $A \Rightarrow (X - A)$ je zbytočne neefektívne. Ak napr. takéto pravidlo nespĺňa podmienku minimálnej spoľahlivosti, potom žiadne pravidlo typu $A' \Rightarrow (X - A')$, kde $A' \subseteq A$ nemôže spĺňať podmienku minimálnej spoľahlivosti. Dôvod spočíva v tom, že podpora množiny A' môže byť nanajvýš rovná, alebo väčšia ako podpora A, teda spoľahlivosť pravidla $A' \Rightarrow (X - A')$ môže byť nanajvýš rovná alebo menšia ako spoľahlivosť pravidla $A \Rightarrow (X - A)$. Ak napr. pravidlo $abc \Rightarrow d$ nespĺňa

podmienku minimálnej spoľahlivosti, potom ju nemôžu spĺňať ani pravidlá $ab \Rightarrow cd$, $ac \Rightarrow bd$, $bc \Rightarrow ad$, $a \Rightarrow bcd$, $b \Rightarrow acd$, $c \Rightarrow abd$ a nemusia sa teda vôbec uvažovať. Túto vlastnosť využíva aj nasledovný algoritmus:

```
GenerujAsociačnéPravidlá (Frekventované_množiny_položiek L)
for each frekventovanú množinu položiek l_k ∈ L, k ≥ 2 do
          GenerujĎalšiePravidlá (l_k, l_k);

GenerujĎalšiePravidlá (l_k, a_m)
     A := {(m-1)-položkové množiny a_m-1 | a_m-1 ⊂ a_m};
for each a_m-1 ∈ A do
          konf := podpora(l_k) / podpora(a_m-1);
          if konf ≥ min_c then
               return Pravidlo l_k ⇒ (l_k - a_m-1) so
spoľahlivosťou konf a podporou podpora(l_k);
               if m-1 > 1 then
                    GenerujĎalšiePravidlá (l_k, a_m-1);
```

5.1.2 Zaujímavosť jednoduchých asociačných pravidiel

Pri posudzovaní zaujímavosti nájdených asociačných pravidiel je potrebné byť opatrný a všímať si aj nasledovné súvislosti.

Príklad 5.3

Predajca čokoládových tyčiniek sleduje správanie sa detí ráno v škole s 5000 deťmi. Z dát vyplýva, že 60% detí hrá futbal (t.j. 3000 detí), 75% detí je čokoládové tyčinky (t.j. 3750 detí) a 40% detí robí oboje, teda aj hrajú futbal aj jedia čokoládové tyčinky (t.j. 2000 detí).

Po aplikovaní algoritmu Apriori so zadanou hranicou minimálnej podpory 40% a minimálnej spoľahlivosti 60% na uvedené dáta sa získa aj nasledovné asociačné pravidlo: „hrá futbal $\Rightarrow$ je čokoládové tyčinky“. Podpora takéhoto pravidla je totiž 40% (2000 detí) a jeho spoľahlivosť 2000/3000 = 67%.

Ale spomenuté asociačné pravidlo je v skutočnosti zavádzajúce, nakoľko medzi všetkými deťmi je percento takých, čo jedia čokoládové tyčinky vyššie (75%) ako u tých, čo hrajú futbal (67%). Takže hranie futbalu a jedenie čokoládových tyčiniek je v danej databáze záporne korelované, hranie futbalu totiž znižuje pravdepodobnosť jedenia čokoládových tyčiniek.

Aby sa zabránilo objavovaniu takýchto zavádzajúcich pravidiel, musia byť po ukončení algoritmu Apriori ešte získané pravidlá prefiltrované. Pre každé pravidlo $A \Rightarrow B$ možno napríklad ešte testovať, či platí aj $\frac{P(A \cap B)}{P(A)} - P(B) > d$ pre danú konštantu $d > 0$. Intuitívne to znamená, že spoľahlivosť pravidla $A \Rightarrow B$ (t.j. podmienená pravdepodobnosť B za podmienky A) musí byť väčšia ako pravdepodobnosť samotného dôsledku B pravidla.

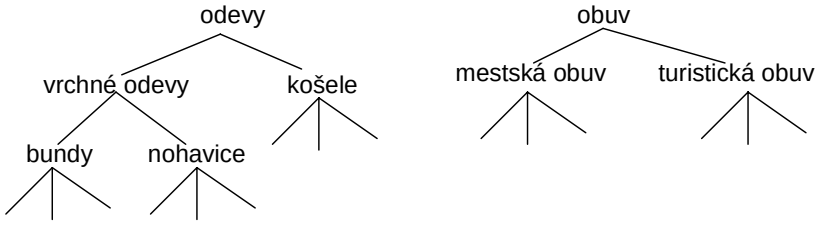
Hodnotu výrazu $\frac{P(A \cap B)}{P(A)} - P(B)$ možno zároveň považovať za mieru zaujímavosti pravidla, t.j. čím je táto hodnota pre určité pravidlo $A \Rightarrow B$ väčšia, tým zaujímavejší vzťah medzi A a B toto pravidlo vyjadruje.

5.1.3 Hierarchické asociačné pravidlá vzhľadom na taxonómiu položiek

V praktických aplikáciách u jednoduchých asociačných pravidiel položky v transakciách zodpovedajú obvykle jednotlivým tovarom na úrovni ich čiarového kódu. Vo všeobecnosti asociačné pravidlá s viacerými položkami na úrovni čiarových kódov dosahujú len veľmi nízke hodnoty podpory a nevedú k nájdeniu žiadnych skutočne využiteľných výsledkov. T.j.:

- Pri vysokej hodnote minimálnej podpory nájde algoritmus Apriori len veľmi málo, obvykle len jednoduchých pravidiel.
- Pri nízkej hodnote minimálnej podpory nájde algoritmus Apriori veľmi veľké a neprehľadné množstvo pravidiel.

Často ale v praktických aplikáciách sú k dispozícii aj taxonómie jednotlivých položiek (*is-a* hierarchie, viď. Obr. 5.2), ktoré zoskupujú položky do hierarchických skupín. Tieto taxonómie je možné využiť pre hľadanie asociačných pravidiel na vyššej úrovni všeobecnosti, t.j. medzi celými skupinami položiek. Takéto asociačné pravidlá bývajú nielen zaujímavejšie, ale často zahŕňajú aj väčší počet položiek, lebo položky na všeobecnejšej úrovni majú aj väčšiu podporu.



Obr. 5.2 Hierarchie položiek pre artikle na oblečenie

Asociačné pravidlo môže pozostávať z artiklov na rôznych úrovniach taxonómie. Napr. je možné, že sa nájde pravidlo „vrchné odevy $\Rightarrow$ turistická obuv“ s dostatočnou podporou, ak sa pomerne často spolu nakupujú bundy s turistickými topánkami a lyžiarske nohavice s lyžiarskymi topánkami, aj keď tieto dve asociácie samostatne nedosahujú prah minimálnej podpory.

Nech $I = \{i_1, ..., i_m\}$ je opäť množina literálov, nazývaných *položky*. Okrem identifikátorov jednotlivých tovarov sú ale teraz prípustné aj označenia skupín položiek v I.

Nech H je orientovaný acyklický graf $H = (I, G)$, $G \subseteq I \times I$ nad množinou literálov I, reprezentujúci množinu hierarchických relácií typu *is-a* nad položkami z I. Hrana v H z i do j znamená, že i je zovšeobecnením j. V tomto prípade sa i nazýva otcom alebo *priamym predkom* j a j sa nazýva synom, alebo *priamym potomkom* i. Uzol $\overline{x}$ sa nazýva *predkom* x (a x zase *potomkom* $\overline{x}$) vzhľadom na H, ak existuje v H cesta z $\overline{x}$ do x. Množina položiek $\overline{Z}$ sa nazýva *predkom* množiny položiek Z, ak možno $\overline{Z}$ vytvoriť zo Z takým spôsobom, že sa aspoň jedna položka v Z nahradí jedným zo svojich predkov. Opačne sa potom aj Z nazýva *potomkom* $\overline{Z}$.

D je množina transakcií *T*, pričom $T \subseteq I$. Typicky obsahujú transakcie *T* z databázy *D* len položky z listových uzlov orientovaného grafu *H* (t.j. len konkrétne tovary). Ale v ďalšom uvedený postup nie je viazaný týmto predpokladom. Transakcia *T podporuje položku* $i \in I$, ak je *i* obsiahnutá v *I*, alebo ak *i* je predkom nejakej položky obsiahnutej v *I*. *T podporuje množinu položiek* $X \in I$, ak *T* podporuje každú položku z *X*. Podpora množiny položiek $X \subseteq I$ v *D* je definovaná ako percentuálny podiel tých transakcií v *D*, ktoré podporujú *X*.

Hierarchické asociačné pravidlo je implikácia vo forme $X \Rightarrow Y$, pričom opäť platí že *X* a *Y* sú dve množiny položiek, ktoré nemajú spoločný prvok, t.j. $X \subseteq I$, $Y \subseteq I$ a $X \cap Y = \emptyset$. Naviac musí ale pre hierarchické asociačné pravidlo platiť aj to, že žiadna položka z *Y* nie je predkom položky z *X* vzhľadom na *H*. Dôvodom tejto podmienky je skutočnosť, že pravidlo typu „$x \Rightarrow$ predok(x)“ triviálne platí so 100% spoľahlivosťou a je teda redundantné (definuje vlastne vzťah hierarchie v taxonómii).

Podpora s hierarchického asociačného pravidla $X \Rightarrow Y$ v *D* je opäť definovaná ako podpora množiny $X \cup Y$ v *D*. *Spoľahlivosť c hierarchického asociačného pravidla* $X \Rightarrow Y$ v *D* je definovaná ako percentuálny podiel transakcií ktoré podporujú množinu *Y* v podmnožine tých transakcií, ktoré podporujú aj *X*.

Príklad 5.4

Význam podpory *s* a spoľahlivosti *c* hierarchického asociačného pravidla je ukázaný na príklade z nasledovnej databázy *D* (Tabuľka 5.5Tabuľka). Daná je aj taxonómia položiek podľa Obr. 5.2.

Transakcia	Nakúpené položky
1	košeľa
2	bunda, turistické topánky
3	lyžiarske nohavice, turistické topánky
4	mestská obuv
5	mestská obuv
6	bunda

Tabuľka 5.5 Daná transakčná databáza *D*.

Podpora skupiny položiek {odevy} je daná 4 transakciami (poradové čísla transakcií 1, 2, 3 a 6) z celkového počtu 6 transakcií v *D*, t.j. 67%. Podpora skupiny položiek {obuv} je tiež daná 4 transakciami (poradové čísla transakcií 2, 3, 4, 5) z celkového počtu 6 transakcií v *D*, t.j. 67%.

Podpora množiny skupín položiek {odevy, obuv} je rovnaká ako podpora množiny skupín položiek {odevy, turistické topánky}, t.j. 2 transakcie (s poradovými číslami 2 a 3) z celkového počtu 6 transakcií v *D*, t.j. 33%.

Asociačné pravidlo „obuv $\Rightarrow$ odev“ má teda podporu 2 transakcie zo 6 v *D*, t.j. 33% a spoľahlivosť tohto pravidla je podiel 2 transakcie (obsahujúce obuv aj odevy) zo 4 (obsahujúce obuv), t.j. 50%.

Asociačné pravidlo „turistická obuv $\Rightarrow$ odev“ má síce rovnakú podporu ako predchádzajúce pravidlo, t.j. 2 transakcie zo 6 v *D* (33%), ale jeho

spoľahlivosť je až 100%, čo je podiel 2 transakcií (obsahujúcich turistickú obuv aj odevy súčasne) z 2 (obsahujúcich len turistickú obuv).

Objavovanie hierarchických asociačných pravidiel prebieha v troch krokoch. Prvé dva sú podobné Apriori algoritmu, tretí filtruje nájdené pravidlá s ohľadom na ich zaujímavosť.

1. Nájdenie všetkých takých množín položiek, ktorých podpora je vyššia ako stanovená hranica *min_s*.
2. Pomocou nájdených frekventovaných množín položiek nájsť všetky také asociačné pravidlá, ktorých spoľahlivosť je vyššia ako stanovená hranica *min_c*,
3. Odstránenie tých pravidiel nájdených v predchádzajúcom kroku, ktorých zaujímavosť je menšia ako používateľom stanovená minimálna zaujímavosť.

V treťom kroku ide o prípady podobné tomu z predchádzajúceho príkladu (Príklad 5.4). Ak bolo totiž objavené pravidlo „obuv ⇒ odev“ (podpora 33%, spoľahlivosť 50%), potom na základe danej taxonómie, kde skupina položiek obuv pozostáva z dvoch podskupín (mestská obuv a turistická obuv), je možné predpokladať, že podpora pravidla „turistická obuv ⇒ odev“ bude zhruba polovičná (za predpokladu rovnomerného rozdelenia predajov rôznych typov obuvi) v porovnaní s všeobecnejším pravidlom „obuv ⇒ odev“, t.j. okolo 16,5%. Ak podpora takéhoto pravidla skutočne korešponduje s očakávanou hodnotou podpory, možno toto pravidlo považovať za redundantné. Ak ale podpora špecifickejšieho pravidla je väčšia ako predpokladaná (to je aj prípad pravidla „turistická obuv ⇒ odev“ v uvedenom príklade), potom možno takéto pravidlo považovať za tým zaujímavejšie, o čo vyššia je jeho podpora oproti očakávanej.

```
AprioriH-základný (Položky I, Databáza D, Float min_s)
     L1 := {frekventované jednopoložkové množiny z I};
     k := 2;
     while Lk-1 ≠ ∅ do
          Ck := GenerovanieKandidátovApriori(Lk-1);
          for each Transakciu T ∈ D do
               T' := T ∪ {ī | i ∈ T a ī je predkom I v H};
               CT := Podmnožina(Ck, T'); // všetci kandidáti
                    z Ck, obsiahnutí v rozšírenej transakcii T'
               for each Kandidát c ∈ CT do c.count++;
          Lk := {c ∈ Ck | c.count / |D| ≥ min_s};
          k++;
     return ⋃k Lk ;
```

Uvedený základný hierarchický algoritmus Apriori sa líši od pôvodného iba v jedinom (siedmom) riadku, kde sa najskôr každá transakcia z *T* nahradí novou transakciou *T'*, v ktorej sú zahrnutí aj všetci predkovia jednotlivých položiek z *T*. Potom je možné už použiť ten istý prístup ako pre jednoduché asociačné pravidlá. Uvedený základný algoritmus je možné samozrejme vo viacerých smeroch vylepšiť. Podrobnosti možno nájsť napr. v [1].

Užitočnosť asociačných pravidiel sa však neobmedzuje len na transakčné databázy, ale môže priniesť zaujímavé znalosti aj pri použití

v relačných databázach. Vtedy sa hovorí o tzv. kvantitatívnych asociačných pravidlách.

5.1.4 Kvantitatívne asociačné pravidlá

Transakčnú databázu (napr. Tabuľka 5.6Tabuľka) je možné previesť na jednoduchú boolovskú databázu (Tabuľka 5.7), v ktorej jednotlivé atribúty vlastne zodpovedajú položkám vyskytujúcim sa v transakčnej databáze a majú dve možné hodnoty: 1 ak sa daná položka v transakcii vyskytla a 0 ak nie (viď. nasledujúci príklad).

Transakcia	Nakúpené položky
1	chlieb, káva, mlieko, koláč
2	káva, mlieko, koláč
3	chlieb

Tabuľka 5.6 Transakčná databáza D.

Transakcia	chlieb	maslo	káva	mlieko	koláč
1	1	0	1	1	1
2	0	0	1	1	1
3	1	0	0	0	0

Tabuľka 5.7 Boolovská reprezentácia transakčnej databázy.

V mnohých oblastiach sú ale v databázach obsiahnuté zaujímavé dáta zložitejších typov (najmä numerické a kategorické). Cieľom je prispôsobiť algoritmus ktorý funguje pre hľadanie asociačných pravidiel v boolovských databázach (teda transakčné dáta) na zložitejšie dátové typy (tzv. *kvantitatívne asociačné pravidlá*).

Hlavná myšlienka pritom spočíva v tom, že sa databázy s numerickými a kategorickými atribútmi transformujú tak, aby bolo možné použiť podobný postup ako pre boolovské databázy. Pritom sa najprv numerické a kategorické atribúty transformujú na boolovské. Kategorický alebo numerický atribút A s niekoľko málo diskrétnymi hodnotami w_1, ..., w_k možno transformovať na k boolovských atribútov A_1, ..., A_k, pričom každej hodnote w_i pôvodného atribútu A zodpovedá jeden nový boolovský atribút A_i, ktorý nadobúda hodnotu 1 práve vtedy, ak hodnota atribútu A v pôvodnej databáze bola w_i, ináč má hodnotu 0.

ID záznamu	vek	stav	počet áut
1	23	slobodný	0
2	38	ženatý	2

Tabuľka 5.8 Príklad databázy s dvoma numerickými a jedným kategorickým atribútom

Numerické atribúty s veľkým rozsahom hodnôt sa najprv diskretizujú a ďalší postup transformácie je už potom rovnaký ako pre kategorické atribúty (viď. Tabuľka 5.9, kde je uvedená boolovská databáza, ktorá vznikla transformáciou vyššie uvedenej databázy s numerickými atribútmi – Tabuľka 5.8).

ID záznamu	vek: <20..29>	vek: <30..39>	stav: slobodný	stav: ženatý	počet áut: 0	počet áut: 1	počet áut: 2
1	1	0	1	0	1	0	0
2	0	1	0	1	0	0	1

Tabuľka 5.9 Transformácia na boolovskú databázu

Cieľom je opäť nájsť nad danou množinou *D* záznamov v databáze všetky zaujímavé, kvantitatívne asociačné pravidlá s aspoň minimálnou stanovenou podporou a spoľahlivosťou. Postup je nasledovný.

1. Pre každý numerický atribút určiť rozdelenie rozsahu hodnôt na vhodné intervaly (diskretizácia) a prevedenie týchto intervalov na množinu po sebe idúcich celých čísel tak, aby bolo zachované pôvodné usporiadanie intervalov.
2. Zobrazenie hodnôt kategorických atribútov na po sebe nasledujúce celé čísla.
3. Transformácia záznamov pôvodnej databázy na novú množinu záznamov pomocou stanovených zobrazení atribútov.
4. Určenie podpory najprv pre každý pár atribút-hodnota. Susediace hodnoty pôvodných numerických atribútov sa potom zlučujú do spoločných intervalov dovtedy, kým podpora novovzniknutých intervalov neprekročí používateľom zadanú minimálnu hodnotu podpory *min_s*. Tieto sa potom stávajú frekventovanými jednoprvkovými množinami (*frekventované kvantitatívne položky*). Nasleduje nájdenie všetkých frekventovaných kvantitatívnych množín položiek variantov Apriori algoritmu.
5. Určenie kvantitatívnych asociačných pravidiel na základe množiny frekventovaných kvantitatívnych množín položiek (analogicky ako pri jednoduchých asociačných pravidlách).
6. Odstránenie tých pravidiel, ktorých zaujímavosť je menšia ako používateľom zadaná minimálna hodnota.

Príklad 5.5

Daná je databáza D (Tabuľka 5.10).

ID záznamu	vek	stav	počet áut
100	23	slobodný	1
200	25	ženatý	1
300	29	slobodný	0
400	34	ženatý	2
500	38	ženatý	2

Tabuľka 5.10 Daná databáza D.

Kategorický atribút stav má len dve hodnoty, takže je možné ho transformovať na celočíselný atribút s hodnotami 1 (pre ženatý) a 2 (pre slobodný). Rozsah hodnôt numerického atribútu vek možno rozdeliť na 4 disjunktné intervaly a priradiť im po sebe idúce celé čísla zachovávajúc poradie nasledovne: intervalu hodnôt <20..24> bude zodpovedať hodnota 1

transformovaného atribútu, intervalu <25..29> hodnota 2, intervalu <30..34> hodnota 3 a konečne intervalu <35..39> hodnota 4. Numerický atribút počet áut nadobúda len 3 rôzne hodnoty, ktoré už sú po sebe idúce celé čísla, takže nie je nutné ho transformovať. Po vykonaní spomenutých transformácií databáza vyzerá nasledovne (Tabuľka 5.11Tabuľka).

ID záznamu	vek	stav	počet áut
100	1	2	1
200	2	1	1
300	2	2	0
400	3	1	2
500	4	1	2

Tabuľka 5.11 Transformovaná databáza

Nasledujú príklady frekventovaných kvantitatívnych množín položiek (Tabuľka 5.12) a kvantitatívnych asociačných pravidiel (Tabuľka 5.13).

Množina položiek	Podpora
(<vek:20..29>)	3
(<vek:30..39>)	2
(<stav:ženatý>)	3
(<stav:slobodný>)	2
(<autá:0..1>)	3
(<vek:30..39><stav:ženatý>)	2

Tabuľka 5.12 Príklady frekventovaných kvantitatívnych množín položiek

Kvantitatívne asociačné pravidlo	Podpora	Spoľahlivosť
<vek:20..29> $\Rightarrow$ <autá:0..1>	40%	100%
<vek:30..39> a <stav:ženatý> $\Rightarrow$ <autá:2>	60%	67%

Tabuľka 5.13 Príklady kvantitatívnych asociačných pravidiel

5.2 Použitá literatúra

[1] Ester, M., Sander, J.: Knowledge Discovery in Databases – Techniken und Anwendungen. Springer Verlag, (2000)

[2] Han, J., Kamber, M.: Data Mining – Concepts and Techniques. Morgan Kaufmann Publishers, (2000)

[3] Klösgen, W. and Zytkow, J.M.: The Knowledge Discovery Process. In Handbook of Data Mining and Knowledge Discovery. Oxford University Press, (2002)

[4] Wrobel, S., Morik, K., Joachims, T.: Maschinelles Lernen und Data Mining. Handbuch der Künstlichen Intelligenz, 3. vydanie, Oldenbourg Verlag München, pp. 517 – 598, (2000)

Autor: Ján Paralič
Názov: Objavovanie znalostí v databázach
Vydanie: prvé
Náklad: 150
Rozsah: 80 strán
Vydavateľ: elfa s.r.o., Letná 9, Košice
Formát: b5
Tlač: elfa s.r.o., Letná 9, Košice
Vytlačené: január 2003

ISBN ??-?????-??-?