Data mining v telefonní ústředně
Data mining of switchboard

Bakalářská práce

Tomáš Bumba

Vedoucí práce: Ing. Jan Jára, Ph.D.

Jihočeská univerzita v Českých Budějovicích

Pedagogická fakulta

Katedra informatiky

Rok 2009

Prohlášení

Prohlašuji, že svoji bakalářskou práci jsem vypracoval samostatně pouze s použitím pramenů a literatury uvedených v seznamu citované literatury.

Prohlašuji, že v souladu s § 47b zákona č. 111/1998 Sb. v platném znění souhlasím se zveřejněním své bakalářské práce, a to v nezkrácené podobě elektronickou cestou ve veřejně přístupné části databáze STAG provozované Jihočeskou univerzitou v Českých Budějovicích na jejích internetových stránkách.

V Českých Budějovicích dne 3. 12. 2009

Anotace

Výkon výpočetních systémů je dnes natolik dostačující, že se uchovávají objemné databáze dat, které bez použití dalších nástrojů nemají vypovídací schopnost. Jedním z takových systémů je i telefonní ústředna, jejíž databáze bude podrobena bližšímu bádání. Tato databáze obsahuje stovky uskutečněných hovorů, které jsou realizovány skrze telefonní ústřednu, jak v okruhu firmy tak mimo ni. Úkolem této bakalářské práce je nalezení co nejvíce skrytých souvislostí v databázi z telefonní ústředny, které odpovídají navrženým formám lidských vztahů.

Abstract

Nowadays is the system performance of computing systems on such a sufficient level, so there are stored up large databases, but those are useless without using proper software tools. One of those computing systems is as well a central switchboard and its database will be scanned. It consists of hundreds of phone calls, realized throught switchboard, within and without company´s area. The target of following thesis is to discover unapparent relations in the database of central switchboard. And institute those relations into patterns of human relation.

Poděkování

Rád bych poděkoval Ing. Janu Járovi, Ph.D. za odborné vedení, připomínky a cenné rady při vypracování bakalářské práce.

Obsah

1 Úvod

Současný výkon výpočetních systémů je schopen zaznamenat a uchovat velké množství informací. Tyto informace ovšem nemají podstatný význam, ale mají potenciál. Proto se počátkem 90. let minulého století začalo mluvit o dobývání znalostí z databází. Na rozdíl od ostatních statistických metod, metody dobývání znalostí z databází kladou důraz na přípravu, zpracování a interpretaci výsledných znalostí.

Tato práce se bude věnovat data miningu v telefonní ústředně. Testován bude jeden měsíc telefonních hovorů z nejmenované firmy, konkrétně měsíc prosinec roku 2005. Databáze s podrobným výpisem hovorů bude zkoumána autorem vytvořeným nástrojem, aplikaci MS Access a také deklarativnímu programovacímu jazyku, Prologu. Budou zjišťovány vztahy a povahy zaměstnanců a jejich vztah k firmě. Závěrem bude provedeno subjektivní zhodnocení použitých nástrojů a bude zhotoveno možné doporučení firmě, jak snížit náklady za telefonní služby.

Práce je rozdělena do tří základních částí. První částí je teoretický úvod do problematiky data miningu, druhá část se věnuje konkrétním potřebám pro dolování dat v telefonní ústředně a třetí část se věnuje testování a vyhodnocení výsledků.

Všechny použité metody a programy, včetně zdrojových kódů jsou přiloženy CD. Příloha v tištěné podobě jsou ukázky porovnání kódů jazyků C# a SQL.

1.1 Úvod do problematiky

Tato kapitola bude věnována úvodnímu zasvěcení do problematiky bakalářské práce. Bude zde osvětlena problematika telefonní ústředny, data miningu, zdrojů dat a budou zde zmíněny i cíle práce.

1.1.1 Data mining telefonní ústředny

Data mining je definován jako „netriviální extrakce implicitních dříve neznámých a potencionálně užitečných informací z dat[1]“. Jednodušeji řečeno jedná se o hledání informací v datech, které nejsou explicitně vyjádřeny. Ideálním prostředím pro data mining je telefonní ústředna. Telefonní ústředna je zařízení, ke kterému jsou připojeny telefony (pobočky) nebo další ústředny. Hlavním úkolem telefonních ústředen je spojování hovorů. Vybrané parametry těchto hovorů jsou monitorovány a zaznamenávány do souborů.

Z telefonní ústředny pochází i zkoumaná databáze, obsahující kompletní výpis hovorů jednoho měsíce. Telefonní ústředna je ideální místo pro zkoumání, hledání skrytých souvislostí a zajímavostí mezi jednotlivými hovory. Právě tímto směrem se vydává tato bakalářská práce.

V poskytnuté databázi budou hledány speciálně navržené formy, které budou o uživatelích odkrývat určité vlastnosti, popisovat povahy a jistým způsobem i charakter. Následně po zkoumání bude sestaveno doporučení pro danou firmu, které by mělo posloužit jako návod na snížení nákladů za telefonní služby.

1.1.2 Zkoumaná databáze

Od vedoucího práce byla poskytnuta databáze ve formátu textového souboru (příloha CD, databaze.txt), která obsahovala výpis hovorů, čítající 18 569 záznamů, za celý kalendářní měsíc prosinec roku 2005. Databáze byla upravena do výsledné podoby, byly odstraněny nedůležité parametry, které nabývaly ve všech hovorech hodnoty null či 0 nebo byly konstantní. Původní databáze obsahovala svazky nabývající pouze dvou hodnot a to 0 (služební hovor) a 6 (soukromý hovor). Zde vznikla potřeba rozdělení svazku 0 na menší

[1] Fayyad a kol, 1996

skupiny, které jsou popsány dále. Toto rozdělení vzniklo z důvodu zvýšení přesnosti metod, zvláště pak metod, kdy technické hovory velice zkreslovaly výsledky.

Každý uvedený hovor má dané parametry. A to datum a čas, pobočka, název, svazek, volané číslo, místo, operátor, délka, cena.

Pro upřesnění se uvádí, že pobočka je číslo telefonního přístroje dané kanceláře. Každá pobočka má svůj název, který byl použit pouze pro ověření hypotéz a následně byl pro citlivost informací a nedůležitost z této databáze vyřazen.

Svazek je parametr pro určení charakteru hovoru. Svazek nabývá následujících hodnot:

- 0 – hovor veden jako služební,
- 1 – technický hovor,
- 2 – „multiLidi“,
- 3 – faxy,
- 6 – soukromý hovor.

Služební hovory jsou hovory volané na účet firmy a určené pouze pro potřeby výkonu zaměstnání.

Technické hovory využívají zejména modemy, které provádí pravidelné hovory na stejná čísla pro kontrolu funkčnosti volaného zařízení. Tyto technické hovory byly prováděny každých cca 7 minut a velice zkreslovaly výstupy z použitých metod. Podobným druhem hovorů jsou faxy, které mají vlastní pobočky, kde je každému faxu uveden i uživatel.

„MultiLidi“ jsou pobočky, které nemají stálého uživatele, kdy se uživatelé na nich střídají nebo je uživatelů více. Jako příklad může být použito callcentrum nebo spojovatelka, kde není známo přesné jméno uživatele.

Ovšem zaměstnavatel zná osoby využívající pobočku, avšak tyto údaje nebyly poskytnuty. Proto je svazek „MultiLidi“ zařazen do některých forem používaných v této práci.

Soukromé hovory se používají pro soukromé záležitosti zaměstnanců. Jako soukromý hovor by měl být veden každý hovor nesouvisející s výkonem pracovní činnosti. Soukromé hovory jsou uživatelům pobočky účtovány k náhradě, pokud firma nestanovila jinak. S touto kapitolou souvisí nejvíce formy „Kukaččí vejce“ a „Soukromé/služební hovory“, o kterých bude zmíněno níže.

Volané číslo je parametr ve tvaru telefonního čísla, které bylo voláno z dané pobočky. Vyskytují se zde i chyby či omyly, které musely být v daných metodách vyfiltrovány, jako například pouze vytočená předvolba a pak zavěšeno. V těchto případech funguje ústředna následujícím způsobem: V okamžiku zvednutí sluchátka je ústředna v pohotovosti a v případě stisknutí prvního tlačítka ústředna okamžitě začíná vytáčet číslo, které bylo stisknuto. Princip je podobný jako na starších telefonních aparátech, kdy se pro vytočení cifry muselo otáčet celým ciferníkem.

Místo označuje zeměpisnou polohu, kam směřoval telefonní hovor. Většinou bývá uveden kraj, například Jihočeský, Moravskoslezský apod. V jiných případech je označena země jako Rakousko, Německo a další. Za skutečnosti, že není místo uvedeno, je tento parametr považován za výjimku, případně chybu.

Operátor označuje provozovatele telefonní sítě, do které je hovor směřován. Firma má s každým operátorem domluvené tarify, podle kterých se řídí ceny hovorů.

Délka hovoru se počítá od okamžiku vytočení první cifry na telefonním přístroji až do ukončení hovoru zavěšením sluchátka. Délka je uvedena v minutách a sekundách.

Cena hovoru se vypočítá vynásobením délky hovoru a ceny za jednotku hovoru. Cena je uvedena v korunách a zaokrouhlena na dvě desetinná místa.

1.2 Cíle práce

Cílem je seznámení a praktické vyzkoušení technologií dobývání dat z databáze telefonní ústředny. Získané informace jsou dosazeny do vymyšlených forem lidských vztahů a povah. Významné je porovnání efektivity mezi autorovým nástrojem, speciálně vytvořeným pro daný úkol, aplikací MS Access 2003 a deklarativním jazykem Prologem.

2 Dobývání znalostí z databází[2]

O dobývání znalostí z databází (Knowledge Discovery in Databases, KDD) se začalo ve vědeckých kruzích mluvit počátkem 90. let minulého století. První impuls přišel z Ameriky, kde se na konferencích věnovaných umělé inteligenci pořádaly první workshopy věnované této problematice. Nebyla to ale jen umělá inteligence (přesněji řečeno metody strojového učení), které stály u zrodu dobývání znalostí z databází. Databázové technologie jsou osvědčeným prostředkem pro uchování rozsáhlých dat a vyhledávání informací v nich obsažených. Statistika je prostředek pro modelování a analyzování závislosti v datech. Dlouhou dobu se tyto disciplíny vyvíjely nezávisle, až přišel okamžik, kdy rozsah automaticky sbíraných dat přerostl uživatelům přes hlavu a vznikla potřeba využívat automaticky sbíraná data pro podporu rozhodování ve firmách. Celé to bylo podpořeno zájmem bonitně silných uživatelů o tyto aplikace. Tento zájem dal vzniknout (a hlavně věhlas) dobývání znalostí z databází. Dnes není nic neobvyklého, že firmy zabývající se počítačovou problematikou používají v reklamách termíny jako data mining, knowledge discovery, nebo business inteligence.

Dobývání znalostí z databází (KDD) lez definovat jako „netriviální extrakci implicitních dříve neznámých a potencionálně užitečných informací z dat[3]“. Zpočátku se pro tuto oblast razily nejrůznější názvy: information harvesting, data archeology, data destilery, data dredging. Nakonec ovšem zvítězila hornická metafora – dobývání a dolování znalostí z dat (data mining). Dnes jsou metody data miningu tvořeny kroky selekce, předzpracování, transformace, „dolování“ (data mining) a interpretace.

[2] Berka, 2003

[3] Fayyad a kol, 1996

Hlavním cílem procesu dobývání znalostí z dat je získat co nejvíce užitečných informací vhodných k řešení daného problému.

Prvním úkolem je stanovení přesné specifikace problému, který je třeba řešit.

Po specifikaci problému je třeba získat všechna dostupná data, která mohou být použita pro řešení problému. Znamená to posoudit všechny dostupné zdroje a zvážit, zda odpovídají či souvisí s daným problémem.

Často je také vhodné uvažovat i o externích datech popisujících prostředí ve kterém se analyzované děje odehrávají. To slouží zejména pro zpřesnění představy.

Výběrem metody se musí zvolit vhodné metody analýzy dat. V rámci dobývaní znalostí z databází je používána řada typů metod analýzy dat. Ve většině případů je k řešení konkrétní úlohy zapotřebí kombinovat více různých metod. Mezi používané typy metod patří například klasifikační metody, metody explorační analýzy, metody pro získávání asociačních pravidel, rozhodovací stromy, genetické algoritmy, Bayesovské sítě, neuronové sítě, velmi používané jsou i metody vizualizace a další.

V rámci předzpracování dat se data získaná k řešení specifikovaného problému připravují do formy vyžadované pro aplikaci vybraných metod. V řadě případů se může jednat o značně náročné a komplikované výpočetní operace.

Data mining zahrnuje aplikaci vybraných analytických metod pro vyhledávání zajímavých vztahů v datech. Obvykle jsou jednotlivé metody aplikovány vícekrát, hodnoty vstupních parametrů jednotlivých běhů závisí na výsledcích předchozích běhů. Zpravidla se nejedná o aplikace metod jen jednoho typu, ale jednotlivé typy se kombinují na základě průběžných výsledků.

Cílem interpretace je nezbytné zpracování obvykle značného množství výsledků jednotlivých metod. Některé z těchto výsledků vyjadřují skutečnosti, které jsou z hlediska uživatele nezajímavé nebo samozřejmé. Některé výsledky je možné použít přímo, jiné je nutné vyjádřit způsobem srozumitelným pro uživatele. Výstupem může být analytická zpráva, zobrazení výsledku v grafickém rozhraní nebo i provedení vhodné akce jako například zapnutí monitorovacího programu apod.

2.1 ÚLOHY

V případě dobývání znalostí z databází můžeme mluvit o různých typech úloh. Jsou to především[4]:

- klasifikace nebo predikce,
- deskripce,
- hledání „nuggetů“.

Při klasifikaci (třídění, řazení do skupin), popřípadě predikci (předpovědi), je cílem nalézt znalosti použitelné pro klasifikaci nových případů – zde je požadováno, aby získané znalosti co nejlépe odpovídaly danému konceptu. Je dávána přednost přesnosti pokrytí na úkor jednoduchosti. Rozdíl mezi klasifikací a predikcí spočívá v tom, že u predikce hraje důležitou roli čas. Ze starších hodnot se pokoušíme odhadnout vývoj v budoucnosti (např. předpověď počasí nebo pohybu cen akcií).

Při deskripci (popisu) je cílem nalézt dominantní strukturu nebo vazby, které jsou skryté v daných datech. Požadují se srozumitelné znalosti pokrývající daný koncept. Dává se tedy přednost menšímu množství méně přesných znalostí.

[4] Klosgen a Zytkov, 1997

Hledáme-li nuggety, požadujeme zajímavé (nové překvapivé) znalosti, které nemusí plně pokrývat daný koncept.

Úlohy dobývání znalostí lze nalézt v celé řadě aplikačních oblastí:

- Segmentaci a klasifikaci klientů banky (např. rozpoznání problémových nebo naopak vysoce bonitních klientů),
- predikci vývoje kurzů akcií,
- predikci spotřeby elektrické energie,
- analýze příčin poruch v telekomunikačních sítích,
- určení příčin poruch automobilů.

2.2 Metodiky

S postupem času vývoje data miningu začaly vznikat metodiky, jejichž cílem je poskytnou uživatelům jednotný vzor pro řešení různých úloh z oblasti dobývání znalostí. Tyto metodiky přenášejí a sdílí zkušenosti získané v úspěšných projektech. Za některými metodikami stojí producenti programových systémů (metodika 5A firmy SPSS), jiné vznikají ve spolupráci výzkumných a komerčních institucí jako „softwarově nezávislé“ (CRISP-DM).

V práci je použita zejména metodika SEMMA. Tato metodika velice vyhovuje potřebám práce. Z počátku byly vybrány vhodné objekty z databáze, které byly následně prozkoumány a upraveny. Podle exploračních výsledků byly sestaveny formy a podle těchto forem byly vytvořeny metody. Závěrem jsou výsledky porovnány podle použitých způsobů data miningu.

2.2.1 Metodika 5A

Metodiku 5A nabízí firma SPSS jako svůj pohled na proces dobývání znalostí. Název metodiky vznikl složením počátečních písmen jednotlivých prováděných kroků:

- Assess – posouzení potřeb projektu

- Access – shromáždění potřebných dat
- Analyze – provedení analýz
- Akt – přeměna znalostí na akční znalosti
- Automate – převedení výsledků analýzy do praxe

Je třeba podotknout, že žádná data nemají význam, jestliže jsou oddělena od kontextu.

2.2.2 Metodika SEMMA

Firma SAS, vychází z vlastní metodiky pro dobývání znalostí z databází. Název SEMMA opět charakterizuje jednotlivé prováděné kroky:

- Sample – vybrání vhodných objektů,
- Explore – vizuální explorace a redukce dat,
- Modify – seskupování objektů a hodnot atributů datové transformace,
- Model – analýza dat: neuronové sítě, rozhodovací stromy, statistické techniky, asociace a shlukování,
- Assess – porovnání modelů a interpretace.

Důraz se klade na snadnou interpretaci výstupů ve formě srozumitelné uživateli.

2.2.3 Metodika CRISP-DM[5]

Metodiky CRISP-DM (CRoss-Indrustry Standard Process for Data Mining) vznikla v rámci Evropského výzkumného projektu. Cílem tohoto projektu bylo navrhnout univerzální postup (tzv. standardní model procesu dobývání znalostí z databází), který bude použitelný v nejrůznějších komerčních aplikacích.

[5] CRISP-DM, 2000

Na projektu spolupracovaly firmy NCR (přední dodavatel datových skladů), DaimlerChrysler (výrobce automobilů), SPSS (tvůrce systému Clementine) a OHRA (holandská pojišťovna). Všechny tyto firmy mají bohaté zkušenosti s reálnými úlohami dobývání znalostí z databází.

2.2.3.1 Hierarchický rozklad metodologie CRISP-DM

Hierarchická struktura se skládá z popisu úkolu ve čtyřech úrovních:

- fáze projektu,
- obecné úlohy,
- specializované úlohy,
- aplikace v procesu.

Na nejvyšší úrovni, úroveň fáze projektu, je data miningový proces uspořádaný do řady fází. Každá fáze se skládá z několika úkolů na druhé (nižší) úrovni.

Druhá úroveň se nazývá obecné úlohy, protože musí být dostatečně obecná, aby pokryla všechny možné data miningové situace. Na této úrovni se rozdělí jednotlivé fáze z první úrovně na obecné úlohy nezávisle na typu projektu. Tyto úlohy musí být úplné a stabilní. Úplné znamená, že pokrývají celý postup data miningu a všechny možné data minigové aplikace. Stabilní znamená, že model by měl být validní i pro ještě nepředvídaný vývoj (například nové modelovací techniky).

Třetí úroveň, specializované úlohy, je úroveň, kde se převádějí obecné úlohy na konkrétní akce podle řešeného problému.

Čtvrtá úroveň, aplikace v procesu (konkretizace), je záznam všech činností, rozhodnutí a skutečných výsledků při data miningu. Proces implementace je uspořádán podle úloh definovaných na vyšší úrovni, ale představuje spíše to, co se skutečně stalo v jednotlivých činnostech, než co by se stát mělo.

To znamená, že na této úrovni dochází k technické realizaci specializovaných úloh.

2.2.3.2 Fáze CRISP-DM

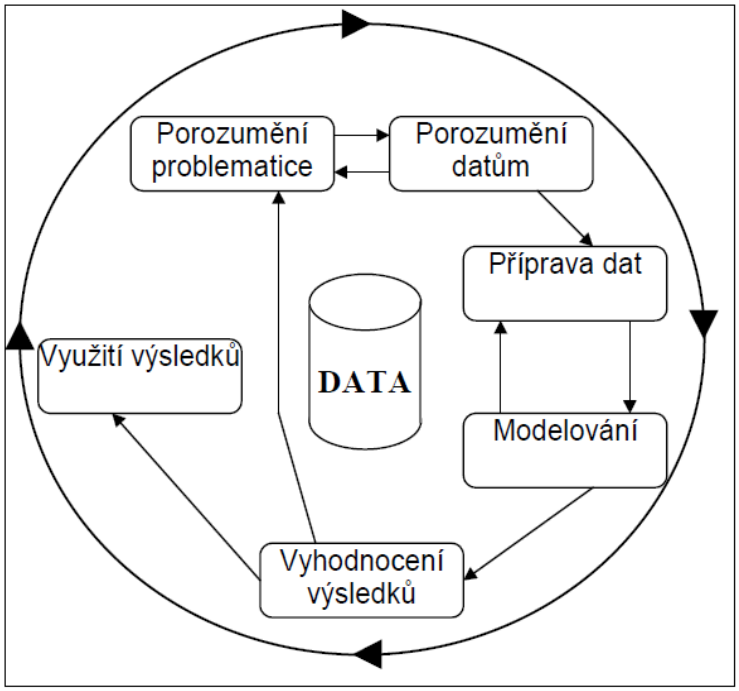
Životní cyklus projektu DM je podle metodiky CRISP-DM tvořen následujícími šesti fázemi:

1. Porozumění problému,
2. porozumění datům,
3. příprava dat,
4. modelování,
5. hodnocení,
6. využití v praxi.

Výsledek dosažený v jedné fázi ovlivňuje volbu následujících kroků. Často je třeba se k některým krokům a fázím vracet.

2.2.3.3 Etapy a úkoly metodologie CRISP-DM

Pro následující objasňování jednotlivých etap a úkolů řešených podle metodiky CRISP-DM bude používáno následující strukturální schéma.



Obrázek 1: Metodiky CRISP-DM

2.2.3.3.1 Porozumění problematice (Business Understanding)

Tato úvodní fáze je zaměřena na pochopení cílů úlohy a požadavků na řešení formulovaných z manažerského hlediska. Manažerská formulace musí být následně převedena do zadání úlohy pro dobývání znalostí z databází.

Manažerský problém, ke kterému jsou pomocí metod DM hledány informace, může být formulován (téměř) bez vazby na informace získávané pomocí metod data miningu z dostupných dat.

V této fázi se rovněž provádí inventura zdrojů (datových, výpočetních i lidských), hodnotí se možná rizika, náklady a přínos použitých metod data miningu a stanovuje se předběžný plán prací.

2.2.3.3.2 Porozumění datům (Data Understanding)

Fáze porozumění datům začíná prvotním převzetím dat. Následují činnosti, které umožní získat základní představu o datech, která jsou k dispozici.

Obvykle se zjišťují různé charakteristiky dat (četnost hodnot atributů, průměrné hodnoty, minima, maxima atd.). Výhodné bývá využití vizualizační techniky.

2.2.3.3.3 Příprava dat (Data Preparation)

Příprava dat zahrnuje činnosti, které vedou k vytvoření datového souboru, který bude zpracováván jednotlivými metodami. Tato data by měla:

- Obsahovat údaje význačné pro danou úlohu.
- Mít podobu, která je vyžadována vlastními analytickými algoritmy.

Příprava dat tedy zahrnuje selekci dat, čištění dat, transformaci dat, vytváření dat, integrování dat a formátování dat. Tato fáze je obvykle nejpracnější částí řešení úlohy. Jednotlivé úkony jsou obvykle prováděny opakovaně, v různém pořadí pro dosažení kvalitního výstupu.

2.2.3.3.4 Modelování (Modelling)

V této fázi jsou nasazeny analytické metody (algoritmy pro dobývání znalostí). Obvykle existuje řada různých metod pro řešení dané úlohy, je tedy třeba vybrat ty nejvhodnější (doporučováno je použít více různých metod a jejich výsledky kombinovat) a vhodně nastavit jejich parametry. Jde tedy opět o iterační činnost a použití analytických algoritmů může navíc vést k potřebě modifikovat data, a tedy k návratu k datovým transformacím z předcházející fáze.

Součástí této fáze je rovněž ověřování nalezených znalostí z pohledu metod dobývání znalostí. To může představovat např. testování klasifikačních znalostí na nezávislých datech, ověřování hypotéz a pravidel..

Znalosti „deskriptivní“ (charakteristika skupiny „nepoctivých“ zaměstnanců z hlediska připravovaných opatření) jsou předkládány personálnímu oddělení. Klasifikační znalosti jsou testovány například na novém vzorku dat.

2.2.3.3.5 Vyhodnocení výsledků (Evaluation)

V této fázi je dospěno do stavu, kdy byly nalezeny znalosti, které se zdají být v pořádku z hlediska metod dobývání znalostí. Avšak dosažené výsledky je třeba ještě vyhodnotit z pohledu manažerů, zda byly splněny cíle formulované při zadání úlohy.

2.2.3.3.6 Využití výsledků

Vytvořením vhodného modelu řešení úlohy obecně nekončí. Dokonce i v případě, že řešenou úlohou byl pouze popis dat, je třeba získané znalosti upravit do podoby použitelné pro zadavatele úlohy.

Podle typu úlohy může využití výsledků znamenat na jedné straně prosté sepsání závěrečné zprávy, na straně druhé pak zavedení systému pro automatickou klasifikaci nových případů.

Ve většině případů je to zadavatel a nikoliv analytik, kdo provádí kroky vedoucí k využívání výsledků analýzy. Proto je důležité, aby pochopil, co je nezbytné učinit pro to, aby mohly být zjištěné výsledky využívány efektivně.

2.3 TŘI HLAVNÍ ZDROJE DATA MININGU

V této kapitole budou rozebrány tři nejhlavnější a nejčastější zdroje data miningu. Jsou to databáze, statistika a umělá inteligence (strojové učení).

2.3.1 Databáze

2.3.1.1 Relační databáze

V prehistorii databází byla data ukládána v jednom velkém „plochém“ souboru (tzv. flat file), ke kterému se přistupovalo indexovanými sekvenčními metodami (ISAM). Soubor byl indexován na základě předpokládaných způsobů dotazování. Nevýhodou bylo, že se informace v záznamech

opakovaly. Další nevýhodou bylo předurčení typu dotazů (byly dány dopředu zvoleným způsobem indexování).

Velkým krokem kupředu bylo zavedení relačních databází. Jeden velký datový soubor byl rozdělen do řady relací. Relační databáze je tedy tvořena:

- Množinou relací – relace je reprezentována dvourozměrnou tabulkou,
- operacemi selekce, projekce a spojení pro manipulaci s tabulkami.

Pro kladení dotazů nabízejí relační databáze dva způsoby:

- QBE (query by example),
- SQL (structured query language).

Oba tyto způsoby navrhla v 70. letech minulého století firma IBM. QBE nabízí uživateli relativně jednoduchý, intuitivní způsob kladení dotazů. V předem připraveném formuláři uživatel vyplní, co ho zajímá, zadá tedy jakousi „masku“, které budou odpovídat nalezené záznamy. Je tedy tento způsob vhodnější pro méně zkušené uživatele.

SQL je naopak určen uživatelům zkušeným. Jde vlastně o jednoduchý programovací jazyk pro definování dat a pro manipulaci s nimi. SQL je daleko mocnější a flexibilnější nástroj než dotazování pomocí indexů. Klade však zvýšené požadavky na uživatele. Uživatel musí znát syntaxi jazyka, navíc musí znát i detailní strukturu databáze (názvy souborů a polí). Tento způsob dotazování tedy příliš nepřirostl k srdcím manažerů a analytiků, pokud se nechtěli učit syntaxi jazyka SQL. V tom případě museli zformulovat dotaz a vydat se za programátorem, který dotaz přeložil do jazyka SQL. V případě, že obdržené výsledky nepřinesly hledanou odpověď, museli dotaz přeformulovat a znovu se vydat do výpočetního centra.

2.3.1.2 EIS

EIS (Executive Information Systems) byl první pokus jak přiblížit dotazování do databáze manažerům. Zavádění EIS bylo spojeno se zaváděním

osobních počítačů ve firmách. Počítače přestaly být doménou programátorů, objevily se na stolech „prostých“ uživatelů. Základním požadavkem se tedy stalo snadné ovládání. Uživatelsky přátelský interface EIS odvedl uživatele od syntaxe SQL a od podmínky znát strukturu databáze, se kterou chtěli pracovat. Analýzu tedy mohl analytik provádět sám z počítače, který měl na svém pracovním stole. Dotaz, které si uživatel vybral v menu, byl pak převeden do jazyka SQL a proveden standardním způsobem. Nevýhodou tohoto přístupu bylo, že uživatel měl k dispozici pouze určitý soubor připravených dotazů. Chtěl-li se zeptat na něco, co tvůrce daného EIS nepředpokládal, byl opět nucen připravit si dotaz v přirozeném jazyce, zajít za programátorem, který dotaz převedl do SQL.

EIS tedy byly sice uživatelsky přátelské, ale málo flexibilní nástroje pro analýzu dat v databázích.

2.3.1.3 OLAP

OLAP (On-line Analytical Processing) konečně nabídl uživatelům obojí; flexibilitu, rychlost i příjemné, intuitivní ovládání. OLAP umožňuje analytikům firmy snadno získávat odpovědi, ve velice srozumitelné podobě. Typické pro OLAP jsou totiž možnosti vizualizace. Grafické rozhraní umožňuje uživateli nahlížet na data jak v numerické podobě, tak v podobě nejrůznějších grafů.

Základem OLAP je pohled na data jako na mnohorozměrnou tabulku nazývanou datová krychle (data cube). Databáze se může převést na datovou krychli tak, že jednotlivé sledované atributy budou tvořit dimenze krychle, buňky krychle pak odpovídají jednotlivým záznamům v databázi. Tento způsob uložení umožňuje různé pohledy na data, ale plýtvá se při něm místem. Řada buněk je v krychli prázdných.

Datová krychle obsahuje jak data z operačních databází, tak dílčí souhrny. Právě tyto souhrny umožňují rychlou odezvu na ad-hoc dotazy uživatele

a flexibilitu systému. Práce s krychlí spočívá v různém natáčení (pivot), provádění řezů (slice), výběru určitých částí (dice) a zobrazování různých agregovaných hodnot. Velmi často lze hodnoty atributů sdružovat do hierarchií. Tyto hierarchie se využívají při práci s krychlí při operacích roll-up a drill-down. Při roll-up se přechází na hierarchicky vyšší, obecnější úroveň, při drill-down se přechází na podrobnější pohled na data; někdy se mluví o různých úrovních podrobnosti (granularity) pohledu na data.

Základní multidimenzionální model má podobu n-rozměrné krychle. To je však podoba v logickém smyslu. Z hlediska implementačního se nabízí několik možností jak tuto strukturu uložit v počítači. Hlavní důvod pro odlišné fyzické implementace logického modelu je především velká řídkost dat a jejich nestejnoměrné rozmístění. Pro implementaci se nabízejí v zásadě dva přístupy:

Hyperkrychle (hypercube) – jedna velká krychle, která obsahuje nástroje pro práci s řídkými daty. Výhodou je jednoduchá struktura a srozumitelnost pro uživatele.

Multikrychle (multicube) – více navzájem propojených menších krychlí obsahujících jen několik dimenzí. Výhodou je efektivní uložení dat.

Uložení dat v krychli ale nepřináší jen výhody. Za rychlost přístupu k datům platíme zvýšenými nároky na datový server. Tyto nároky někdy vedou k tomu, že se místo řešení OLAP, založeného na datové krychli (někdy nazývaného MOLAP – multidimenzionální OLAP), použije tzv. ROLAP – relační OLAP založený na klasické relační databázi. V tomto druhém případě se dotazy OLAP převádějí do klasických dotazů jazyka SQL.

Nutno ještě podotknout, že OLAP není totéž co data mining. OLAP – jednoduše řečeno, mění pohled na danou problematiku. Data mining – hledá v datech nějaké zajímavé souvislosti. OLAP tedy přináší odpovědi na konkrétní, přesně specifikované otázky, ale sám o sobě nic „neobjevuje“.

2.3.1.4 Datové sklady a tržiště

Zatímco OLAP představuje nástroj pro analýzu (a vizualizaci) dat o firmě, datový sklad představuje místo, kde jsou analyzovaná data uložena. Podle W. H. Inmona, který v 80. letech zformuloval koncept datového skladu, je datový sklad

- subjektově orientovaný,
- integrovaný,
- časově proměnný,
- leč stálý

soubor dat sloužící pro podporu rozhodování[6].

Prvním charakteristickým rysem datového skladu je, že je orientován na subjekty, kterými se daná firma zabývá. To je výrazný rozdíl od tzv. produkčních databází, které se zabývají operacemi a transakcemi. Datový sklad neuchovává data, která nejsou vhodná pro podporu rozhodování na manažerské úrovni, produkční databáze uchovávají data potřebná pro operativní řízení bez ohledu na to, zda budou využitelná při strategickém rozhodování.

Vzhledem k tomu, že do datového skladu vstupují data z různých produkčních databází firmy, je důležitá integrace a sjednocení dat. Toto integrování zahrnuje sjednocení názvů stejných ukazatelů, sjednocení měřítek (různý způsob měření týchž veličin – např. doba hovoru v sekundách nebo minutách), sjednocení kódování (např. volací kódy ve firmě) apod.

Všechna data v datovém skladu představují „časový snímek“ dat z produkčních databází sejmutý v určitém okamžiku. Datový sklad je

[6] Inmon,1999

aktualizován off-line v určitých časových intervalech (měsíčně, čtvrtletně, ročně) a je rovněž analyzován odděleně od produkčních databází. Výhodou je, že nešetrný zásah do datového skladu (například dotaz, který vede k zacyklení) neovlivní operativní řízení firmy. Rovněž odezva na dotaz položený do datového skladu je rychlejší, než by byla odezva do produkční databáze. Produkční databáze je totiž plně vytížena zaznamenáváním transakcí a analytikovi by odpovídala jen okrajově. Nevýhodou je, že data v datovém skladu postupně stárnou. Časovou proměnností se tedy myslí v první řadě toto zafixování dat z produkčních databází. Druhé časové hledisko spočívá v tom, že časové údaje jsou v datovém skladu explicitně přítomny jako jedna z důležitých informací.

Dotazy, které do datového skladu směřují uživatelé – analytici, nezpůsobují změnu zde uložených dat. Je tedy datový sklad v tomto ohledu stálý.

Vytvoření datového skladu zahrnuje úkoly jako načtení dat, konverzi dat, čištění a transformaci. Data uložená v datovém skladu představují jakýsi neutrální datový prostor, který není vytvářen s myšlenkou konkrétních analýz. Proto se doporučuje vytvářet v návaznosti na datový sklad řadu specializovanějších datových tržišť (data mart), kam se z datového skladu přesunou data relevantní pro určitý typ analýz.

2.3.2 Statistika

Statistika nabízí celou řadu teoreticky dobře prozkoumaných, zdůvodněných a léty praxe ověřených metod pro analýzu dat. Pro oblast dobývání znalostí mají význam (ať už přímo jako používané metody nebo nepřímo jako zdroj inspirace):

- Kontingenční tabulky – pro zjišťování vztahu mezi dvěma kategoriálními veličinami,
- regresní analýza – pro zjišťování funkční závislosti jedné numerické veličiny na jiných numerických veličinách,

- diskriminační analýza – pro odlišení příkladů patřících do různých tříd,
- shluková analýza – pro nalezení skupin navzájem si podobných příkladů.

Z dalších metod je uvedena ještě korelační analýza (pro posouzení, zda je mezi dvěma numerickými veličinami lineární závislost), analýza rozptylu (pro posouzení rozdílu mezi průměry z různých výběrů) a faktorová analýza (pro zjišťování závislosti jedné veličiny na tzv. faktorech vytvořených jako lineární kombinace jiných veličin).

2.3.3 Strojové učení

Důležitou vlastností živých organismů je schopnost přizpůsobovat se měnícím podmínkám, eventuelně se učit na základě vlastních zkušeností. Schopnost učit se bývá někdy dokonce považována za definici inteligence. Je proto přirozené, že vybavit touto vlastní inteligencí i systémy technické je jedním z cílů umělé inteligence. Navíc v řadě praktických případů, kdy není dostatek apriorních znalostí o řešeném problému, ani jinak postupovat nelze.

Prvky učení můžeme pod různými názvy nalézt v řadě vědních disciplín. Ve statistice se objevují explorační analýzy dat (exploratory data analysis) nebo inteligentní analýzy dat (inteligent data analysis), v umělé inteligenci se hovoří o metodách rozpoznávání obrazů (pattern recognition), či strojového učení (machine learning) nebo automatizovaného získávání znalostí (automated knowledge acquisition), v kybernetické teorii řízení najdeme adaptivní a učící se systémy, v souvislosti se získáváním znalostí z databází (knowledge discovery in databases) se používá termín dolování z dat (data mining). V různých disciplínách se k problematice učení přistupuje z různých pohledů, používá se rozdílná terminologie, různé metody reprezentace znalostí i různé algoritmy pro získávání znalostí či jejich využívání.

3 Řešení problematiky data miningu

Práce se v této kapitole bude zaobírat zejména použitými nástroji a hledanými formami vztahů.

3.1 Nástroje

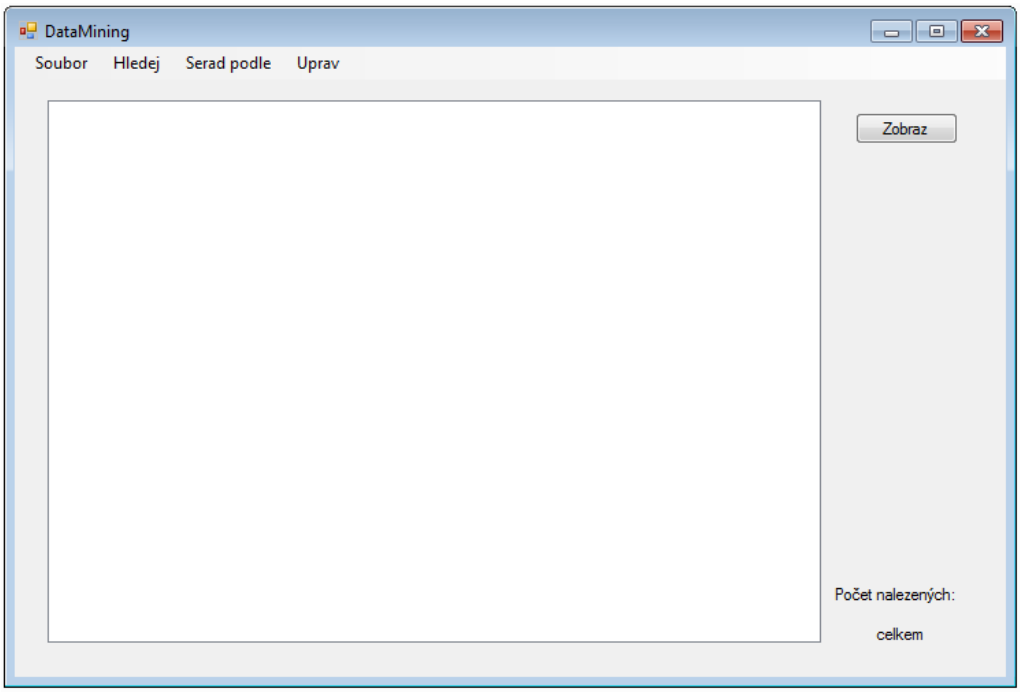
Pro potřeby této bakalářské práce byly využity násedující nástroje a techniky. Stěžejním je jazyk C# a v něm naprogramovaný program DataMining. Tento nástroj byl vytvořen autorem bakalářské práce a je specializován na daný problém. Dalším nástrojem byl použit Microsoft Office Access 2003, program používaný pro správu relačních databází. Jako protiklad daným prostředkům poslouží deklarativní programovací jazyk Prolog a program SWI-Prolog. Dále vznikla myšlenka zkusit použít přímo specializované nekomerčně šířené softwary. O výsledcích se dozvíte dále. Veškeré použité zdrojové kódy jsou k nalezení na přiloženém CD. Ukázky zdrojových kódů hlavních metod, použitých v programu DataMining a MS Access, jsou i v tištěné příloze.

3.1.1 DataMining

DataMining je program vytvořený autorem bakalářské práce, ovládání je intuitivní pro všechny uživatele operačního systému Microsoft Windows. Pro vývoj programu byl použit programovací jazyk C# a software Microsoft Visual Studio 2008 poskytnutý pod licencí MSDN Academic Alliance od společnosti Microsoft.

Program se spouští souborem „DataMining.exe“ nacházejícím se na přiloženém CD ve složce Programy\DataMining\DataMining\bin\Debug. V této složce se nachází i soubor s databází telefonních hovorů pojmenovaný „db1.txt“. Jako vstup program používá zmíněnou databázi, se kterou je následně pracováno. Jako vizuální výstup slouží „Listbox“. Jako výstup do souboru slouží funkce „Ulož“, nacházející se v menu v položce „Soubor“, která vytvoří textový soubor naplněný daty z „Listboxu“.

Po spuštění se program dotáže, které svazky chce uživatel odstranit. Tato funkce byla vytvořena pro případ, že by uživatel nechtěl pracovat s určitými typy hovorů. Následně se načte okno se samotným programem a v tuto dobu je již načtena databáze a lze se na ní dotazovat pomocí metod. Možné dotazovací metody se nalézají v menu „Hledej“. Zobrazené výsledky lze seřadit podle atributů, nalézajících se v menu „Seřaď podle“. Poslední položka menu je „Uprav“, zde se nalézá tatáž volba jako na začátku programu a to výběr svazků, které chceme používat. Možnosti úpravy vzhledu a výstupu obsahuje tlačítko zobraz, sloužící k zobrazení nebo naopak skrytí určitých zobrazovaných atributů v „Listboxu“. Počet nalezených hovorů se zobrazuje v pravém dolním rohu v „Labelu“ s názvem „Počet nalezených“. Program se ukončí stisknutím křížku v horním pravém rohu.



Obrázek 2: Program DataMining

3.1.2 MS Access 2003

Microsoft Access 2003 (plným názvem Microsoft Office Access 2003) je nástroj pro správu relačních databází od společnosti Microsoft, který je typicky součástí Microsoft Office a kombinuje relační Microsoft Jet Database Engine s grafickým uživatelským rozhraním. Jedná se o program pomocí kterého uživatel může rychlým a efektivním způsobem zpracovávat a vyhodnocovat data, která jsou uložena v databázi. MS Access umí přistupovat k datům z Access/Jet, Microsoft SQL Server, Oracle či ke kterékoliv další databázi přes rozhraní ODBC.

Do MS Access byla naimportována databáze v podobě textového souboru, ze které byla vytvořena tabulka s názvem „databaze“. Metody jsou tvořeny pomocí výběrových dotazů a dotazují se na atributy v tabulce. Pro spuštění programu je nutné mít nainstalovaný MS Office Access, soubor s tabulkou a dotazy s názvem „dataMiningAccess“ se nachází na přiloženém CD ve složce „Programy\DataMining“.

3.1.3 SWI-Prolog

SWI-Prolog nabízí komplexní Free Software Prolog prostředí, licencované pod Lesser GNU Public Licence. Jeho vývoj začal v roce 1987 a byl řízen potřebami reálných aplikací. SWI-Prolog je široce používán ve výzkumu a vzdělání, jakož i pro komerční aplikace.

Prolog je logický programovací jazyk. Patří mezi tzv. deklarativní programovací jazyky, ve kterých se popisuje pouze cíl výpočtu, přičemž přesný postup jakým se k výsledku dostává je ponechán na systému. Prolog se snaží o abstraktní vyjádření faktů a logických vztahů mezi nimi. Syntaxe jazyka je velice jednoduchá a snadno použitelná z důvodu prvotního určení pro počítačově nepříliš gramotné uživatele. Prolog je založen na predikátové logice prvního řádu; konkrétně se omezuje na Hornovy klauzule. Běh programu je pak představován aplikací dokazovacích technik na zadané klauze.

U deklarativního programování bylo postupováno nepatrně jinak. Zde nebyla možnost načítat data přímo do programu a proto musela být importována do zdrojového souboru s názvem „zdrojProlog.pl“ (příloha CD, zdrojProlog.pl). Importovaná data byla navíc upravena do podoby nejvíce vyhovující softwaru SWI-Prolog. Z databáze byly z kapacitních důvodů vyřazeny všechny technické hovory a faxy, tudíž zbyly pouze hovory soukromé, služební a multiLidi. Další pro-vedené úpravy jsou tyto:

- Datum a čas se změnil pouze na číselný údaj označující den (např. 10.12.2005 se změnilo na cifru 10),
- čas ve formátu HH:MM se změnil na oddělené hodiny a minuty (např. 12:34 se změnilo na 12, 34),
- délka hovoru ve formátu MM:SS se změnila na oddělené minuty a sekundy (např. 2:34 bylo upraveno do podoby 2, 34),
- v parametru cena hovoru byla desetinná čárka nahrazena desetinou tečkou,
- byly odstraněny parametry místo a operátor, jednak z kapacitních důvodů a také z důvodů nevelké důležitosti.

Zde je vidět ukázka klauzule jednoho hovoru. Je ve formátu: h(den, hodina, minuta, pobočka, svazek, volané číslo, délka v minutách, délka v sekundách, cena).

```
h(1,06,07,329,2,971104411,0,43,2.58).
```

3.1.4 Nekomerčně distribuovaný software

3.1.4.1 Který systém zvolit?

Je obtížné zvolit si jeden software pro dobývání znalostí z databází. V současné době totiž neexistuje nějaký standardní, všeobecně uznávaný

systém. Jedním z mnoha vodítek by mohlo být, jak jednotlivý produkt podporuje jednotlivé kroky procesu dobývání znalostí. Běžně se posuzuje:

- Zda systém umožňuje snadný přístup k externím datovým zdrojům, zda umožňuje číst standardní formáty dat (databáze, tabulkové kalkulátory, ASCII) a zda umožňuje pracovat s rozsáhlými soubory.
- Jaké metody (a jak propracované algoritmy metod) systém nabízí, jestli nabízí vizualizační možnosti (kancelářská grafika, statistické grafy), zda nabízí průvodce pro usnadnění práce s algoritmy, zda nabízí nástroje pro datové transformace, manipulaci se soubory, zda nabízí pracovní prostředí umožňující automatizovat opakovaně prováděné kroky (skripty, ukládání sekvence kroků).
- Zda systém nabízí snadné začlenění výstupů do již používaných aplikací (OLE, šablony pro přeformátování výstupu, podporu vytváření reportů).
- Zda lze vytvářet samostatně běžící aplikace (generování spustitelného kódu).

Důležitou roli hraje také fakt, zda je požadován komerční či nekomerční software. Ceny komerčních systémů dobývání znalostí mnohonásobně převyšují ceny například běžného kancelářského softwaru. Podstatná je také snadnost ovládání a dokumentace k systému. Musíme zvážit, zda hledáme univerzální systém použitelný pro různé aplikační oblasti nebo zda hledáme systém vyvinutý pro jeden typ aplikací.

3.1.4.1.1 LISP-Miner

LISp-Miner je systém vyvíjený na VŠE v Praze. Systém v současnosti nabízí dvě základní metody – tvorbu asociačních pravidel a tvorbu klasifikačních pravidel. Co se týká asociačních pravidel, systém navazuje na mnohaletý výzkum v této oblasti spojený s metodou GUHA, konkrétně jde o proceduru GUHA 4FT-Miner. Algoritmus pro tvorbu klasifikačních pravidel

byl převzat ze systému KEX, rovněž vyvinutý na VŠE. Systém také nabízí bohaté možnosti pro předzpracování založené na SQL. Systém je úzce svázán s databázemi (současná implementace s databází MS Access), ze kterých načítá data i kam ukládá výsledky. Tento rys umožňuje uživatelům vytvářet vlastní interpretační procedury nad asociačními pravidly (tzv. open 4FT-Miner).

Tento software byl vyzkoušen a prozkoumán. Následně na něm byla otestována databáze, avšak bylo zde několik problémů. Jedním ze stěžejních problémů bylo načtení databáze z MS Access, kde program LISP-Miner nevyužíval běžný atribut jakým je DateTime. Dalším z problémů bylo samotné pochopení fungování systému, s tímto problémem nepomohla ani nepříliš podrobná dokumentace a proto bylo poměrně obtížné se systémem pracovat. Po bližším zkoumání bylo zjištěno, že pro problém data miningu telefonní ústředny je tento software nevhodný, jelikož používá pouze asociační a rozhodovací pravidla a žádné další možnosti nenabízí. Proto bylo zkoumání na tomto programu zastaveno.

3.1.4.1.1.1 Weka

Weka je systém vyvinutý na univerzitě Waikato na Novém Zélandu. Přestože jde o freeware volně dostupný na internetu (program je šířen jako open source software spadající pod licenci GNU), v ničem si nezadá s komerčními systémy. Weka nabízí celou řadu algoritmů pro učení i předzpracovávání, známých v akademickém světě. K dispozici jsou také možnosti vizualizace a kombinování modelů. Systém je řešen jako knihovna programů v jazyce Java volaných z jednotného (grafického) rozhraní. Většina modelů si ale ponechává původní textový výstup.

Tento systém se podle úvodu zdál být opravdu dobrým a kvalitním nástrojem. Jeho schopnosti a možnosti jsou opravdu rozsáhlé, avšak nepodařilo se spojit ani s jedním typem v práci používané databáze. Vyžadoval pouze speciální formáty, kterých nebylo možno dosáhnout z různých důvodů

(nekompatibilita s operačním systémem, nedostatek technických prostředků apod.). To byla jedna z příčin, proč byl systém také vyřazen ze seznamu možných nekomerčních softwarů.

3.1.4.1.1.2 Ostatní nekomerční programy

Bohužel podobné problémy se objevovaly i u ostatních programů. Jednou z častých obtíží byly chybně nahrané soubory nebo viry. Zvláštní byl případ, kdy byl instalační soubor stahován z oficiálních stránek, avšak stažený soubor obsahoval trojského koně a proto nebyl záměrně instalován a také nemohl být testován.

Bohužel nejčastějším případem, kdy programy selhávaly bylo nedoladění některých výjimek a nedostatečná dokumentace k programům. Často byly dokumentace psány spíše pro vývojáře než pro uživatele. A proto bylo používání programu a zejména využití možných metod prakticky nemožné. Některé programy využívají jakého grafického rozhraní, kdy jednotlivé metody představují ikony. Tyto ikony se přetahují na plochu a může s nimi být dále pracováno, pokud uživatel zvládne správně nastavit atributy či načíst zdrojová data. K tomu aby tyto parametry mohly být nastaveny je nutná dokumentace s patřičným podrobným vysvětlením, ne jen ledajaký text stručně popisující funkci programu.

3.2 Hledané formy

V této kapitole jsou rozebrány hledané formy v databázi. Předem je poznamenáno, že byly vyřazeny všechny pobočky technického charakteru (svazek 1 a 3), jakými jsou například modemy nebo faxy. Pokud není psáno jinak, v hledaných metodách nejsou žádná další omezení a restrikce. Uvedené názvy jsou pouze pracovní nikoliv oficiálně vedené. Hledané formy jsou rozděleny na dvě hlavní části a to na hlavní a pomocné, sloužící zejména jako pomoc k pochopení případně ověření některé z hlavních forem.

U hlavních forem je i popis funkce metody, v jazyku C#, navržené podle metod. Zdrojové kódy jsou k nalezení v tištěné podobě i na přiloženém CD. Navíc v tištěné podobě jsou přiloženy i zdrojové kódy z MS Access ve formě SQL dotazů, pro porovnání obtížnosti tvorby metod.

3.2.1 Hlavní formy

3.2.1.1 Obědáři

Jsou pobočky volající si alespoň 5× za měsíc mezi 11 a 12 hodinou. U této metody nerozhoduje svazek volajícího.

Metoda vytvořena pro hypotézu, kdy lidé pracují v oddělených kancelářích a alespoň v době oběda chtějí být ve společnosti. Proto se mohou během hovoru domlouvat kam, kdy a v kolik hodin půjdou na oběd.

Metoda vytvoří seznam všech hovorů, které byly uskutečněné mezi 11. - 12. hodinou. Následně v cyklu zkontroluje, zda se pobočka v seznamu vyskytuje vícekrát. Pokud je podmínka splněna a pobočka se vyskytuje více než 5× uloží se do výsledného seznamu. Ten je posléze vypsán.

3.2.1.2 Soukromé/služební hovory

Je pobočka volající na totéž telefonní číslo jednou jako soukromý, jindy jako služební hovor.

Záleží jen a pouze na zaměstnavateli zda nechá hovory vedené jednou jako soukromé (tudíž si je pobočka platí sama) a jednou jako služební (hovor jde na účet firmy) zaměstnanci zaplatit. Jedná se zde o evidentní podvod. Je nepravděpodobné volání jednoho telefonního čísla dvěma způsoby. Buď je hovor soukromého charakteru nebo služebního.

Tato metoda nejprve vytvoří z původního seznamu, seznam hovorů, který obsahuje pouze soukromé hovory. Následně tento seznam porovnává s původním a kontroluje, zda volané číslo se rovná volanému číslu z původního

seznamu a zároveň jestli se rovná i volající pobočka. Pokud ano, přidá hovor do výsledného seznamu a po dokončení ho vypíše.

3.2.1.3 Kukaččí vejce:

Domácí pobočka volající telefonní číslo jako soukromý hovor a následně je voláno totéž telefonní číslo jako služební hovor z jiné pobočky.

Jedná se o hypotézu, kdy domácí pobočka volá telefonní číslo a následně si uvědomí, že kolega z vedlejší kanceláře není na svém pracovišti a jeho kancelář je volná. Proto si jde zavolat do vedlejší kanceláře. Zde je velká pravděpodobnost informačního šumu. Může také nastat případ, kdy volané číslo je obecně známé, například když je volán hydrometeorologický ústav pro předpověď na daný den a následně podobná informace zajímá i jinou pobočku. V tomto případě se nejedná o kukaččí vejce, ale pouze o informační šum, který nelze více odfiltrovat.

Tato metoda nejprve vytvoří z původního seznamu, seznam hovorů, který obsahuje pouze soukromé hovory. Následně tento seznam porovnává s původním a kontroluje, zda volané číslo ze seznamu soukromých hovorů se rovná volanému číslu z původního seznamu a zároveň jestli se nerovná volající pobočka. Pokud je podmínka splněna, přidá hovor do výsledného seznamu
a po dokončení jej vypíše.

3.2.1.4 Zahraniční hovory

Pobočky volající do zahraničí jak soukromě, tak i služebně.

Je běžnou záležitostí, že mají firmy s operátory domluvené levnější hovory než jaké mají v nabídkách běžní zákazníci. Proto se může vyplatit volat do zahraničí z firemní pobočky i za cenu, že by byl hovor veden jako soukromý a tudíž by byl účtován zaměstnanci k náhradě. Tato metoda slouží pro zobrazení všech zahraničních hovorů.

Tato metoda není zcela přesná. Po bližším zkoumání databáze bylo zjištěno, že hovory do zahraničí mají o jednu či více cifer více než běžné hovory. Proto se vypisují jen hovory s devíti a více ciframi. Zároveň byla zjištěna skutečnost, že místní hovory začínají předvolbou „0042“, proto jsou z metody filtrovány. Více rozdílů nebylo zjištěno, proto je zde menší přesnost.

3.2.1.5 Začátek telefonního dne

První uskutečněný hovor v jednotlivých dnech.

Zde je původem hypotéza povah zaměstnanců, kteří jsou velice rádi dochvilní a potrpí si na přesnost. Proto jsou v zaměstnání raději dříve než aby přišli později. Informační šum zde tvoří například pobočky bufetů, kdy jsou zásoby objednávány s dostatečným předstihem dopředu, aby bylo možno všechny pokrmy připravit do času obědů. Dalším příkladem je například pobočka spojovatelky, která je v pohotovosti celý den a mimo jiné přijímá i hovory nouzového charakteru, jakýmiž jsou například havárie a případné výpadky.

Metoda vytvoří seznam hovorů a seřadí ho vzestupně podle data. Vytvoří pomocnou proměnou. Poté se porovnává, zda datum obsažené v seznamu je také v proměnné. V případě, že to pravda není, vezme první prvek ze dne a uloží ho do výsledného seznamu. Po dokončení se seznam vypíše.

3.2.1.6 Kontroloři

Pobočky volající opakovaně stejná telefonní čísla alespoň 10× za měsíc a to nejdéle jednu minutu.

Kontroloři vznikli spojením forem „Žárlivec“, „Starostlivý rodič“, „Starostlivý potomek“. Tyto formy mají společné kontrolování. „Žárlivec“ kontroluje svou partnerku či partnera. „Starostlivý rodič“ volá svým dětem obvykle ráno z různých důvodů. Například jestli nezaspaly, jestli si vzaly

svačinu atd. „Starostlivý potomek“ volá většinou dopoledne či odpoledne svým rodičům.

Metoda vytvoří seznam hovorů trvajících méně než 1 minutu. Následně se tento seznam prohledává a zjišťuje se, zda se hovor vyskytuje se stejnými parametry (pobočka, volané číslo) vícekrát, pokud ano, přidá se do výsledného seznamu. Druhý cyklus zjišťuje zda volané číslo se ve výsledném seznamu vyskytuje více než 10×, pokud ano, tak se všechny výskyty přeřadí do nového seznamu a ten je po dokončení vypsán.

3.2.1.7 Mimo pracovní dobu

Pobočky volající mimo běžnou pracovní dobu.

Pobočky splňující tyto parametry jsou pozoruhodné. Je jen málo zaměstnanců, kteří zůstávají dobrovolně v zaměstnání déle než musejí. Nicméně i takoví zaměstnanci existují. Tato metoda vznikla spíše na základě hypotézy, která byla založena na faktu, že zaměstnanec nestihl přidělený úkol během pracovní doby, tak ho musí dodělávat po pracovní době. Jedním ze speciálních případů je tzv. „zlobivá uklízečka“, která si při úklidu kanceláří tu a tam zavolá z volné pobočky. Tento případ je těžko prokazatelný, pokud nebyla uklízečka chycena při činu. Ovšem firma má možnost z docházkového systému získat informace, kdo byl v inkriminovanou dobu na pracovišti a podle toho se rozhodnout, případně udělat určitá opatření.

Vytvoří se seznam poboček splňující podmínky. Podmínkami jsou hovory, které nejsou uskutečněny v pracovní době, tj. mezi 8 a 16 hodinou. Výsledný seznam je následně seřazen a vypsán.

3.2.1.8 Nejpilnější pobočka

Zobrazí pobočku s nejvíce provolaným časem.

Zde je původcem hypotéza velice vytíženého zaměstnance, který neustále vykonává svou náplň práce a důsledkem toho má nejvíce provolaného času.

Opakem je pracovník, který nemá dostatek vytížení a proto telefonuje, aby se zabavil.

Metoda vytvoří seznam poboček. Následně je proveden součet parametrů „delka“ a „cena“ u jednotlivých poboček. Výsledek je seřazen vzestupně podle ceny a vypsán.

3.2.1.9 Barevné linky

Zobrazí volání na barevné linky.

Barevné linky se liší zejména účelem, pro který jsou zřízeny. Odlišují se názvem barvy, která se určuje podle předvolby.

Vytvoří se seznam hovorů splňujících požadavek na předvolbu barevné linky. Ten se po skončení vypíše.

Rozlišují se následující barevné linky.

3.2.1.9.1 Zelená linka (800)

Služba na účet volaného.

Pro volajícího je volání na 800 linku odkudkoliv bezplatné. Náklady na tato volání jsou účtovány volanému. 800 linky jsou dostupné z kteréhokoliv telefonního přístroje v ČR bez ohledu na poskytovatele. Tento způsob služby umožňuje nejpohodlnější způsob získávání informací.

3.2.1.9.2 Bílé linky (840/841/848)

Služby na účet volajícího.

Bílá linka je zpřístupněna na jediném telefonním čísle s jednotnou sazbou volání z celé ČR a s nulovými náklady na provoz bílé linky (náklady hradí v plné výši volající). Naopak, za příchozí volání bude inkasována provize.

3.2.1.9.3 Modré linky (810/844/855)

Služby se sdílenými náklady.

Modrá linka umožňuje komunikaci za sdílený tarif, kdy část hovorného platí volající a část volaný. Díky tomu patří tato služba k jedněm z nejefektivnějších.

3.2.1.9.4 Žlutá linka (900)

Služba se zvláštním tarifem pro obchodní a odborné informace.

Žlutá linka umožňuje volajícím snadný přístup k placeným informacím. Za tyto informace je volajícímu účtován jeho operátorem zvolený tarif. Část z takto vybraných poplatků připadne volanému – poskytovateli informací. Volající k této službě přistupuje vytočením čísla ve tvaru 900 XX YY YY, kde XX určuje cenu za minutu volání na tuto linku.

3.2.1.9.5 Duhová linka (906)

Služba se zvláštním tarifem pro soutěže a hry po telefonu, seznamky, inzerci, horoskopy a obdobné služby.

Duhová linka je určena pro společnosti podnikající v oblasti zábavy a her. Firma provozující duhovou linku je oprávněna poskytovat volajícím soutěže, loterie a jiné podobné hry na základě povolení Ministerstva financí České republiky. Současně má zajištěno, že duhová linka nebude spojována se službami pro dospělé. Volající přitom k této službě přistupuje vytočením čísla ve tvaru 906 XX YY YY, kde XX určuje cenu za minutu volání na tuto linku.

3.2.1.9.6 Linka 909 se zvláštním tarifem

Služby pro dospělé (+18 let).

Linka 909 nabízí zábavu pro dospělé, nicméně není určena pouze pro společnosti podnikající v této oblasti zábavy. Volající k této službě

přistupuje vytočením čísla ve tvaru 909 XX YY YY, kde XX určuje cenu za minutu volání na tuto linku.

3.2.1.10 Nejvýřečnější pobočka

Zobrazí nejvýřečnější pobočky volající číslo 1180.

Telefonní číslo 1180 je službou, kterou poskytuje Telefonica O2. Zavoláním na 1180, lze zjistit libovolné číslo pevné linky, faxu nebo Zelené linky a to v rámci celé České Republiky. Nově lze také získat telefonní čísla, která poskytli ostatní mobilní operátoři. Databáze telefonních čísel je denně aktualizována.

Metoda tvoří seznam z poboček, které volali telefonní číslo 1180. Výsledný seznam je seřazen sestupně, aby bylo zřetelně vidět, která pobočka je nejvýřečnější, která volala nejkratší dobu.

3.2.2 Pomocné formy

3.2.2.1 Hledej číslo

Metoda sloužící pro hledání volaného telefonního čísla v celé databázi.

Tato metoda pochází z předpokladu, že některé pobočky volají určitá telefonní čísla pro své zájmy. Jedním z příkladů jsou sázkaři – lidé volající do sázkových kanceláří, ohledně výsledků zápasů, nebo volají na erotické linky apod.

3.2.2.2 Výjimky

Zobrazí pobočky volající na telefonní číslo kde není uveden operátor nebo některý z jiných parametrů.

Tato metoda byla vytvořena pro objevení chyb telefonní ústředny.

3.2.2.3 Vyber den hodinu

Zde lze vyfiltrovat hovory v daných hodinách a dnech.

Uživatel si může vytvořit vlastí filtr. V práci použito obzvláště pro zjištění četnosti hovorů během pracovních dní a jejich porovnání.

3.2.2.4 Ukaž vše

Metoda, která zobrazí veškeré údaje v databázi.

3.2.2.5 Soukromé hovory

Metoda zobrazující všechny pobočky volající telefonní číslo se svazkem 6.

Tyto hovory jsou pobočce účtovány k náhradě.

3.2.2.6 Služební hovory

Metoda zobrazující všechny pobočky volající telefonní číslo se svazkem 0.

Svazek 0 se vytáčí u hovorů souvisejících s výkonem povolání.

3.2.2.7 Technické hovory

Metoda zobrazující modemy a hovory mající svazek roven hodnotě 1.

U této formy je nutno opět zařadit dříve odebraný svazek s hodnotou 1, jinak nebude funkční.

3.2.2.8 Faxy

Metoda zobrazující faxy a hovory mající hodnotu svazku rovnu 3.

U této formy je nutno opět zařadit dříve odebraný svazek s hodnotou 3, jinak nebude správně fungovat.

3.2.2.9 MultiLidi

Metoda ukazující všechny pobočky, kde není jasně stanovený uživatel, případně pobočky, kde se uživatelé střídají, jakými jsou například callcentra, bufety apod.

3.2.2.10 Délka celkem

Metoda, která sečítá všechny hovory za daný měsíc a vypíše je společně s celkovou cenou.

4 Testování

V této kapitole budou rozebrány praktické části data miningu telefonní ústředny.

4.1 Popis vzorku dat

```
1.12.2005 8:18|204|0|602259028|Eurotel GSM|ET-Biz 600|0:42|2,84
```

Zde je vidět vzorek dat, konkrétně tedy jeden hovor z databáze telefonních hovorů. Znak „|“ z kapacitních důvodů v této práci nahrazuje tabulátor, kterým jsou odděleny jednotlivé parametry v textovém souboru.

Prvním atributem je datum a čas. DataMining a Access tento formát načetly bez problémů jako tzv. „DateTime“, do programu SWI-Prolog musela být databáze upravena poněkud více.

Druhým parametrem je pobočka, tedy konkrétní telefonní přístroj, který má telefonní ústředna vedený pod číslem 204.

Třetí parametr je svazek. Tento parametr nabývá 5 hodnot (viz výše) a určuje charakter hovoru.

Čtvrtým parametrem je volané telefonní číslo.

Pátým parametrem je místo hovoru.

Šestým parametrem je operátor, který tento hovor spojoval.

Sedmým parametrem je délka hovoru.

Osmým parametrem je cena hovoru.

```
1.12.2005 8:18|204|0|602259028|Eurotel GSM|ET-Biz 600|0:42|2,84
```

Tento hovor je chápán takto: „Dne 1. 12. 2005 v 8 hodin a 18 minut, volala pobočka číslo 204, jako služební hovor, telefonní číslo 602 259 028. Tento hovor byl veden do sítě Eurotel GSM, operátor zaštiťující tento hovor byl ET - Biz 600. Hovor trval 42 sekund a cena hovoru byla 2,84 Kč.“

4.2 Zjištění

Výše zmíněné formy byly implementovány do nástrojů. Některé formy nebyly obtížné převést do požadovaného jazyka, případně syntaxe. Jiné na tom byly o poznání hůře a některé metody nebylo vůbec možné použít v daném prostředku. Důvody jsou různé, jedním z důvodů byla neznalost prostředků nutných k provedení dané formy u jiných obzvláště nekomerčních systémů byl problém s neodbornou, neúplnou a pouze částečnou dokumentací k programu. Zvláště tento poslední popsaný problém byl velkým břemenem pro vypracování této práce.

Co se týče tvoření jednotlivých programů, či jen používání jejich metod a funkcí dospěl jsem k následujícím poznatkům.

4.2.1 DataMining

Pro vývoj programu DataMining stačila pouze základní znalost jazyka C#. Program při spuštění načte databázi telefonních hovorů a následně s nimi pomocí metod, vytvořených podle forem, pracuje. Majoritní část metod je tvořena dvěma cykly do sebe vnořenými. Tyto cykly většinou porovnávají dva seznamy podle určitých podmínek, typických pro jednotlivé formy. Výsledek je zobrazován v „ListBoxu“, případně je možno ho uložit do textového souboru.

4.2.2 Microsoft Access 2003

Je program pro správu relačních databází. Program pracuje se zdrojovou tabulkou (databází) na kterou se dotazuje jednotlivými dotazy. Tyto dotazy jsou opět tvořené podle vzoru metod avšak způsob tvorby dotazů je u MS Access různorodější. Lze tu použít dotazování pomocí jazyka SQL, kdy je možno psát přímo dotazy podle syntaxe. Případně využít návrhového zobrazení, které využívá syntaxe jazyka Visual Basic. Obě tyto metody zadávání dotazů jsou mezi sebou převoditelné, což znamená, že lze dotaz

definovat pomocí jazyka SQL a také ho zobrazit pomocí návrhového zobrazení. Ovládání a zadávání dotazů v návrhovém zobrazení vyžadovalo určitě množství cviku a praxe, avšak postupem času se objevily zvláště dobré stránky tohoto způsobu dolování dat, jakými jsou zejména jednoduchost a logické zadávání dotazů

4.2.3 Deklarativní jazyk Prolog

a prostředí SWI-Prolog není úplně nejlepším způsobem pro dolování znalostí z databází v telefonní ústředně. Prolog je využíván především v oboru umělé inteligence a strojového učení a proto není divu, že pro dotazování se do databáze není nejvhodnější. Jedním ze základních problémů bylo načtení databáze. SWI-Prolog uznává pouze vlastní soubory s příponou „.pl“ a proto zde nebylo možné pracovat přímo s databází. Bylo nutné upravit databázi tak, aby nejvíce vyhovovala prostředí logického programování.

Podstata logického programování je hlavně v zadávání cílů a postup jakým se výsledku program dobere již není důležitý. Jazyk obsahuje několik základních metod a proto se zde vše musí deklarovat přesně a výstižně, většina metod a postupů se musí napsat ručně. Syntaxe jazyka není složitá, avšak o jednoduchém programování zde řeč být také nemůže. Jako příklad určitého úskalí je uvedena metoda „Obědáři“. Kdy se vybírají hovory poboček mezi 11 a 12 hodinou a tyto hovory musí být uskutečněny alespoň 5× za měsíc. Největším problémem jak prologu vysvětlit byl násobek 5×. Byl vyzkoušen způsob nejjednodušší, avšak neefektivní. A to napsat predikát 5× za sebou, čímž bych docílil požadovaného efektu. Avšak výsledku nebylo dosaženo. Již při třetím procházení klauzulí, vyhodnocení trvalo 3 minuty. U pátého program přestal odpovídat, proto bylo od této metody pomocí deklarativního programování upuštěno. Podobnou metodou jsou „Kontroloři“, kde daný výskyt musí být větší než 10×. Jiné problémy se týkaly řazení, nebylo možné

napsat metodu, která by data řadila ať již vzestupně nebo sestupně. U jiných metod již takové závažné problémy zjištěny nebyly.

4.2.4 Nekomerční program

Tato část je čistě subjektivní pocit autora, který je podložen špatnými zkušenostmi s těmito programy. Při úvodním pomyšlení na data mining pomocí těchto programů byla cítit určitá radost a uspokojení. Radost z toho, že se do programu nahraje databáze a po krátkém zpracování program vyhodnotí a ukáže výsledky, uspokojení z ušetřené práce. Avšak opak byl krutou pravdou. Jak bylo později zjištěno, tyto programy nefungují na způsobu automatu (zde vložit a zde odebrat výsledek), ale na principu použití a nastavení určitých typů analýz věnujících se danému typu problematiky (shluková analýza, rozhodovací pravidla apod.). Jinými slovy řečeno, tyto programy mají v sobě zabudované jádro, které je schopno vykonávat kýžené operace. Avšak nastavení parametrů, programu, nalezení správného způsobu dolování, spojení databáze s programem a další neméně důležitá nastavení musí uživatel udělat sám. Pokud se již povedlo načíst databázi do programu, pak byl obrovský problém jak s touto databází pracovat. Každý program funguje na jiném principu, má jiné uživatelské rozhraní (grafické, textové), používá jiné způsoby dolování a co je nejhorší ani jeden program neměl kvalitní dokumentaci (v porovnání s jazykem C# a Visual Studiem 2008), podle které by bylo možno pochopit fungování programu.

5 Vyhodnocení

Cílem práce bylo seznámení a praktické vyzkoušení technologii dolování dat. Výsledky a výstupy z jednotlivých metod lze najít a samozřejmě si i vyzkoušet na přiloženém CD. Počty výskytů a názvy poboček splňujících parametry jednotlivých forem nejsou pro tuto práci důležité (významné jsou spíše pro firmu) a proto nejsou v práci záměrně uvedeny (na CD jsou vypsané v souboru „pocetVysledku.xls“ ve složce „Vysledky“).

Cíle práce byly splněny za použití následujících prostředků:

5.1 Formy

Bylo vymyšleno celkem 20 druhů lidských vztahů, povah a charakteristik. Hlavních formy (celkem 10) značí opravdové lidské vztahy a povahy, vedlejší formy jsou spíše pomocné a slouží k pochopení případně upřesnění informací k hlavním formám. Některé formy byly vymyšleny ještě před vznikem práce, některé z nich byly navrženy vedoucím práce jako vzory, jiné vznikaly v průběhu vzniku práce. Formy vztahů jsou zajímavé jak z pohledu firmy, tak z pohledu personálního oddělení, které zkoumá schopnosti svých kolegů, případně podřízených. Některé formy poukazují na nekalé praktiky některých zaměstnanců, jiné poukazují spíše na pozitivní schopnosti.

Počet úkazů vyskytujících se u jednotlivých forem není důležitý. Podstatné je zjištění, že u každé formy byla nalezena alespoň jedna pobočka hodící se na parametry formy. Tudíž hypotézy, podle kterých byly formy navrženy byly správné a tím pádem byl data mining v telefonní ústředně z toho pohledu úspěšný.

5.2 Nástroje

Hlavním cílem bylo sestavit nástroje, odpovídající formám vztahů. Postup byl tedy takový, že pro jednotlivé formy byly vytvořeny metody, které byly

implementovány do programů. Programy pak testovaly jednotlivé metody na databázi a jako výstup většinou posloužil výpis poboček splňujících dané podmínky.

Byly použity tři hlavní nástroje. Každý nástroj má své klady a zápory, avšak z pohledu autora se jako nejužitečnější pro danou problematiku jevily jazyky C# (DataMining) a SQL (MS Access), na třetím místě se umístil deklarativní jazyk Prolog. Možné příčiny úspěchu a neúspěchu jednotlivých nástrojů mohou být ve zkušenostech s používáním dílčích jazyků. Jazykům C# a SQL bylo věnováno několik semestrů ve školní výuce. Oproti tomu Prologu byl věnován jen jeden semestr a Prolog není úplně běžný prvek, se kterým se lze setkat. Další příčinou může být fakt, že Prolog je využíván zejména v oboru umělé inteligence, kdyžto C# je univerzálním programovacím jazykem.

5.3 Použitelnost

Všechny použité nástroje jsou použitelné i na databázích z jiných firem. Avšak je tu podmínka, že dané databáze musejí mít aplikacemi použitelný formát, jinými slovy stejný formát jako zkoumaná databáze (pokud má program správně a plně fungovat).

5.4 Doporučení pro snížení nákladů

Možná opatření mohou vést k určitému psychickému tlaku na zaměstnance a proto by se mělo postupovat velice obezřetně.

5.4.1 Selekce poboček

Databáze obsahuje 173 poboček, tyto pobočky by bylo vhodné rozdělit do menších skupin. Tyto skupiny nechť jsou nazvány úseky. Počet poboček v úseku se obtížně stanoví, proto je mnohem užitečnější rozdělit pobočky do úseků podle zařazení. Pobočky jako jsou sekretářky, právní oddělení,

ředitel, personalisté apod. zařadit do úseku nazvaného příkladně „vedení“. Pobočky zabývající se ekonomikou jako například účetní, ekonomové, fakturační oddělení apod. zařadit do „ekonomického úseku“ a podobným způsobem pokračovat i s ostatními pobočkami.

Ke každému úseku zvolit vedoucího, který bude za příslušný úsek zodpovídat a kterému budou chodit faktury za telefonní hovory pro příslušné oddělení. Většinou již podobné úseky vedoucího mají a tak je příhodné funkci „kontrolora telefonních hovorů“ přidělit již určenému vedoucímu oddělení.

Faktury společně s výpisy budou chodit na příslušný počet poboček v daném úseku. Faktura obsahuje jak celkovou částku, provolanou za stanovené období (většinou měsíc), tak podrobný výpis hovorů ke každé pobočce jednotlivě.

Náměstek sleduje daný úsek z hlediska nákladovosti a do nákladů se samozřejmě počítají i náklady za telefonní služby. Je pouze na náměstkovi, zda kontroluje provolané částky či ne. Může nastavit limity a omezení pro jednotlivé pobočky jak v podobě provolané částky, tak v podobě provolaného času. Důležité je nastavit správnou výši limitů, což vyžaduje zkušenosti a určitě množství experimentů.

Náměstek je ohodnocen prémiemi, které se odvíjí od bilancí hospodaření daného úseku.

5.4.2 Analýza

Poplatky za telekomunikační služby jsou pro firmu náklad, opakující se pravidelně každý měsíc. Jejich snížení je tedy úsporou, která z dlouhodobého pohledu reprezentuje nepochybně velice zajímavou částku.

5.4.2.1 Výběr operátora

Poskytovatelé hlasových služeb se v dnešním vysoce konkurenčním prostředí prezentují řadou akčních nabídek a cenových akcí – ty jsou mnohdy

podmíněny splněním minimální výše hovorného nebo uzavřením smlouvy na určité období, ve kterém není možné měnit tarif a také cena hovorného se v průběhu trvání smlouvy obvykle nesnižuje.

Faktické srovnání nabídek je problematické s ohledem na různý způsob tarifikace, různý objem a druh předplacených služeb v balíčku nebo zahrnutých v paušálu, případně kombinováním služeb pevných linek, mobilních služeb a internetového připojení. Cenové porovnání je pak pro běžného zákazníka bez patřičného „know how“ prakticky nemožné – přitom úspory v případě volby vhodného operátora a tarifů, často činí desítky procent. Je tedy vhodné obrátit se na odborníka, který se touto problematikou zabývá. Avšak pozor na odborníky, kteří jsou spíše zástupci jednotlivých operátorů a nesnaží se najít nejlepší řešení problému. Snaží se spíše upsat zákazníka k danému operátorovi, od kterého jsou za tyto služby placeni. Často se pak může stát, že nekvalifikovaný zprostředkovatel objedná služby, které nejsou od vybraného operátora v určitých oblastech k dispozici a pak se úspory nedostaví nebo se dostaví pouze v případě snížení kvality služeb.

Výsledkem kvalitní analýzy telefonních služeb nemusí být jen změna operátora, která z pohledu zákazníka může být spojena s organizačně technickými komplikacemi. Předložení analýzy stávajícímu operátorovi může být obvykle prostředkem k obdržení zvýhodněné konkurenceschopné nabídky. Je třeba si uvědomit, že v dnešní době již operátoři nemají za cíl získávat nové zákazníky (nových zákazníků je velice málo), jejich snaha je udržet si ty stávající a případně převést zákazníky od konkurenčních firem.

5.4.2.2 Výběr tarifu

Tarif za hovorné zahrnuje jednak cenu, nejčastěji vyčíslenou za minutu hovoru a také intervaly, podle kterých operátor účtuje. Někteří operátoři účtují první minutu celou a následný čas je účtován po vteřinách. To znamená

například, že deseti vteřinový hovor stojí jako hovor trvající celou minutu. Jiní operátoři účtují hovor první minutu celou a následně po půl vteřinách.

5.4.2.3 Oblast směru hovoru

Vesměs se operátoři dělí na dvě části. Jedna část jsou operátoři nerozlišující síť kam je hovor směřován. Druhá část jsou operátoři rozlišující sítě na domácí a cizí sítě. Hovory do domácích sítí jsou zvýhodněny oproti ostatním. S tím souvisí i volání z pevných linek na mobilní a naopak. Zvláště problematické a poměrně drahé je volání z pevných linek na mobilní. Využitím nových technologií a způsobů spojení takových druhů hovorů může ušetřit nemalé finanční prostředky. Takovou technologií může být například využití GSM bran. Ve stručnosti lze říct, že GSM brána umožní volání z pevného telefonu na mobilní telefon za tarif mobilního operátora. Ten je možné poté zvolit s lepší cenou, případně neomezený s voláním do všech sítí za paušální platbu. Tím, že volání přes GSM bránu prostřednictvím telefonní ústředny ve firmě využívají všichni pracovníci, je zaručeno dobré využití neomezených tarifů.

Paušální platbou se myslí poplatek, který reprezentuje stálou platbu za telekomunikační službu. Platba se platí pravidelně, nejčastěji měsíčně. Tato položka u různých operátorů představuje různé služby. Někdy je to prostá platba a platí se i když se v daném měsíci nevolá a někdy je to naopak částka, která kromě platby za službu již zahrnuje část, která se může použít na úhradu volání. Paušální platby bývají často zdrojem možných úspor. Platby je možné omezit zrušením neefektivní služby jako takové nebo jejím přeobjednáním na komplexní variantu, která umožní zachovat šíři potřebných služeb, ale značně omezit platbu.

5.4.2.4 Shrnutí

V dnešní době telefonní ústředny z části nahradili služby operátorů, kteří mají své velké telefonní ústředny. Následně s vyúčtováním jsou operátoři

schopni poskytnout podrobný výpis hovorů dané pobočky a tím téměř vyřadit nutnost aby každá firma měla vlastní telefonní ústřednu.

Vzhledem ke komplikovanosti problematiky volby vhodného operátora, tarifů a skladby služeb se doporučuje tuto oblast nepodceňovat a naopak se jí čas od času věnovat a využívané služby prověřit. Mnohdy je tato oblast zdrojem značných úspor a zbytečně zaplacených poplatků, což se mnohdy zjistí téměř náhodně, přitom optimalizací by se mohly získat nemalé částky.

Pro dosažení vysokých úspor při optimalizaci se rozhodně vyplatí spojit se při posuzování stavu s oborníkem nebo na operátorech nezávislou společností, která se touto problematikou zabývá a pomůže konkrétní situaci analyzovat.

6 Závěr

V této práci byl proveden data mining telefonní ústředny. Byly vytvořeny nástroje pomocí jistých programových komponent (programovacího jazyka C#, programu Microsoft Office Access 2003 a deklarativního programovacího jazyka Prologu). Tyto nástroje využívají metody, které jsou sestavené podle forem lidských vztahů, které odpovídají skutečným vlastnostem lidí.

Data miningem bylo zjištěno několik nekalých praktik, které zaměstnanci využívají poměrně běžně a záleží jen na firmě, zda tyto praktiky bude omezovat nebo je bude nečinně přehlížet. Samozřejmě jsou zde i formy, které nepoukazují na špatné vlastnosti či charakter zaměstnanců, ale ukazují v dobrém světle jejich schopnosti a povahu. Veškeré použité metody a techniky lze vyzkoušet i osobně, jsou obsaženy na přiloženém CD.

K dolování dat se v práci používá zejména clusteringu (shlukové analýzy) a určitého druhu selekce, kdy jsou vybírány pobočky daných vlastností. Práci bylo věnováno velké množství času avšak použité metody nejsou zcela univerzální, aby se daly použít i v jiných sektorech. Není vyloučené použití i na jiných databázích z jiných ústředen a jiných firem, ovšem podmínkou je stejný typ a stejná struktura databáze. Pouze u deklarativního jazyka Prologu je nutné upravit celou databázi do předdefinované podoby, aby jednotlivé metody pracovaly správně.

Data mining je velice mocný nástroj, pomocí kterého se dají zjistit zajímavé a podnětné informace o zkoumaných objektech. Data mining není jen jakýsi automat, do kterého se nahrají data a on je vyhodnotí. Je to soubor nástrojů a analýz a metod, které je nutno správně použít a nastavit aby fungovaly tak, jak je požadováno. Toto nastavení a použití je podmíněno zkušenostmi a znalostmi v oblasti data miningu a zejména v oblasti jednotlivých nástrojů. Existují pro data mining komerčně i nekomerčně distribuované nástroje, avšak není žádný univerzální, což potvrzuje složitost problematiky. Každý nástroj

se specializuje na určitý druh dolování dat, případně na více druhů, kde určitá univerzálnost je vykoupena poměrně obtížným obsluhováním. Navíc cena komerčních nástrojů je poměrně vysoká. Je nutno, na obranu komerčních systémů, podotknout, že k systémům jsou často pořádány určitá školení pro získání zkušeností a praxe s obsluhou systémů a navíc je tu možnost i technické podpory. Komerční systémy většinou propagují známé a v oblasti dolování dat zkušené firmy, jakou je například IBM. Nekomerčně distribuované systémy jsou většinou vědecké projekty, které nejsou tolik uživatelsky přívětivé jako bývají komerční systémy a navíc nemají zcela úplnou a vysvětlující dokumentaci a proto je poměrně obtížné se s nimi naučit smysluplně pracovat.

Použitá literatura

BERKA, Petr. *Dobývání znalostí z databází*. 1. vyd. Praha : Akademie věd České republiky, 2003. 366 s. ISBN 80-200-1062-9.

PETR, Pavel. *Data Mining : I. díl*. 1. vyd. Pardubice : Univerzita Pardubice, 2006. 144 s. ISBN 80-7194-886-1.

KOTÁSEK, Petr. *DMSL: the data mining specification language : short version of Ph.D. Thesis = DMSL: jazyk pro dolování z dat*. 1. vyd. Brno : Vysoké učení technické v Brně, 2003. 27 s. ISBN 80-214-2685-3.

POJER, Josef. *Statistické metody zpracování dat*. 1. vyd. Praha : Policejní akademie České republiky, 2001. 65 s. ISBN 80-7251-077-0.

HOLEŇA, Martin. *Statistické aspekty dobývání znalostí z dat*. 1. vyd. Praha : Univerzita Karlova, 2006. 106 s. ISBN 80-246-1186-4.

Fayyad U., Piatetsky-ShapiroG., Smyth P., Uthurusamy R. eds: *Advances in Knowledge Discovery and Data Mining.* AAAI Press/MIT Press, 1996.

Klosgen W., Zytkow J.: *Knowledge Discovery and Data Mining.* Tutorial Notes. PKDD'97, Trondheim 1997.

O2 [online]. 2009 [cit. 2010-03-15]. Informace o telefonních číslech. Dostupné z WWW: <http://www.cz.o2.com/osobni/sluzby-podle-abecedy/60332-informace_o_telefonnich_cislech.html>.

PLCHÚT, Martin. *Dobývání znalostí z databází* [online]. [s.l.] : [s.n.], 2008 [cit. 2010-03-15]. Úvod a oblasti aplikací, s. . Dostupné z WWW: < http://www.fit.vutbr.cz/study/courses/ZZD/public/seminar0304/Uvod.pdf>.

MSDN Academic Alliance Program [online]. 2008 [cit. 2010-03-15]. Dostupné z WWW: <http://www.microsoft.com/cze/education/licence/msdn_academic_alliance/default.mspx>.

Ipex.cz [online]. 2007 [cit. 2010-03-15]. 800 a další barevné linky. Dostupné z WWW: <http://www.ipex.cz/sluzby-a-produkty/hlas/800-a-dalsi-barevne-linky>.

Ustredny.cz [online]. 2006 [cit. 2010-03-15]. Jak ušetřit za hlasové služby. Dostupné z WWW: <http://www.ustredny.cz/scripts/index.php?id_nad=516>.

SWI-Prolog [online]. 2007 [cit. 2010-03-15]. Dostupné z WWW: <http://www.swi-prolog.org/ >.

Prolog (programovací jazyk) In *Wikipedia : the free encyclopedia* [online]. St. Petersburg (Florida) : Wikipedia Foundation, , [cit. 2010-04-15]. Dostupné z WWW: <http://cs.wikipedia.org/wiki/Prolog_(programovací_jazyk)>.

Microsoft Access In *Wikipedia : the free encyclopedia* [online]. St. Petersburg (Florida) : Wikipedia Foundation, , [cit. 2010-04-15]. Dostupné z WWW: <http://cs.wikipedia.org/wiki/Microsoft_Access>.

CRISP-DM : CRoss Industry Standard Process for Data Mining [online]. 2000 [cit. 2010-03-18]. Dostupné z WWW: <http://www.crisp-dm.org/index.htm>.

Software-matters.co.uk [online]. 2009 [cit. 2010-03-21]. Ms_access_logo.png. Dostupné z WWW: <http://www.software-matters.co.uk/images/ms_access_logo.png>.

Rejstřík

Rejstřík obrázků

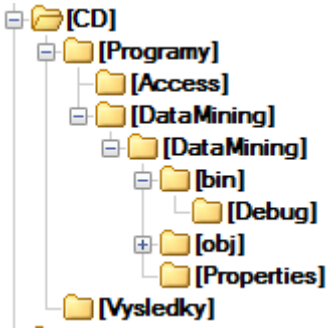
Seznam příloh

A) Zdrojové kódy k jednotlivým formám

Tyto kódy jsou poskytnuty jako ukázka porovnání kódu mezi jazyky C# a SQL. Případné porovnání rozdílů je ponecháno na subjektivním pocitu čtenáře.

B) Přiložené CD

Přiložené CD obsahuje všechny použité programy (složka „Programy“), včetně zdrojových kódů a databází. Lze si jednotlivé metody a formy vyzkoušet. Ve složce „Vysledky“ se nachází výstupní textové soubory z programu DataMining a některé souhrnné tabulky. Na CD je také přiložena původní databáze, se kterou je čerpáno („databaze.txt“) a celý text práce („data_mining_v_telefonni_ustredne.pdf“).



Obrázek 3: Diagram složek na CD

Barevné linky

C#

```
//zobrazi barevne linky. Urcuje se podle predvolby
public void barevneLinky(List<Hovor> hovory)
{
    List<Hovor> result = new List<Hovor>();
    foreach (Hovor hvr in hovory)
    {
        try
        {
              if (hvr.VolaneCislo.ToString().Substring(0, 3).Equals("840")
              || hvr.VolaneCislo.ToString().Substring(0, 3).Equals("841")
              || hvr.VolaneCislo.ToString().Substring(0, 3).Equals("848")
              || hvr.VolaneCislo.ToString().Substring(0, 3).Equals("844")
              || hvr.VolaneCislo.ToString().Substring(0, 3).Equals("810")
              || hvr.VolaneCislo.ToString().Substring(0, 3).Equals("855")
              || hvr.VolaneCislo.ToString().Substring(0, 3).Equals("900")
              || hvr.VolaneCislo.ToString().Substring(0, 3).Equals("906")
              || hvr.VolaneCislo.ToString().Substring(0, 3).Equals("909")
              || hvr.VolaneCislo.ToString().Substring(0, 3).Equals("800"))
            {
                result.Add(hvr);
            }
        }
        catch (Exception e)
        {
            string mes = e.Message;
        }
    }
    vypis(result);
}
```

SQL

```
SELECT databaze.Pobočka, databaze.Svazek, databaze.[Volané číslo]
FROM databaze
GROUP BY databaze.Pobočka, databaze.Svazek, databaze.[Volané číslo]
HAVING (((databaze.Svazek)=0 Or (databaze.Svazek)=6 Or (databaze.Svazek)=2)
       AND ((databaze.[Volané číslo]) Like "840*"
       Or (databaze.[Volané číslo]) Like "841*"
       Or (databaze.[Volané číslo]) Like "800*"
       Or (databaze.[Volané číslo]) Like "848*"
       Or (databaze.[Volané číslo]) Like "844*"
       Or (databaze.[Volané číslo]) Like "810*"
       Or (databaze.[Volané číslo]) Like "855*"
       Or (databaze.[Volané číslo]) Like "900*"
       Or (databaze.[Volané číslo]) Like "906*"
       Or (databaze.[Volané číslo]) Like "909*"));
```

Kontroloři

C#

```
//pobocka vola opakovane nejake tel cislo, max 1 minutu
//vice jak 10× za mesic
 private void kontrolori(List<Hovor> hovory)
 {
     List<Hovor> casLimit = new List<Hovor>();
     foreach (Hovor hvrs in hovory)
     {
         if (hvrs.Delka<60)
             casLimit.Add(hvrs);
     }

     List<Hovor> result = new List<Hovor>();
     List<Hovor> result2 = new List<Hovor>();
     foreach (Hovor hvr in casLimit)
     {
         foreach (Hovor hvrs in casLimit)
         {
             if (!hvrs.Equals(hvr) && hvr.Pobocka.Equals(hvrs.Pobocka)
               && hvr.VolaneCislo.Equals(hvrs.VolaneCislo))
             {
                 if (!result.Contains(hvr))
                 {
                     result.Add(hvr);
                 }
             }
         }
     }
     foreach (Hovor hvr in casLimit)
     {
         Int64 cislo = hvr.VolaneCislo;
         List<Hovor> cisla;
         cisla = result.FindAll(delegate(Hovor hvr2)
         {
             return hvr2.VolaneCislo == cislo && hvr2.Pobocka == hvr.Pobocka;
         });
         if (cisla.Count > 10 && !result2.Contains(cisla[0]))
             result2.AddRange(cisla.ToArray());
     }
     result2.Sort();

     vypis(result2);
 }
```

SQL

```
//zde pozor na delku hovoru. MS Access nechape,ze delka je v minutach a
//sekundach ne v hodinach a minutach.
SELECT databaze.Pobočka, databaze.[Volané číslo],Count(databaze.[Volané číslo])
       AS kolikrat
FROM databaze
WHERE (((databaze.Svazek)<>1 And (databaze.Svazek)<>3)
       AND ((Hour([Délka]))<Hour(#12/30/1899 1:0:0#)))
GROUP BY databaze.Pobočka, databaze.[Volané číslo]
HAVING (((Count(databaze.[Volané číslo]))>10));
```

Kukaččí vejce

C#

```
//jedno cislo volane z vice pobocek, puvodni volani musi byt volane s 6
private void kukacciVejce(List<Hovor> hovory)
{
    List<Hovor> svazekSest = new List<Hovor>();
    List<Hovor> casLimit = new List<Hovor>();
    foreach (Hovor hvrs in hovory)
    {
            casLimit.Add(hvrs);
    }
    foreach (Hovor hvr in hovory)
    {
        if (hvr.Svazek == 6)
            svazekSest.Add(hvr);
    }

    List<Hovor> result = new List<Hovor>();
    foreach (Hovor hvrs in svazekSest)
    {
        foreach (Hovor hvr in casLimit)
        {
              if (!hvr.Pobocka.Equals(hvrs.Pobocka) &&
                  hvrs.VolaneCislo.Equals(hvr.VolaneCislo) &&
                  !hvr.Svazek.Equals(hvrs.Svazek))
            {
                if (!result.Contains(hvr))
                {
                    result.Add(hvr);
                }
                if (!result.Contains(hvrs))
                {
                    result.Add(hvrs);
                }
            }
        }
    }
    result.Sort(new seradCislo());

    vypis(result);
}
```

SQL

```
SELECT databaze.Pobočka, kuk01.[Volané číslo], kuk01.Pobočka
FROM databaze INNER JOIN kuk01 ON databaze.[Volané číslo]=kuk01.[Volané číslo]
WHERE (((databaze.Svazek)=0 Or (databaze.Svazek)=2))
GROUP BY databaze.Pobočka, kuk01.[Volané číslo], kuk01.Pobočka
HAVING (((kuk01.Pobočka)<>[databaze].[Pobočka]));
```

Mimo pracovní dobu

C#

```
// pobocky volajici po pracovni dobe (pracovni doba od 8 do 16 hod)
 private void mimoPracDobu(List<Hovor> hovory)
 {

     List<Hovor> casLimit = new List<Hovor>();
     foreach (Hovor hvrs in hovory)
     {
         if (hvrs.DatumCas.Hour > 15 || hvrs.DatumCas.Hour < 8)
             casLimit.Add(hvrs);
     }
     casLimit.Sort();

     vypis(casLimit);
 }
```

SQL

```
SELECT databaze.Pobočka
FROM databaze
WHERE (((Hour([Datum a cas]))>=Hour(#12/30/1899 16:0:0#)
       And (Hour([Datum a cas]))<=Hour(#12/30/1899 23:0:0#))
       AND ((databaze.Svazek)=2 Or (databaze.Svazek)=0 Or (databaze.Svazek)=6))
       Or (((Hour([Datum a Cas]))>=Hour(#12/30/1899#)
       And (Hour([Datum a cas]))<Hour(#12/30/1899 8:0:0#))
       AND ((databaze.Svazek)=2 Or (databaze.Svazek)=0
       Or (databaze.Svazek)=6))
GROUP BY databaze.Pobočka;
```

Nejpilnější pobočka

C#

```
//zobrazi pobocky serazene sestupne podle provolane delky
private List<Hovor> nejpilnejsiPobocka(List<Hovor> hovory)
{
    List<Hovor> casLimit = new List<Hovor>();
    foreach (Hovor hvrs in hovory)
    {
        casLimit.Add(hvrs);
    }
    casLimit.Sort();
    string poslPobocka = "";
    List<Hovor> result = new List<Hovor>();
    foreach (Hovor hvr in casLimit)
    {
        if (!poslPobocka.Equals(hvr.Pobocka))
        {
            if (!result.Contains(hvr))
            {
                result.Add(hvr);
            }
            poslPobocka = hvr.Pobocka;
        }
        else
        {
            Hovor hovor = result.Find(delegate(Hovor hv)
            {
                return hv.Pobocka == hvr.Pobocka;
            });
            result.Remove(hovor);
            hovor.Delka += hvr.Delka;
            hovor.Cena += hvr.Cena;
            result.Add(hovor);
        }
    }
    result.Sort(new seradDelka());
    result.Reverse();
    data = new Reader();
    return result;
}
```

SQL

```
SELECT databaze.Pobočka, Sum(databaze.Délka) AS celkem, Round(([celkem]*24)*60)
       AS minut
FROM databaze
WHERE (((databaze.Svazek)=0 Or (databaze.Svazek)=2 Or (databaze.Svazek)=6))
GROUP BY databaze.Pobočka
ORDER BY databaze.Pobočka, Sum(databaze.Délka) DESC;
```

Nejvýřečnější pobočka

C#

```
//nejvyrecnejsi pobocka, serazeno vzestupne
public void nejvyrecnejsiPobocka(List<Hovor> hovory)
{
    List<Hovor> result = new List<Hovor>();
    foreach (Hovor hvr in hovory)
    {
        try
        {
            if (hvr.VolaneCislo.Equals(1180))
            {
                result.Add(hvr);
            }
        }
        catch (Exception e)
        {
            string mes = e.Message;
        }
    }
    result.Sort(new seradDelka());

    vypis(result);
}
```

SQL

```
SELECT databaze.Pobočka, databaze.Délka
FROM databaze
GROUP BY databaze.Pobočka, databaze.Délka, databaze.[Volané číslo]
HAVING (((databaze.[Volané číslo])=1180))
ORDER BY databaze.Délka;
```

Obědáři

C#

```
//pobocka vola opakovane nejake tel cislo, mezi 11:00 a 12:00
//vice jak 5× za mesic
 private void obedari(List<Hovor> hovory)
 {

     List<Hovor> casLimit = new List<Hovor>();
     foreach (Hovor hvrs in hovory)
     {
        if hvrs.DatumCas.Hour >= 11 && hvrs.DatumCas.Hour <12)
           casLimit.Add(hvrs);
     }
     List<Hovor> result = new List<Hovor>();
     List<Hovor> result2 = new List<Hovor>();
     foreach (Hovor hvr in casLimit)
     {
         foreach (Hovor hvrs in casLimit)
         {
             if (
                      !hvrs.Equals(hvr)&&hvr.Pobocka.Equals(hvrs.Pobocka)
                      && hvr.VolaneCislo.Equals(hvrs.VolaneCislo))
             {
                 if (!result.Contains(hvr))
                 {
                     result.Add(hvr);
                 }
             }
         }
     }
     foreach (Hovor hvr in casLimit)
     {
         Int64 cislo = hvr.VolaneCislo;
         List<Hovor> cisla;
         cisla = result.FindAll(delegate(Hovor hvr2)
         {
             return hvr2.VolaneCislo == cislo&&hvr2.Pobocka==hvr.Pobocka;
         });
             if (cisla.Count > 5&&!result2.Contains(cisla[0]))
                 result2.AddRange(cisla.ToArray());
     }
     result2.Sort();

     vypis(result2);
 }
```

SQL

```
SELECT databaze.Pobočka, databaze.[Volané číslo],
       Count(databaze.[Volané číslo]) AS pocet
FROM databaze
GROUP BY databaze.Pobočka, databaze.Svazek, databaze.[Volané číslo],
       Hour([datum a cas])
HAVING (((databaze.Svazek)=0 Or (databaze.Svazek)=2 Or (databaze.Svazek)=6)
       AND ((Hour([datum a cas]))<Hour(#12/30/1899 12:0:0#)
       AND (Hour([datum a cas]))>=Hour(#12/30/1899 11:0:0#))
       AND ((Count(databaze.[Volané číslo]))>5))
ORDER BY databaze.Pobočka;
```

Soukromé/služební hovory

C#

```
//hleda se pobocka, ktera vola opakovane na jedno tel cislo,
//jednou se svazkem 0 jednou 6
 private void soukromeSluzebni(List<Hovor> hovory)
 {
     List<Hovor> svazekSest = new List<Hovor>();
     List<Hovor> result = new List<Hovor>();
     foreach (Hovor hvr in hovory)
     {
         if (hvr.Svazek == 6)
             svazekSest.Add(hvr);
     }
     foreach (Hovor hvrs in svazekSest)
     {
         foreach (Hovor hvr in hovory)
         {
             if (hvrs.Pobocka.Equals(hvr.Pobocka) &&
                 hvrs.VolaneCislo.Equals(hvr.VolaneCislo)
                 && !hvr.Svazek.Equals(hvrs.Svazek))
             {
                 if (!result.Contains(hvr))
                 {
                     result.Add(hvr);
                 }
                 if (!result.Contains(hvrs))
                 {
                     result.Add(hvrs);
                 }
             }
         }
     }
     result.Sort();

     vypis(result);
 }
```

SQL

```
SELECT [Soukrome hovory].Pobočka, [Soukrome hovory].[Volané číslo]
FROM [Sluzebni hovory] INNER JOIN [Soukrome hovory]
ON [Sluzebni hovory].[Volané číslo] = [Soukrome hovory].[Volané číslo]
GROUP BY [Soukrome hovory].Pobočka, [Soukrome hovory].[Volané číslo],
        [Sluzebni hovory].Pobočka
HAVING ((([Sluzebni hovory].Pobočka)=[Soukrome hovory].[Pobočka]))
ORDER BY [Soukrome hovory].Pobočka;
```

Začátek telefonního dne

C#

```
//prvni hovor v pracovnim dni
private void zacatekPracDne(List<Hovor> hovory)
{

    List<Hovor> casLimit = new List<Hovor>();
    foreach (Hovor hvrs in hovory)
    {
        casLimit.Add(hvrs);
    }
    casLimit.Sort(new seradDatum());
    DateTime poslDen = new DateTime();
    List<Hovor> result = new List<Hovor>();
    foreach (Hovor hvr in casLimit)
    {
        if (!poslDen.Date.Equals(hvr.DatumCas.Date))
        {
            if (!result.Contains(hvr))
            {
                result.Add(hvr);
            }
            poslDen = hvr.DatumCas;
        }
    }
    result.Sort(new seradDatum());

    vypis(result);
}
```

SQL

```
SELECT Day([Datum a cas]) AS den, First(databaze.[Datum a cas]) AS
       [FirstOfDatum a cas], IIf(Weekday([Datum a
       cas])=1,"ne",IIf(Weekday([Datum a cas])=2,"po",
       IIf(Weekday([Datum a cas])=3,"út",
       IIf(Weekday([Datum a cas])=4,"st",
       IIf(Weekday([Datum a cas])=5,"čt",
       IIf(Weekday([Datum a cas])=6,"pá","so"))))))
       AS DenVTydnu, First(databaze.Pobočka) AS FirstOfPobočka
FROM databaze
WHERE (((databaze.Svazek)=0 Or (databaze.Svazek)=2 Or (databaze.Svazek)=6))
GROUP BY Day([Datum a cas]), IIf(Weekday([Datum a cas])=1,"ne",
       IIf(Weekday([Datum a cas])=2,"po",
       IIf(Weekday([Datum a cas])=3,"út",
       IIf(Weekday([Datum a cas])=4,"st",
       IIf(Weekday([Datum a cas])=5,"čt",
       IIf(Weekday([Datum a cas])=6,"pá","so"))))))
ORDER BY Day([Datum a cas]), First(databaze.[Datum a cas]);
```

Zahraniční hovory

C#

```
//cislo delsi nez 9 znaku, obsahuje predvolbu 0042
// nebo zacina cislem 4
 private void zahranicniHovor(List<Hovor> hovory)
 {
     List<Hovor> result = new List<Hovor>();
     foreach (Hovor hvr in hovory)
     {
         if (hvr.VolaneCislo.ToString().Length > 9 &&
             !hvr.VolaneCislo.ToString().Substring(0,4).Equals("0042")
             && hvr.VolaneCislo.ToString().Substring(0,1).Equals("4"))
         {
             if (!result.Contains(hvr))
             {
                 result.Add(hvr);
             }
         }

     }
     result.Sort(new seradCislo());

     vypis(result);
}
```

SQL

```
SELECT databaze.Pobočka, databaze.Místo
FROM databaze
GROUP BY databaze.Pobočka, databaze.Místo, databaze.Svazek,
        databaze.[Volané číslo]
HAVING (((databaze.Místo) Like "Rak*" Or (databaze.Místo) Like "Něm*"
        Or (databaze.Místo) Like "Slov*" Or (databaze.Místo) Like "*sko"
        Or (databaze.Místo) Like "*cko" Or (databaze.Místo) Like "* republika")
        AND ((databaze.Svazek)=0 Or (databaze.Svazek)=2
        Or (databaze.Svazek)=6)) Or (((databaze.[Volané číslo]) Like "420*"))
ORDER BY databaze.Pobočka;
```