VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

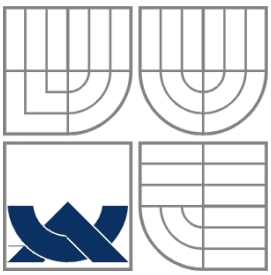
VYTVOŘENÍ MODULU PRO DOLOVÁNÍ DAT Z DATABÁZÍ

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

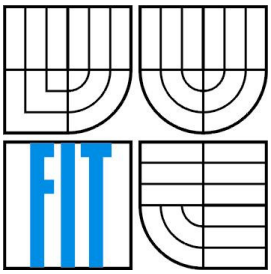
AUTOR PRÁCE
AUTHOR

Bc. DAVID KRÁSENSKÝ

BRNO 2008



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

VYTVOŘENÍ MODULU PRO DOLOVÁNÍ DAT Z DATABÁZÍ

CREATION OF UNIT FOR DATAMINING

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE
AUTHOR

Bc. DAVID KRÁSENSKÝ

VEDOUCÍ PRÁCE
SUPERVISOR

Ing. ROMAN LUKÁŠ, Ph.D.

BRNO 2008

Abstrakt

Cílem této práce je vytvořit modul pro získávání znalostí z databází pro informační systém Belinda firmy Voxnet s.r.o.

V první fázi budou pomocí programu SAS Enterprise Miner analyzována data z databáze klientů firmy pomocí několika dolovacích metod a výsledky budou porovnány. Ve druhé fázi bude vhodná metoda implementována jako modul informačního systému Belinda. Součástí práce je také zhodnocení dosažených výsledků a možného využití v praxi.

Klíčová slova

Získávání znalostí z databází, dolování z dat, regrese, neuronová síť, backpropagation, rozhodovací strom, SAS Enterprise Miner, predikce, klasifikace.

Abstract

The goal of this work is to create data mining module for information system Belinda.

Data from database of clients will be analyzed using SAS Enterprise Miner. Results acquired using several data mining methods will be compared. During the second phase selected data mining method will be implemented such as module of information system Belinda. The final part of this work is evaluation of acquired results and possibility of using this module.

Keywords

Knowledge Discovery in Databases, Data mining, regression, neural network, backpropagation, decision tree, SAS Enterprise Miner, prediction, classification.

Citace

Krásenský David: Vytvoření modulu pro dolování dat z databází. Brno, 2008, diplomová práce, FIT VUT v Brně.

Vytvoření modulu pro dolování dat z databází

Prohlášení

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně pod vedením Ing. Romana Lukáše,Ph.D.
Další informace mi poskytl Dušan Stloukal z firmy Voxnet s.r.o.
Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

……
Jméno Příjmení
Datum

Poděkování

Rád bych poděkoval ing. Romanovi Lukášovi, Ph.D. za odborné vedení a užitečné rady, které mi pomohly při řešení diplomové práce a tvorbě technické zprávy.

Dále také děkuji Dušanovi Stloukalovi a ing. Radkovi Škvařilovi za pomoc při implementaci a poskytnutí informací a dat firmy Voxnet s.r.o.

Obsah

1 Úvod

Získávání znalostí z dat je analytický proces získání informací z velkého objemu dat. Potřeba dolování v datech vznikla se vzrůstající velikostí databází a datových skladů a s objemem dat v nich uložených. Získávání znalostí má své uplatnění například ve vědeckém výzkumu, finančních analýzách a především v komerční sféře.

Cílem této práce je vytvořit nástroj pro podporu rozhodování. Tento nástroj bude implementován jako modul do existujícího webového informačního systému Belinda firmy Voxnet s.r.o. Vytvořený modul bude otestován na zvolených dolovacích úlohách. Dále budou analyzována data z databáze firmy. Analýza bude provedena v prostředí aplikace SAS Enterprise Miner pomocí několika dolovacích metod. Výsledky dosažené pomocí jednotlivých metod budou porovnány mezi sebou a také s výsledky, které budou získány pomocí implementovaného modulu. Součástí práce je také shrnutí a porovnání používaných dolovacích metod a algoritmů.

Kapitola nazvaná Získávání znalostí z databází pojednává o dané problematice, o praktickém využití a detailně popisuje celý proces získávání znalostí. Následující kapitola popisuje metody dolování a srovnání konkrétních algoritmů. V kapitole Informační systém Belinda je stručně popsán informační systém firmy Voxnet s.r.o., jehož součástí bude vytvořený modul pro dolování z dat. Kapitola pátá obsahuje výsledky dolovacích úloh provedených v prostředí aplikace SAS Enterprise Miner. V kapitole 6 je popsána implementace modulu včetně implementovaných algoritmů a výsledků dolovacích úloh provedených pomocí vytvořeného modulu.

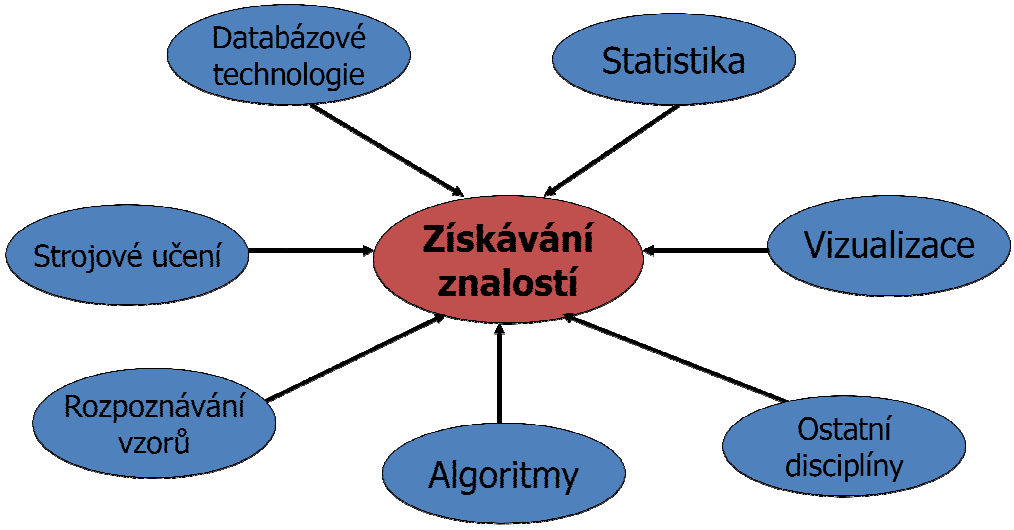
Tato diplomová práce navazuje na Semestrální projekt, v rámci kterého byly provedeny dolovací úlohy pomocí aplikace SAS Enterprise Miner.

2 Získávání znalostí z databází

Získávání znalostí z dat je analytická metodologie získávání netriviálních, skrytých, předem neznámých a potenciálně užitečných informací z velkého množství dat [HAN2006].

Netrivialitou je myšlena nutnost užití složitějšího algoritmu nebo postupu pro získání nějaké informace. V případě použití klasického dotazu do relační či jiné databáze se nejedná o dolování v datech. Skrytost informací znamená, že dané informace nejsou v databázi jasně patrné, je třeba je nějak získat.

V procesu získávání znalostí z databází se spojuje několik disciplín, jako jsou statistika, strojové učení, databáze, algoritmy, rozpoznávání vzorů, vizualizace a jiné, například neuronové sítě či vysoce náročné výpočty.



Obrázek 2.1: Spojení disciplín v získávání znalostí z dat [HAN2006]

V závislosti na typu dat nebo na dolovací aplikaci mohou být v systému pro dolování v datech použity také techniky z analýzy prostorových dat, obrázků, textů na webu či z oborů jako je obchod, bioinformatika či bankovnictví. O využití získávání znalostí dat pojednává kapitola Využití v praxi.

2.1 Historie získávání znalostí

Kořeny získávání znalostí z dat můžeme sledovat podél tří vývojových linií. Nejdelší z těchto linií je klasická statistika. Bez statistiky by získávání znalostí neexistovalo, jelikož na statistice je založeno mnoho technik využívaných při dolování. Klasická statistika zahrnuje koncepty jako například regresní analýza, standardní klasifikace, odchylka, rozptyl, shluková analýza a intervaly spolehlivosti, které jsou využívány při analýze dat a jejich vztahů.

Druhou linii tvoří umělá inteligence. Tato disciplína, která je založena na heuristice, se pokouší aplikovat postup lidského myšlení na statistické problémy. Jelikož tento přístup vyžaduje velký výpočetní výkon, nebylo jeho použití reálné až do počátku 80. let 20. století, kdy tehdejší počítače začaly nabízet odpovídající potřebný výkon za přijatelné ceny. Některé koncepty umělé inteligence byly využity v několika „high-end“ komerčních produktech, jako například dotazovací optimalizační modul pro Relational Database Management Systems (RDBMS).

Třetí vývojovou linií je strojové učení, které lze popsat jako spojení statistiky a umělé inteligence. Zatímco umělá inteligence neměla komerční úspěch, její techniky byly velkou měrou využity ve strojovém učení. Požitím metod strojového učení bylo vytvořeno více aplikací než pomocí umělé inteligence, protože bylo lépe využito zlepšujícího se poměru cena/výkon nabízených počítačů během 80. a 90. let 20. století a také nižších vstupních nákladů při tvorbě aplikací využívajících strojového učení. Strojové učení by mohlo být považováno za evoluci umělé inteligence, jelikož se v něm prolínají heuristika umělé inteligence s pokročilou statistickou analýzou. Cílem strojového učení je nechat počítačové programy naučit se o datech, která studují, provést různá rozhodnutí založená na kvalitě studovaných dat s využitím statistiky a s přidání pokročilé heuristiky a algoritmů umělé inteligence dosáhnout požadovaného cíle či výsledku.

Získávání znalostí v datech je v podstatě adaptací technik strojového učení v byznys aplikacích. Získávání znalostí je také často popisováno jako spojení historického a současného vývoje ve statistice, umělé inteligenci a strojovém učení. Tyto techniky jsou potom společně použity ke studování dat a k nalezení předem neznámých (skrytých) vzorů. V současnosti je získávání znalostí stále více využíváno v oblastech vědy a obchodu, kde je třeba analyzovat velká množství dat ke zjištění informací, které nebylo možné jinak získat.

2.2 Využití v praxi

Jak bylo uvedeno v předchozí kapitole, v současné době je získávání znalostí z databází často využívaným procesem pro získávání informací. V této kapitole bude uvedeno několik oborů, kde je získávání znalostí často využíváno.

Jedním z oborů je finančnictví a bankovnictví. Finanční instituce neustále řeší problém, jak mezi svými klienty zjistit, o kterou další službu by mohli mít zájem a jak jim ji nabídnout. Podobně i obchodníci potřebují analýzy, ze kterých by zjistili, co si trh žádá, jak přilákat nové zákazníky a ty stávající si udržet. Pojišťovny zase například potřebují snížit úroveň rizika, například omezit pojistné podvody. [SSCR] Dále lze získat například i informace o bankovních podvodech, monitorování transakcí provedených pomocí kreditních karet, hledáním abnormalit v transakcích apod.

Získávání znalostí z databází má své využití také v bioinformatice, což je vědní obor na pomezí biologie a informatiky, která se zabývá zpracováním, prohledáváním a analýzou dat o sekvenci, struktuře a popřípadě i funkci biologických makromolekul, tedy hlavně DNA a proteinů. [BUSSW]

S rozvojem molekulární biologie a genomického výzkumu došlo také k získání velkého množství dat, které obsahuje zajímavé informace, které je třeba získat.

Další zajímavou oblastí je získávání znalostí v textu a na webu. Během rozvoje internetu a používání elektronických médií vzniklo (a stále vzniká) velké množství textových dat jakou jsou webové stránky, novinové články, e-maily apod. Se zvětšujícím objemem dat se klasické vyhledávací metody stávají neefektivními. Je třeba vytvořit nástroje, které umožní porovnávat různé dokumenty, hodnotit kvalitu a důležitost dokumentů nebo nalézt v textech různé trendy a vzory. Častým příkladem využití dolování z dat je hledání spamu v elektronické poště.

Získávání znalostí je ovšem nejčastěji využíváno v komerční sféře a tento způsob využití je zajímavý i z hlediska této práce. Získané informace využívají například vedoucí pracovníci při strategickém a taktickém rozhodování nebo provozní manažeři zodpovědní za snižování nákladů. Strategičtí manažeři často využívají proces získávání znalostí pro porovnávání s konkurencí, při hledání příležitostí na trhu nebo při rozhodnutí o vypuštění nového produktu. Manažeři zodpovědní za taktická rozhodnutí využívají obdobné nástroje pro různé předpovědi prodeje, přímý marketing, získání zákazníků nebo pro analýzy marketingových kampaní.

Často používanou a velice známou metodou je analýza nákupního košíku. Cílem úlohy je zjistit informace o nákupech a produktech, konkrétně jaké produkty zákazníci nakupují společně, a najít asociační pravidla, která vyjadřují nějaké získané znalosti. Například při analýze můžeme dospět k závěru, že pokud si zákazník koupí máslo, často si při stejném nákupu koupí i pečivo. Využití analýzy nákupního košíku v praxi z marketingového hlediska publikoval Jaroslav Kuchař z firmy INCOMA Consult v roce 2002 na serveru www.marketingovenoviny.cz:

Vzhledem k tomu, že Česká republika patří z pohledu maloobchodního trhu mezi nejkonkurenčnější oblasti v Evropě, je třeba zvýšit průměrnou realizovanou marži prostřednictvím optimalizace akční (letákové) nabídky tak, že se sníží počet nakoupených položek v akci v průměrném košíku bez negativních důsledků na obrat. Pokladní systémy obchodních firem generují vysoce přesné a hodnotné informace, a to jak z provozního, tak z marketingového pohledu. Zaznamenávají každou nakoupenou položku s aktuální cenou (s možností přiřadit či dopočítat aktuální marži a aktuální slevu) a skutečnost, co s čím se kupuje – tedy jednotlivé nákupní košíky – a tím přesně zobrazují reálné nákupní chování. Pokud je k dispozici ještě identifikace nakupujícího k nákupnímu košíku, je získatelná marketingová informace téměř dokonalá.

V případě použití optimalizace promocí je třeba dále vyhodnocovat následující parametry: seznam položek a jejich přiřazení do skupin výrobků a kategorií, celkový počet položek a kusů zboží v nákupním košíku, označení položek v akci, dosažená marže v procentech na nákupním košíku a snížení marže v procentech vlivem letákových položek.

Při dostatku kvalitních dat lze dojít k značným rozdílům. Některé košíky obsahují málo akčních položek nebo žádné. Mnoho dalších košíků obsahuje vysoký podíl akčních. Předmětem zájmu jsou však košíky, které splňují současně tato kritéria:

a) mají charakter běžného či velkého nákupu (nezajímají nás drobné nákupy)

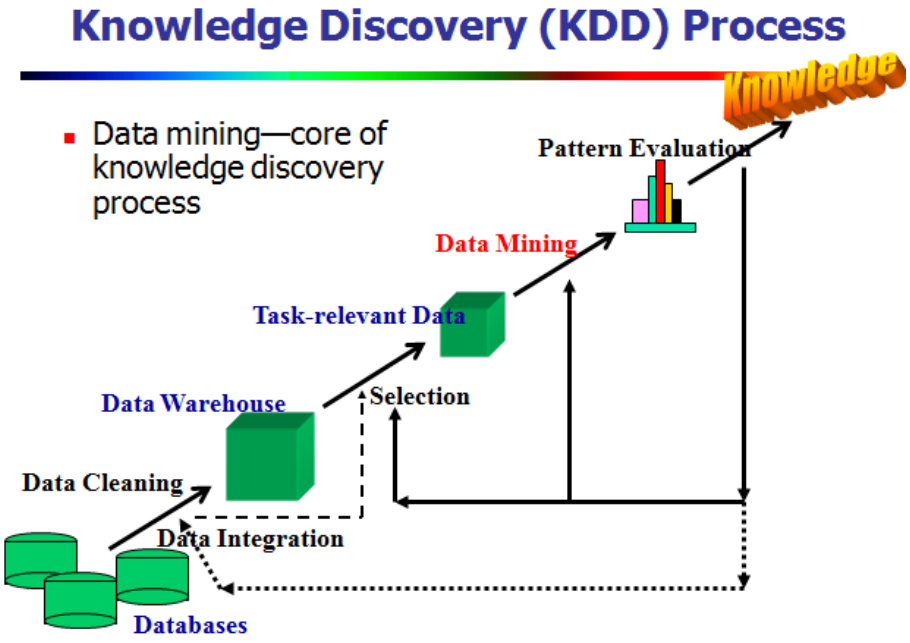
b) mají nadprůměrný výskyt letákových položek (nadprůměrné snížení marže)

U těchto košíků nás zajímá, které akční položky a jejich kategorie jsou často nakupovány společně. Příslušnému analytickému postupu se říká asociační analýza a jejím výsledkem jsou „soubory akčních položek“ (skupin výrobků, kategorií), které jsou často v jednom nákupním košíku. V rámci asociační analýzy je dále možné určit v jednotlivých souborech akčních položek jejich vnitřní hierarchie a tím odhalit závislosti mezi nimi.

Dále se potom vyberou výrobky, které zůstanou v akční nabídce a které ne (nebudou společně v jednom letáku). Při zavádění do praxe je třeba postupovat nanejvýš opatrně. Postupné snížení počtu položek v akční nabídce částečně odradí „tvrdé jádro“ nakupujících (nakupující orientující se výhradně na akční zboží) a na druhou stranu přiláká nakupující a jejich košíky s nadprůměrnými maržemi a bude mít tedy vliv na strukturu nákupních košíků a pozitivní efekt na dosahovanou marži. Je taková investice návratná? I kdyby byl systém využíván jen pro optimalizaci promocí (asi tak jedno promile možností využití), tak očekávaný přínos okolo jednoho procenta na maržích tuto investici vrací u velkých společností nejpozději do jednoho roku po začátku využívání. [SCART]

Jak je vidět na tomto příkladu z praxe, získávání znalostí z dat je velice silným nástrojem pro vedoucí pracovníky a marketingové odborníky a je také velmi často využíváno.

2.3 Proces získávání znalostí



Obrázek 2.2: Proces získávání znalostí [HAN2006]

Obrázek 2.2 ilustruje jednotlivé fáze procesu získávání znalostí. Jak je vidět, jednou z fází dolování je i dolování z dat (data mining). V tomto případě se jedná o jádro celého procesu. Pro proces získávání znalostí existuje více názvů, např. získávání znalostí z databází, dolování z dat, vytěžování dat, extrakce dat aj. V této práci bude používán termín získávání znalostí z dat (databází) pro označení celého procesu, dolováním z dat bude myšlena zmíněná hlavní fáze procesu.
Jak je patrné ze zmíněného obrázku, celý proces získávání znalostí se skládá z několika částí:

2.3.1 Čištění dat

Hrubá data, která jsou k dispozici pro proces získávání znalostí, jsou často v nevhodné formě pro samotné dolování a je třeba je upravit dle požadavků konkrétních metod tak, aby bylo možné dosáhnout co nejlepších výsledků.

Během této fáze dochází k čištění vzorku dat – odstranění odlehlých hodnot, doplnění chybějících hodnot, formátování či úprava nekonzistentních nebo zašuměných dat.

Fáze čištění dat může být považována za klíčovou, co se týká dosažení kvalitních výsledků. V případě, že vstupní data nebudou vhodně zpracována, mohou negativně ovlivnit výsledky celé analýzy.

2.3.1.1 Nejčastější úlohy čištění dat

Chybějící hodnoty

Může nastat (a v praxi také často nastává) situace, kdy ve vzorku dat některé hodnoty chybí. Tento problém je třeba řešit, jelikož chybějící hodnoty mohou ovlivnit výsledek. Některé metody si také s chybějícími hodnotami nedokáží poradit a může pak dojít k výraznému zkreslení výsledné analýzy. Pro řešení problému chybějících hodnot lze použít následující postupy:

1. Ignorování n-tice
   Nejjednodušší metoda, při které je n-tice obsahující chybějící hodnotu vynechána. Tato metoda není příliš efektivní, kromě případů, kdy v dané n-tici chybí ještě další data.

2. Ruční nahrazení chybějící hodnoty
   V případě, že uživatel, který doplňuje data, je znalý problematiky, může být tato metoda poměrně efektivní. Použitelnost této metody ovšem klesá se vzrůstajícím množstvím dat. Vzhledem k tomu, že v praxi se pracuje s velkým množstvím dat, je tato metoda nepoužitelná kvůli časové náročnosti.

3. Doplnění chybějící hodnoty globální konstantou
   Při použití této metody jsou všechny chybějící metody nahrazeny stejnou konstantou, například návěštím „unknown“. Tato metoda však není příliš spolehlivá a má také svá rizika. V případě většího množství chybějících hodnot jsou tyto také nahrazeny stejnou konstantou. Potom ovšem dolovací algoritmus může tuto hodnotu (konstantu) považovat za významnou a může dojít ke zkreslení výsledků analýzy.

4. Doplnění chybějící hodnoty průměrnou hodnotou
   Princip metody je velice jednoduchý. Každá chybějící hodnota je nahrazena průměrnou hodnotou získanou z ostatních n-tic.

5. Doplnění chybějící hodnoty průměrnou hodnotou ze všech vzorků patřících do stejné třídy jako dané n-tice
   Jedná se o rozšíření (a zefektivnění) předchozí metody. Pokud v dané n-tici chybí nějaká hodnota, pokusíme se podle daného klíče (jiný atribut n-tice než ten, který obsahuje prázdnou hodnotu) najít n-tice patřící do stejné třídy. Z těchto n-tic potom získáme průměrnou hodnotu a použijeme ji pro nahrazení chybějící hodnoty.

6. Doplnění chybějící hodnoty nejpravděpodobnější hodnotou

   Jedná se o sofistikovanější metodu než v předchozích případech. Chybějící hodnota je vypočítána pomocí složitějších metod, jako je například regrese nebo rozhodovací strom.

Šum v datech

Šumem jsou označovány chyby nebo odchylky v datech, které vznikají náhodně a mohou být způsobeny například chybou při sběru dat, exportu nebo lidským faktorem.
Postupy pro odstranění šumu:

1. Binning

   Princip této techniky spočívá ve vyhlazování dat, které jsou rozděleny do několika košů (anglicky bins – odtud název binning). Zašuměná data jsou nejprve seřazena a poté rozdělena tak, aby každý koš obsahoval přibližně stejný počet hodnot. Data v každém koši jsou pak vyhlazena.

   Pro vyhlazování se používá více metod:

   - Vyhlazení pomocí průměrné hodnoty – všechny hodnoty v koši jsou nahrazeny průměrnou hodnotou získanou z těchto hodnot
   - Vyhlazení pomocí mediánu – hodnoty v koši jsou nahrazeny mediánem (hodnota rozdělující množinu seřazených čísel na poloviny)
   - Vyhlazení pomocí hraniční hodnoty koše – za hraniční hodnoty jsou považovány minimální a maximální hodnota v koši. Každá hodnota v koši je poté nahrazena hraniční hodnotou, ke které má nejblíže.

2. Regrese

   Pro vyhlazování dat lze také použít metody regrese. Při zvolení lineární regrese se snažíme najít vhodnou regresní přímku tak, aby bylo možné atribut vyhladit za pomocí druhého. V případě, že je použito více atributů, je možné použít vícenásobnou lineární regresi.

3. Shlukování

   Metoda shlukování (clustering) se používá nalezení odlehlých hodnot. Podobná data jsou rozdělena do skupin (clusters) a zbytek dat, které leží mimo shluky jsou označena jako odlehlé hodnoty.

2.3.2 Integrace dat

Získávání znalostí z dat někdy vyžaduje proces integrace dat. Integrací je myšleno spojení dat z několika datových zdrojů (například datových skladů) do jednoho. Během integrace může dojít k různým problémům:

1. Problém identifikace entit

   Problém nastává například při slučování dat z více databází, kdy může dojít ke konfliktu metadat. Například dva sloupce z různých databází mají různý název, ale odkazují na stejný atribut.

2. Redundance dat

   Nadbytečná data mohou vzniknout například sloučením dvou databází, které obsahují stejné atributy. Je třeba ale také řešit redundanci dat, která nastane v případě, že z jedněch dat lze odvodit jiná data.

   Některé typy redundance lze odhalit pomocí korelační analýzy. Zda je mezi dvěma atributy závislost (popřípadě jak silná) a tedy zda na sobě závisí lze zjistit výpočtem korelačního koeficientu. Korelační koeficient může nabývat hodnot od −1 až po +1. Hodnota korelačního koeficientu −1 značí zcela nepřímou závislost, tedy čím více se zvětší hodnoty v první skupině znaků, tím více se zmenší hodnoty v druhé skupině znaků, např. vztah mezi uplynulým a zbývajícím časem. Hodnota korelačního koeficientu +1 značí zcela přímou závislost, např. vztah mezi rychlostí bicyklu a frekvencí otáček kola bicyklu. Pokud je korelační koeficient roven 0, pak mezi znaky není žádná statisticky zjistitelná závislost. [KORDEF]

3. Detekce a řešení konfliktu hodnot

   Tento problém nastává, pokud jsou pro popsání stejných entit reálného světa použity různé hodnoty. Klasickým případem může být popsání délky pomocí metrických a imperiálních jednotek, kdy výsledná délka je stejná, ale hodnoty jsou různé.

2.3.3 Transformace dat

Cílem transformace je uspořádání dat do podoby, která je vhodná pro fázi dolování dat. Během transformace můžeme použít následující procesy:

1. Vyhlazování

   Stejně, jako při odstraňování šumu v datech ve fázi čištění dat, lze použít binning, regresi nebo shlukování dat.

2. Agregace

   Spočívá ve spojení detailnějších údajů do větších celků pomocí agregačních operací. Tento postup je typicky využíván při konstrukci datové kostky.

   Jako příklad lze uvést spojování naměřených denních údajů do měsíčních, případně ročních součtů.

3. Generalizace

   Principem generalizace je nahrazení zdrojových dat na nižší úrovni koncepty z vyšší úrovně. Jako příklad lze uvést generalizaci atributu „ulice“ na atribut „město“.

4. Normalizace

   Během normalizace jsou vstupní data upravena tak, aby po úpravě spadala do předem daného intervalu hodnot, nejčastěji <-1.0,1.0> nebo <0.0,1.0>.

   Normalizace se často využívá při implementaci klasifikačních algoritmů jako jsou neuronové sítě nebo algoritmy založené na vzdálenosti, například metoda nejbližší soused. Normalizací vstupních hodnot lze urychlit fázi učení.

   Nejčastěji používané metody normalizace:

   - min-max normalizace – představuje lineární transformaci originálních dat. Předpokládejme, že minA a maxA jsou minimální, resp. maximální hodnoty atributu A. Min-max normalizace potom mapuje hodnotu *v* atributu A na novou hodnotu *v'* z nového rozsahu hodnot pomocí následujícího vztahu:

$$v' = \frac{v - minA}{\max A - minA}(new_{minA} - new_{maxA}) + new_{minA}$$

   - z-skóre normalizace – při použití z-skóre normalizace jsou hodnoty atributu A normalizovány na základě průměru a standardní odchylky A. Hodnota *v* atributu A je potom pomocí této normalizace mapována na novou hodnotu *v'* následujícím vztahem:

$$v' = \frac{v - \bar{A}}{\sigma_A}$$

   $\bar{A}$ a σ_A jsou průměr a směrodatná odchylka A.

- dekadické škálování – princip metody spočívá ve změně měřítka a posunu desetinné tečky. Výše zmíněné úpravy aplikujeme tak, aby celý rozsah hodnot bylo možné transformovat do daného rozsahu hodnot. Normalizace se provádí dosazením do vztahu

$$v' = \frac{v}{10^j},$$

kde *j* označuje nejmenší možné celé číslo takové, že platí max(|*v'*|) < 1.

2.3.4 Výběr dat

Cílem této fáze je vybrat data, která jsou pro řešení dané analytické úlohy relevantní. V praxi obsahují datové sklady velké množství dat. Komplexní analýza a dolování potom zabere velmi dlouhý čas, což je poněkud nepraktické a tedy nevhodné.

Redukce množství dat by měla proběhnout takovým způsobem, aby nebyla narušena integrita původního vzorku dat. Získávání znalostí na redukovaném vzorku dat by mělo přinést stejné nebo podobné výsledky jako při použití původních (neredukovaných) dat, avšak mělo by být rychlejší a efektivnější.

Používané procesy redukce dat:

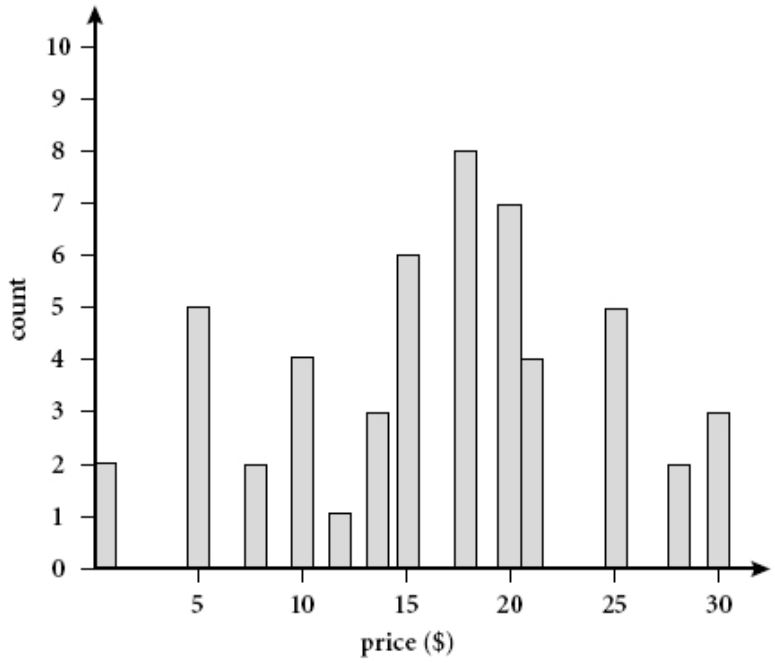
1. Agregace datové kostky
   Aplikací agregačních operací na data se zkonstruuje datová kostka.

2. Redukce dimenzí
   Cílem tohoto procesu je snížit velikost používaného vzorku dat. Data jsou zakódována nebo transformována tak, aby s nimi bylo možné pracovat stejným způsobem jako s původními daty. Redukci je také třeba provést tak, aby při rekonstrukci zakódovaných dat nedošlo ke ztrátě informací.

3. Redukce dat
   Původní data jsou nahrazena jinou, datově menší, reprezentací, například parametrickým modelem nebo neparametrickými metodami jako je shlukování, vzorkování nebo reprezentace dat pomocí histogramu.

Techniky používané pro redukci dat:

- Regrese a Log-lineární modely – příklad použití parametrických metod. Při použití lineární regrese jsou data reprezentována parametry regresní přímky. Pokud je třeba pracovat s více vstupními proměnnými, lze použít vícenásobnou lineární regresi.
  Mějme množinu n-tic v *n* rozměrech (např. popsány *n* atributy). Potom lze uvažovat každou n-tici jako bod v n-rozměrném prostoru. Log-lineární model může být použit k určení pravděpodobnosti každého bodu v multidimenzionálním prostoru pro množinu diskretizovaných atributů na základě podmnožiny kombinací rozměrů.[HAN2006]
  Obě uvedené metody lze použít i pro práci s řídkými daty.
- Histogramy – histogramy využívají binning (rozdělení dat do košů) a jsou často využívanou technikou redukce dat. Při konstrukci histogramu je důležité rozdělení dat do jednotlivých intervalů. Tuto problematiku lze řešit dle daných pravidel jako jsou rozdělení do šířky, rozdělení do hloubky, V-Optimal nebo MaxDiff.



Obrázek 2.3: Příklad histogramu [HAN2006]

- Shlukování – při použití techniky shlukování uvažujeme n-tice jako objekty. Tyto objekty jsou na základě podobnosti seskupovány do jednotlivých shluků. Podobností je myšlena vzdálenost jednotlivých objektů v prostoru.
- Vzorkování – technika vzorkování je pro redukci dat vhodná z toho důvodu, že umožňuje reprezentaci velkého množství dat pomocí malého vzorku nebo

podmnožiny dat. Předpokládejme velkou (původní) množinu dat *D* sestávající z *N* n-tic. Potom množinu *D* lze redukovat pomocí následujících metod:

- Jednoduchý náhodný vzorek bez vložení – vzorek o velikosti *s* vznikne vybráním počtu *s* n-tic z *N* n-tic z množiny *D* tak, že pravděpodobnost použití každé n-tice z *D* je *1/N*. To znamená, že šance výběru n-tic jsou stejné.
- Jednoduchý náhodný prvek s vložením – jedná se o modifikaci předchozí metody. Stejným způsobem dochází k výběru n-tice, po každém vybrání je ovšem daná n-tice vložena zpět do množiny *D* aby ji bylo možné znovu vybrat.
- Shlukový vzorek – n-tice z množiny *D* jsou seskupovány do *M* navzájem disjunktních shluků, z kterých je potom náhodně vybíráno *s* shluků tak, že platí $s < M$.
- Rozvrstvený vzorek – množina dat *D* je dělena na navzájem disjunktní části dat, které jsou nazývány strata. Z každého strata je pak vytvořen vzorek jednoduchým náhodným výběrem. Tento postup napomáhá vytvořit reprezentující vzorek dat, obzvláště platný je tento postup, pokud jsou data vychýlená.

4. Diskretizace a generování konceptuální hierarchie

   Během procesu diskretizace a generování konceptuální hierarchie jsou hodnoty atributů hrubých dat nahrazeny rozsahem nebo hodnotami na vyšší konceptuální úrovni. Diskretizace dat je forma redukce hodnot, která je velmi užitečná při automatickém generování konceptuální hierarchie. Oba uvedené procesy jsou velice účinné nástroje, které umožňují dolování v datech v různých úrovních abstrakce.

   Konceptuální hierarchie pro numerické atributy může být automaticky konstruována pomocí diskretizace dat. Je k tomu využíváno různých metod, například binning (rozdělení dat do košů), histogramová analýza, diskretizace založená na entropii, shluková analýza či diskretizace intuitivním rozdělením dat. Pro použití každé metody je však třeba, aby všechny hodnoty určené k diskretizaci byly vzestupně seřazeny.

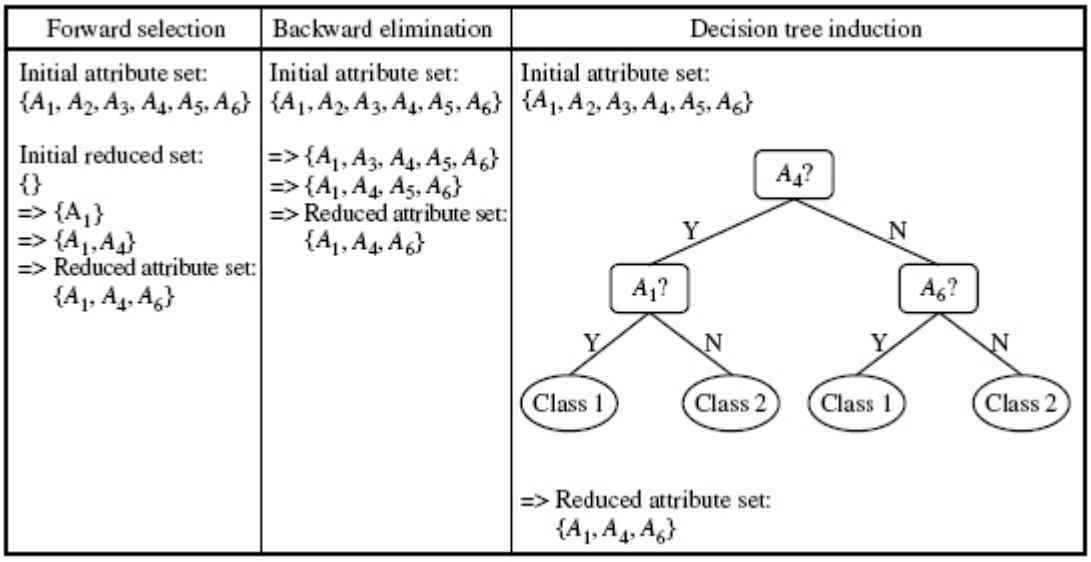
5. Výběr podmnožiny atributů

   Data určená pro analyzování mohou sestávat ze stovek atributů, z nichž mnohé mohou být pro danou úlohu nerelevantní nebo nadbytečné. Proto tyto nerelevantní, málo relevantní nebo redundantní atributy jsou vypuštěny a nejsou zahrnuty do množiny dat použité při dolování.

Cílem výběru podmnožiny atributů je nalézt takovou minimální množinu atributů, aby výsledky analýzy na těchto datech byly stejné nebo podobné jako při použití neredukovaných dat.

Základní heuristické metody pro výběr podmnožin atributů:

- Postupný výběr – procedura výběru začíná s prázdnou množinou redukovaných atributů. Nejvhodnější z atributů z původní množiny dat jsou vybrány a zařazeny do výsledné množiny redukovaných hodnot. V každém dalším kroku je vybrán a zařazen aktuální nejvhodnější atribut.
- Postupná zpětná eliminace – je opakem metody postupného výběru. Na počátku jsou všechny atributy zařazeny do výsledné množiny a v každém iteračním kroku je vyřazen aktuální nejméně vhodný atribut.
- Kombinace postupného výběru a postupné zpětné eliminace – v kombinaci obou výše uvedených metod dochází v každém iteračním kroku k výběru nejvhodnějšího atributu a zároveň k eliminaci nejméně vhodného atributu.
- Indukce rozhodovacího stromu – vhodné atributy jsou zařazeny do rozhodovacího stromu. Pro výběr atributů jsou používány algoritmy pro konstrukci rozhodovacích stromů, jako například ID3, C4.5 nebo CART.



Obrázek 2.4: Heuristické metody pro výběr podmnožin atributů [HAN2006]

2.3.5 Dolování dat

Jedná se o hlavní část celého procesu získávání znalostí. Pomocí různých metod a algoritmů jsou získána data (modely nebo vzory), která jsou poté analyzována.

Tato fáze je podrobněji probrána v kapitole 3 Procesy a algoritmy dolování dat.

2.3.6 Hodnocení modelů a vzorů

V této fázi jsou již dostupné nové informace, které se podařilo získat z původních dat pomocí některého z algoritmů a které je třeba analyzovat a vyhodnotit. Cílem je najít potencionálně zajímavé a užitečné modely či vzory. Může však nastat problém, že systém pro dolování dat najde velké množství modelů či vzorů. Následně je tedy nutné rozhodnout, které modely nebo vzory jsou pro danou úlohu zajímavé a mohou přinést požadovaný výsledek. V praxi je pouze malé množství modelů a vzorů vyhodnoceno jako potencionálně užitečné a tyto jsou předány uživateli.

Rozpoznání zajímavých modelů a vzorů je složitější a důležitý proces, je však možné uvést základní pravidla pro rozpoznání zajímavosti.

Vzor nebo model je zajímavý, pokud je snadno srozumitelný pro člověka, je potencionálně užitečný, nový a je validní na nových či testovacích datech s určitou mírou určitosti. [HAN2006]

2.3.7 Prezentace získaných znalostí

V poslední fázi procesu získávání znalostí jsou již rozpoznány zajímavé a užitečné vzory a je třeba vhodným způsobem prezentovat nově získané znalosti. Prezentované výsledky by měly být snadno srozumitelné a pochopitelné pro uživatele, je tedy třeba zvolit vhodnou formu vizualizace informací, například grafy apod.

3 Procesy a algoritmy dolování dat

3.1 Procesy dolování dat

Procesem je myšlena úloha prováděná během fáze dolování dat v průběhu získávání znalostí. Podle toho, jaký typ modelu či vzorů se snažíme získat, je třeba zvolit vhodný proces a také konkrétní algoritmus tak, aby bylo dosaženo co nejlepších výsledků.

Typy dolovacích úloh je možno rozdělit na klasifikaci, predikci, segmentaci a analýzu vztahů.

Klasifikace a predikce

Klasifikace je využívána pro určení diskrétních hodnot. Jedná se o proces probíhající ve dvou krocích, sestává z učící (trénovací) a aplikační fáze.

Během první fáze se zkonstruovaný model naučí klasifikovat data na určité množině trénovacích dat. Po naučení je vhodné model otestovat a případně upravit. Ve druhé fázi probíhá vlastní klasifikace, kdy naučený model přiřazuje data na vstupu do tříd.

Naopak predikce je využívána pro hodnoty spojité a během samotného procesu se snažíme předpovědět neznámou hodnotu. Nejčastější modely využívané při klasifikaci a predikci jsou model sestavený z klasifikačních pravidel, regrese, rozhodovací stromy či diskriminační analýza.

Otázkou však zůstává, jak zvolit vhodnou metodu. Při porovnávání klasifikačních metod je vhodné se zabývat následujícími vlastnostmi:

- Přesnost – tato vlastnost určuje, do jaké míry je metoda schopna třídit neznámá data do daných tříd. Obdobně je tomu tak u predikce, kdy přesnost určuje procento hodnot, které jsou správně predikovány z dříve neznámých dat.
- Rychlost – udává, za jakou dobu jsou vytvořena klasifikační pravidla a také jak rychle probíhá samotný proces klasifikace.
- Robustnost – jedná se o schopnost klasifikátoru či prediktoru provádět správné hodnoty, ikdyž jsou vstupní data zašuměná nebo obsahují chybějící hodnoty.
- Rozšiřitelnost – popisuje schopnost zkonstruovat klasifikátor nebo prediktor tak, aby efektivně pracoval i s větším množstvím dat.
- Interpretovatelnost – vlastnost určující jak je vytvořený model srozumitelný a pochopitelný. Tato vlastnost je však do určité míry subjektivní a není vždy jednoduché ji správně určit.

Segmentace

Segmentace nebo také shlukování bývá označováno jako učení bez učitele. Ve velkých databázích je často uloženo velké množství neohodnocených dat. Jejich přiřazení do nějakých tříd je poměrně složitý (a drahý) proces. V těchto případech jsou využívány právě shlukové metody, jejichž princip spočívá v hledání struktury neohodnocených dat. Jednotlivé objekty jsou rozřazovány do skupin (shluků) tak, aby objekty ve skupině byly co nejvíce podobné a zároveň, aby podobnost objektů z různých skupin byla co nejmenší. Podobnost objektů je nejčastěji řešena pomocí vzdálenosti. Existuje mnoho algoritmů používaných při shlukování, je možné je rozdělit do následujících skupin:

- Exkluzivní shlukování – v těchto metodách jsou objekty rozdělovány do tříd. Musí platit podmínky, že každý objekt patří jen do jedné skupiny a zároveň každá skupina obsahuje alespoň jeden objekt. Nejznámější algoritmy patřící do této skupiny jsou k-means a k-medoids.
- Shlukování s překryvem – liší se od exkluzivního shlukování tím, že dané objekty mohou patřit do více než jedné skupiny. Tyto skupiny bývají označovány jako fuzzy shluky, což znamená, že skupiny objektů nejsou striktně dané a objekty patří do skupiny s určitou pravděpodobností, která bývá označována jako stupeň příslušnosti. Příkladem takové algoritmu je například c-means.
- Hierarchické shlukování – princip metody spočívá v konstrukci hierarchie shluků, která bývá označována jako dendrogram. Metody konstrukce shluků jsou buď aglomerativní (menší shluky jsou spojovány do větších celků) nebo rozdělovací (větší celky jsou děleny na menší části).
- Pravděpodobnostní shlukování – předpokládejme, že data pocházejí z více zdrojů a je třeba nalézt jejich rozdělení váhy a to tak, aby data z jednoho zdroje tvořily jeden shluk.

Analýza vztahů

V analýze vztahů se snažíme o nalezení zajímavých vztahů (asociačních pravidel), tedy asociací a korelací, ve velkém množství dat. Typickým příkladem dolování asociačních pravidel je tzv. analýza nákupního košíku. Tento proces analyzuje zvyky nakupujících a hledá závislosti mezi různým zbožím, které si zákazníci vloží do svého nákupního košíku. Znalost informace o tom, které výrobky zákazníci obvykle kupují současně, mohou v praxi pomoci v tvorbě katalogů, pomáhají při definování strategie rozmístění zboží v prodejně atd. (podrobněji je princip analýzy nákupního košíku popsán v kapitole 2.2 Využití v praxi).[ASOC]

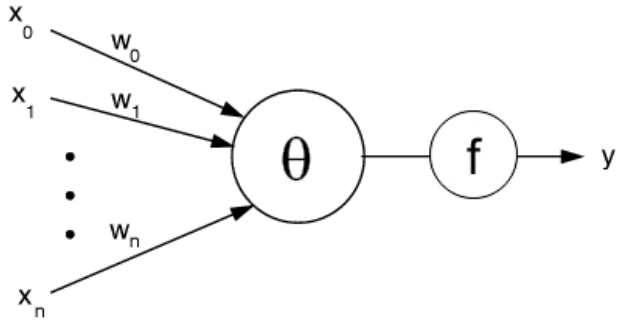
Jako příklad algoritmu lze uvést Apriori algoritmus nebo GUHA.

3.2 Algoritmy dolování dat

3.2.1 Predikční a klasifikační algoritmy

3.2.1.1 Neuronové sítě

Neuronové sítě fungují na principu biologických neuronů a jsou zpravidla tvořeny více propojenými neurony. Neuron je elementární jednotka pro zpracování informace. Na následujícím obrázku lze vidět model umělého neuronu. Každý je složen ze vstupních hodnot x_0, x_1... x_n, vah w_0, w_1 ...w_n a konstanty neuronu θ (tzv. bias).



Obrázek 3.1: Umělý neuron

Při dopředném šíření informace je spočtena suma $\sum_{i=0}^{n} w_i x_i + \theta$ a získaná hodnota je upravena aktivační funkcí *f*. Nově spočtená hodnota je výstupní hodnotou neuronu. Pokud je neuron součástí větší neuronové sítě, potom může být tato hodnota zároveň vstupní hodnotou neuronu v následující vrstvě. Učení neuronové sítě probíhá opakovanou modifikací hodnot vah a biasů.

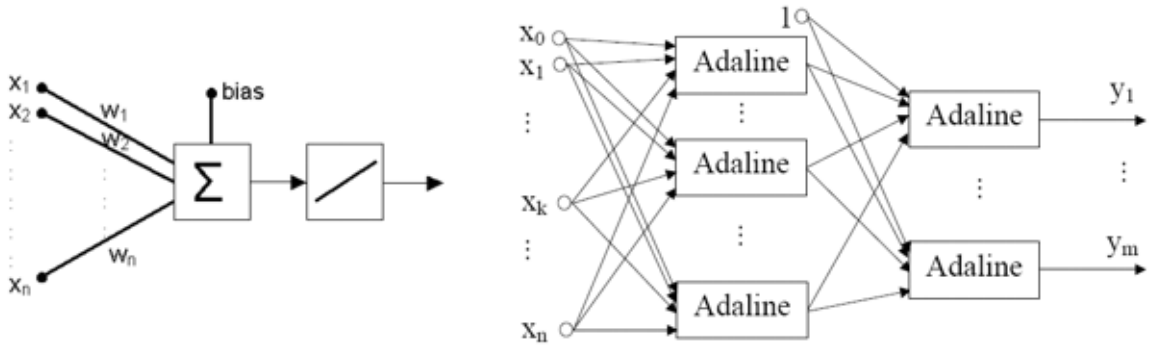
Perceptron

Jedná se o nejjednodušší typ postupné neuronové sítě a sestává pouze s jednoho neuronu. Perceptron je lineární klasifikátor, jenž dokáže vyhodnotit danou úlohu, pokud jsou třídy vzorů oddělitelné jednou hraniční přímkou. Z toho je patrné, že pomocí perceptronu lze nalézt hraniční přímku a pomocí ní klasifikovat data do dvou tříd.

Adaline, Madaline

Adaline (ADAptive LInear Element) je obdobou perceptronu a liší se tím, že neobsahuje aktivační funkci, resp. aktivační funkce je ve tvaru $f(x) = x$.[STECH] Hodnoty vstupů a výstupů tedy mohou nabývat libovolných hodnot. Výhodou tohoto modelu je jeho jednoduchost a snadná implementace, ovšem stejně jako perceptron jej nelze použít pro řešení složitějších úloh.

Na principu Adaline lze zkonstruovat dvouvrstvou neuronovou síť Madaline (Multiple Adaline), jenž sestává z více paralelně pracujících neuronů Adaline.



Obrázek 3.2: Schéma Adaline a Madaline [ADA]

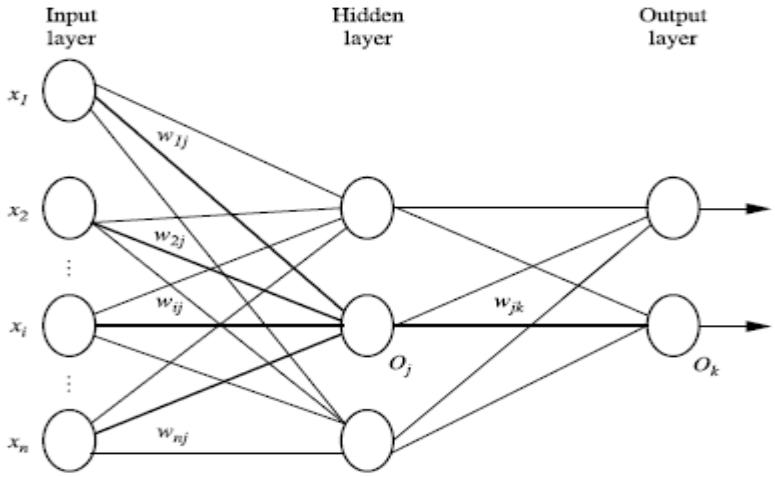
Backpropagation

Backpropagation je klasifikační učící se algoritmus. Neuronovou síť tvoří pole jednotlivých navzájem propojených neuronů uspořádaných do několika vrstev. První vrstva je vstupní – zde neurony pouze přebírají vstupní data a předávají je dále do skryté vrstvy. Skrytá vrstva nemusí být pouze jedna, neuronová síť jich může obsahovat libovolný počet. Ze skryté vrstvy jsou data předána do vrstvy poslední – výstupní, kde výstupní neurony zpravidla představují klasifikační třídy.

Před začátkem samotné klasifikace je třeba neuronovou síť naučit. Učení je iterativní proces, kdy v každém kroku učení jsou modifikovány váhy a biasy jednotlivých neuronů na základě zjištěné chyby. Chyba je zjišťována při každém kroku procesu učení a je šířena zpět od výstupní vrstvy ke vstupní.

Výhodami neuronové sítě je jejich tolerance dat obsahujících šum a také jejich schopnost správně klasifikovat data, na kterých nebyla trénována. Jsou také dobře použitelné při horší znalosti vztahů mezi atributy a třídami.

Na následujícím obrázku je znázorněno schéma neuronové sítě Backpropagation s jednou skrytou vrstvou.



Obrázek 3.3: Schéma neuronové sítě Backpropagation [HAN2006]

Neuronová síť Backpropagation byla implementována jako jeden z algoritmů dolovacího modulu, proto je podrobněji popsána v kapitole Implementované algoritmy.

3.2.1.2 Regrese

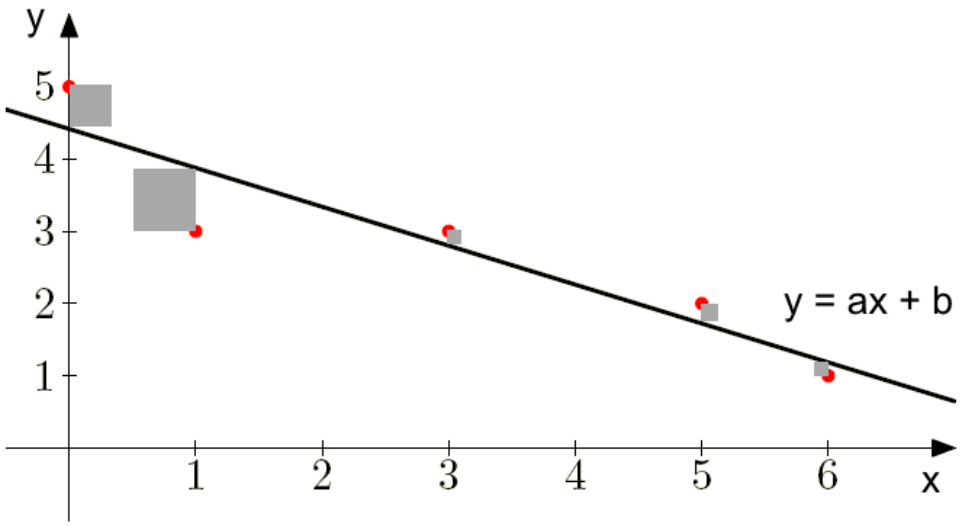
Regrese je nejznámější statistickou metodou pro doplňování hodnot dat spojitého charakteru, tedy predikce. Regresní analýza je označení metod, pomocí nichž odhadujeme hodnotu jisté náhodné veličiny (cílové proměnné, regresandu) na základě znalosti jiných veličin (regresorů).

Jako příklad využití regresní analýzy lze uvést předpověď počasí. Odhadujeme-li ráno, jaké bude přes den počasí (regresand) na základě znalosti předpovědi počasí a toho, jaké je venku počasí nyní (dva regresory).

Jednoduchá lineární regrese

Jednoduchá lineární regrese představuje nejjednodušší formu regrese, zahrnuje vstupní proměnnou x a výstupní proměnnou y. Princip metody spočívá v aproximaci daných hodnot polynomem prvního řádu (přímkou) $Y = aX + b$, kde Y je hodnota získaná dosazením vstupní hodnoty, a a b jsou regresní koeficienty. Jinak řečeno, jedná se o proložení bodů v grafu vhodnou přímkou.

Cílem je tedy nalézt vhodnou přímku pomocí určení regresních koeficientů a a b. Řešení tohoto problému lze provést metodou nejmenších čtverců. Princip metody nejmenších čtverců spočívá v nalezení nejvhodnější přímky takové, aby rozdíl mezi skutečnou hodnotou y a aproximovanou hodnotou Y byl co nejmenší. Výsledný rozdíl je určen jako součet druhých mocnin rozdílu jednotlivých hodnot Y_i a y_i. Druhé mocniny rozdílu souřadnic (obsahy čtverců) jsou znázorněny na následujícím obrázku šedými plochami.



Obrázek 3.4: Metoda nejmenších čtverců

Mějme tedy data ve formátu $(x_1,y_1), (x_2,y_2), ...,(x_n,y_n)$, kde n je počet prvků trénovací množiny dat. Potom regresní koeficienty a a b můžeme spočítat pomocí následujících vztahů:

$$a = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n}(x_i - \bar{x})^2},$$

$$b = \bar{y} - a\bar{x},$$

kde $\bar{x}$ a $\bar{y}$ jsou aritmetické průměry hodnot $x_1, x_2, ..., x_n$, resp. $y_1, y_2, ..., y_n$.

Vícenásobná lineární regrese

Jak bylo uvedeno, jednoduchou lineární regresi lze využít pro predikci hodnoty z jednoho atributu. V případě, že vstupních atributů je více, je třeba tuto metodu modifikovat na tzv. vícenásobnou lineární regresi.

Mějme množinu vstupních atributů $A_1, A_2, ..., A_n$ popsaných množinou n-tic X takových, že $X = (x_1, x_2, ..., x_2)$. Trénovací množina dat T obsahuje t n-tic ve tvaru $(x_{11}, x_{12}, \ldots, x_{1n}, y_1), (x_{21}, x_{22}, \ldots, x_{2n}, y_2), \ldots, (x_{t1}, x_{t2}, \ldots, x_{tn}, y_n)$. Data jsou potom aproximována pomocí rovnice:

$$Y = a_0 + a_1X_1 + a_2X_2 + ... + a_nX_n$$

Cílem úlohy je nalézt koeficienty rovnice $a_0, a_1, a_2, ..., a_n$. Princip výpočtu spočívá v úpravách matic X a Y a ve spočtení matice A, jež je tvořena právě hledanými koeficienty přímky. Matice X je sestavena tak, že první sloupec tvoří hodnoty 1, další sloupce tvoří hodnoty vstupních atributů. Matice Y obsahuje hodnoty výstupního atributu.

$$X = \begin{pmatrix} 1 & x_{11} & x_{12} & \ldots & x_{1n} \\ 1 & x_{21} & x_{22} & \ldots & x_{2n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_{t1} & x_{t2} & \ldots & x_{tn} \end{pmatrix} \qquad Y = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix} \qquad A = \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ \vdots \\ a_n \end{pmatrix}$$

Matice A je potom získána pomocí vzorce $A = (X^TX)^{-1}X^TY$, kde X^T je matice transponovaná a $(X^TX)^{-1}$ je inverzní matice vzniklá součinem matic X^T a X.

Kvadratická a polynomická regrese

V případě, že pomocí aproximace pomocí přímky nedosáhneme požadovaných výsledků, je možné soubor dat aproximovat kvadratickou funkcí (parabolou). Tento postup je nazýván kvadratická

regrese. Kvadratická regrese je zvláštním případem lineární regrese, stejně tak i polynomické regrese. Cílem je proložit skupinu *n* bodů $[x_i,y_i]$ polynomem druhého řádu $f(x) = ax^2 + bx + c$. [KR]

Jak bylo uvedeno, kvadratická regrese je zvláštním případem polynomické regrese. V případě polynomické regrese se jedná o proložení souboru dat obecným polynomem k-tého řádu. Existují dva způsoby řešení.[PR]

Mějme polynom $P_k = p_0 + p_1 x + \cdots + p_k x^k$, který musí mít koeficienty $p_0, \dots, p_k$ takové, aby součet *S* druhých mocnin odchylek e_i od původních hodnot *y* byl minimální. První způsob výpočtu je pomocí parciálních derivací, kdy je sestavena následující soustava pro aproximaci polynomem. Z ní lze potom vypočítat vektor hledaných koeficientů.

$$\begin{pmatrix} \sum x_i^{2k} & \dots & \sum x_i^{k+1} & \sum x_i^k \\ \vdots & \ddots & \vdots & \vdots \\ \sum x_i^{k+1} & \dots & \sum x_i^2 & \sum x_i \\ \sum x_i^k & \dots & \sum x_i & n \end{pmatrix} \cdot \begin{pmatrix} p_k \\ \vdots \\ p_i \\ p_0 \end{pmatrix} = \begin{pmatrix} \sum y_i x_i^k \\ \vdots \\ \sum y_i x_i \\ \sum y_i \end{pmatrix}$$

Druhý způsob spočívá v sestavení soustavy rovnic a vyřešení pomocí metody nejmenších čtverců. Soustava rovnic je zapsána v podobě matic.

$$\begin{pmatrix} 1 & x_1 & \dots & x_1^k \\ 1 & x_2 & \dots & x_2^k \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_n & \dots & x_n^k \end{pmatrix} \cdot \begin{pmatrix} p_0 \\ p_1 \\ \vdots \\ p_k \end{pmatrix} - \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix} = \begin{pmatrix} e_1 \\ e_2 \\ \vdots \\ e_n \end{pmatrix}$$

$$A.x - b = e$$

Cílem je vypočítat vektor koeficientů $[p_0, p_1, \dots, p_k]$ tak, aby velikost vektoru odchylek $[e_0, e, \dots, e_n]$ byla minimální. Koeficienty hledaného polynomu lze potom vypočítat podle vztahu

$$x = (A^T A)^{-1} A^T b,$$

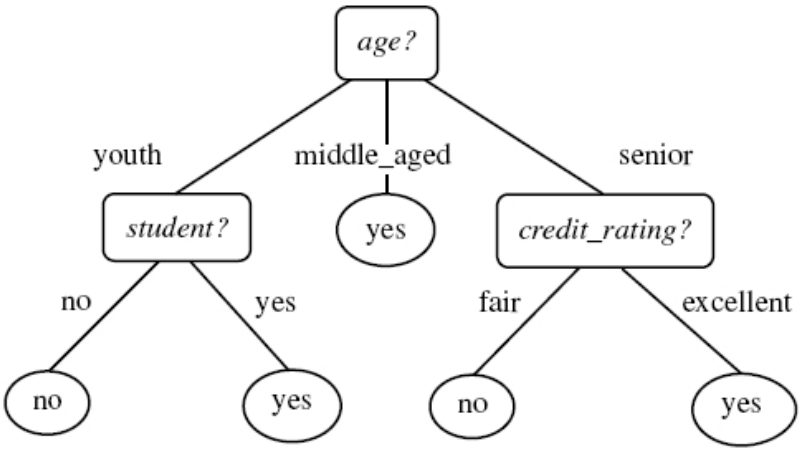
kde A^T značí matici transponovanou a horní index *-1* matici inverzní.

3.2.1.3 Rozhodovací strom

Rozhodovací strom je stromový graf, kde každý vnitřní uzel reprezentuje test hodnoty jistého atributu a listy grafu představují třídu, do které je daný objekt klasifikován. Rozhodovací strom může být poté převeden na odpovídající klasifikační pravidla. Každý uzel stromu představuje jednu (vybranou) vlastnost objektů, z tohoto uzlu vede konečný počet hran. Proto je nutné vlastnosti nejdříve diskretizovat. Problém může nastat při vytváření takového stromu. Ten musí co nejlépe objekty od sebe odlišit. Pro kořenový uzel se vybírá takový atribut, který objekty od sebe maximálně odliší.

Využívá se proto entropie (míra informační hodnoty atributu). Vytváření stromů je popsáno například v algoritmech ID3 a C4.5.

Rozhodovací stromy jsou vhodné pro úlohy, ve kterých má být provedena klasifikace nebo předpověď. Užitečné jsou v oblastech, ve kterých můžeme hodnoty proměnných rozdělit do relativně malého počtu skupin. Na druhou stranu nejsou vhodné pro případy, kdy je úkolem předpovězení kvantitativních hodnot.



Obrázek 3.5: Jednoduchý rozhodovací strom [HAN2006]

Sestavení klasifikačních pravidel

Převedení rozhodovacího stromu na seznam klasifikačních pravidel je poměrně triviální. Mějme jednoduchý rozhodovací strom znázorněný na obrázku 3.5. Klasifikační pravidla potom budou v následujícím tvaru:

if Age = youth **and** student = no **then** result = no

if Age = youth **and** student = yes **then** result = yes

if Age = middle_aged **then** result = yes

if Age = senior **and** credit_rating = fair **then** result = no

if Age = senior **and** credit_rating = excellent **then** result = yes

3.2.1.4 Bayesovská klasifikace

Bayesovský klasifikátor je statistický klasifikátor, který určuje, s jakou pravděpodobností daný prvek patří do každé klasifikační třídy. Vzorek tedy bude zařazen právě do té třídy, pro kterou je spočítána největší pravděpodobnost.

Bayesovská klasifikace je založena na Bayesově vzorci

$$P(X|Y) = \frac{P(Y|X)P(X)}{P(Y)}$$

P(X) a *P(Y)* jsou pravděpodobnosti, za kterých nastane jev *X*, resp. *Y*. *P(Y|X)* je takzvaná podmíněná pravděpodobnost jevu *Y* za podmínky výskytu jevu *X*.

Naivní bayesovská klasifikace

Naivní (někdy též jednoduchá) bayesovská klasifikace funguje následujícím způsobem:

1. Mějme množinu trénovacích dat *D*, kde každá n-tice je reprezentována vektorem $X = (x_1, x_2, \dots, x_n)$, kde *n* udává počet atributů $A_1, A_2, \dots, A_n$.

2. Dále předpokládejme *m* klasifikačních tříd $C_1, C_2, \dots, C_m$. Poté je hledáno takové i z *m*, pro které zvolený prvek (n-tice) *X* patří do daní třídy C_i s největší pravděpodobností. Platí tedy vztah

$$P(C_i|X) = \frac{P(X|C_i)P(C_i)}{P(X)}$$

   P(X) je v daném výrazu konstantou, hledejme tedy třídu C_i, pro kterou je výraz $P(X|C_i)P(C_i)$ maximální.

3. $P(C_i)$ udává pravděpodobnost, že daný prvek patří do klasifikační třídy C_i. Pravděpodobnost tedy lze určit jako $P(Ci) = \frac{s_i}{|S|}$, kde s_i udává počet prvků množiny *S,* který je klasifikován do třídy C_i a $|S|$ značí počet prvků množiny *S*.

4. $P(X|C_i)$ je pravděpodobnost libovolný prvek náležící do klasifikační třídy C_i bude mít stejné hodnoty atributů jako prvek *X*. Tuto pravděpodobnost lze určit pomocí vztahu

$$P(X|C_i) = \prod_{k=1}^{n} P(x_k|C_i)$$

5. Hodnoty $P(x_k|C_i)$ jsou potom určeny podle toho, zda atribut A_k má diskrétní či spojitý charakter.

   a. A_k má diskrétní charakter:

$$P(x_k|C_i) = \frac{s_{ik}}{s_i},$$

   kde s_{ik} je počet vzorků z množiny dat *S* zařazené do klasifikační třídy C_i, jejichž atribut A_k má hodnotu X_k.

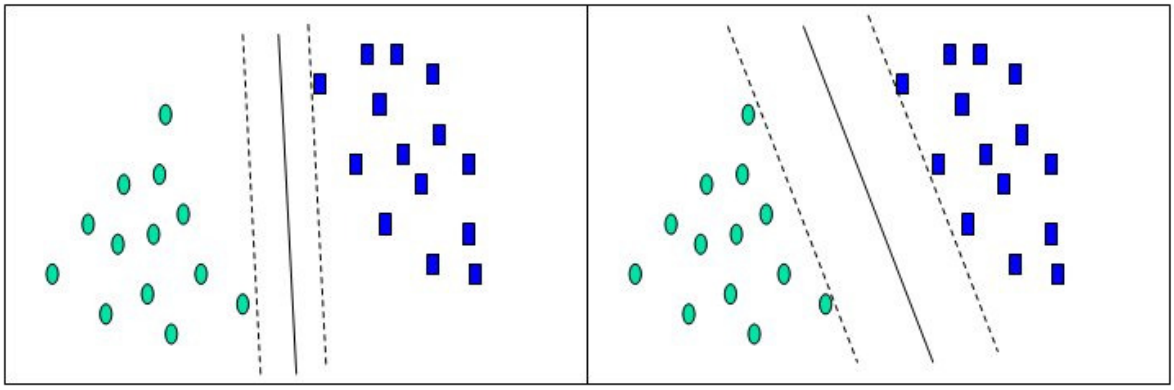
b. A_k má spojitý charakter:

$$P(x_k|C_i) = g(x_k, \mu_{C_i}, \sigma_{X_i}),$$

kde $g(x_k, \mu_{C_i}, \sigma_{X_i})$ je Gaussova normální funkce.

3.2.1.5 SVM (support vector machine)

SVM, neboli metoda podpůrných vektorů patří k poměrně novým metodám a je řazena k tzv. jádrovým algoritmům. SVM je efektivní algoritmus pro nalezení lineární hranice a zároveň umožňuje reprezentovat složité lineární funkce. Je ho tedy možné použít pro klasifikaci lineárních i nelineárních dat. Jedním ze základních principů je převod daného původního vstupního prostoru do jiného, vícedimensionálního, kde již lze od sebe oddělit třídy lineárně. Cílem je nalézt hranici mezi daty tak, aby mezi oddělenými daty vznikl co největší pás.



Obrázek 3.6: Klasifikace pomocí SVM ve 2D [SVM]

Otázkou zůstává jak nalézt optimální rozdělovač. Při hledání lineárního oddělovače se používá metoda kvadratického programování.

Mějme příklady x_i s klasifikací $y_i = \mp 1$ a cílem je nalézt právě optimální rozdělovač. Problém lze převést na hledání parametrů α, které maximalizují výraz

$$\sum_i \alpha_i - \frac{1}{2}\sum_{i,j} \alpha_i \alpha_j y_i y_j (x_i x_j)$$

za platných omezení $\alpha_i \geq 0$ a $\sum_i \alpha_i y_i = 0$. Po vypočítání parametrů α_i je hledaný oddělovač dán rovnicí

$$h(x) = sign(\sum_i \alpha_i y_i (x.x_i)).$$

V případě, že lineární oddělovač je hledán ve vícerozměrném prostoru *F(x)*, je člen $x_i.x_j$ nahrazen členem $F(x_i).F(x_j)$.

3.2.1.6 Klasifikace založená na vzdálenosti

Jako nejznámější algoritmus klasifikace založené na vzdálenosti lze uvést algoritmus založený k-nejbližším sousedství (k-nearest neighbor). Princip metody je následující.

Trénovací n-tice jsou popsány pomocí *n* atributů. Každá n-tice reprezentuje bod v n-rozměrném prostoru. Pokud je na vstup klasifikátoru přivedena neznámá matice, tak klasifikátor vyhledá takový vzor (*k* trénovacích n-tic), který je nejblíže neznámé matici. Tyto *k* trénovací n-tice jsou poté k-nejbližší sousedé neznámé matice.

„Blízkost" n-tic je poté počítána jako vzdálenost v Euklidovském prostoru pomocí vztahu

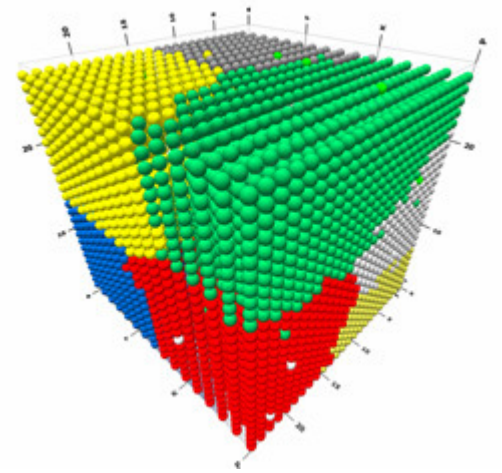
$$d(X,Y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2},$$

kde $X = (x_1, x_2, \dots, x_n)$ a $Y = (y_1, y_2, \dots, y_n)$ jsou dva vzorky dat.

3.2.2 Shlukové algoritmy

3.2.2.1 K-means

K-means neboli metoda založená na centrálním bodu je řazena mezi rozdělovací metody. Princip této metody spočívá v reprezentaci třídy pomocí fiktivního centrálního bodu. Atributy tohoto bodu jsou určeny jako střední hodnota hodnot atributů objektů přiřazených do dané třídy. Rozdělení objektů do tříd je prováděno na základě vzdálenosti od centrálních bodů.

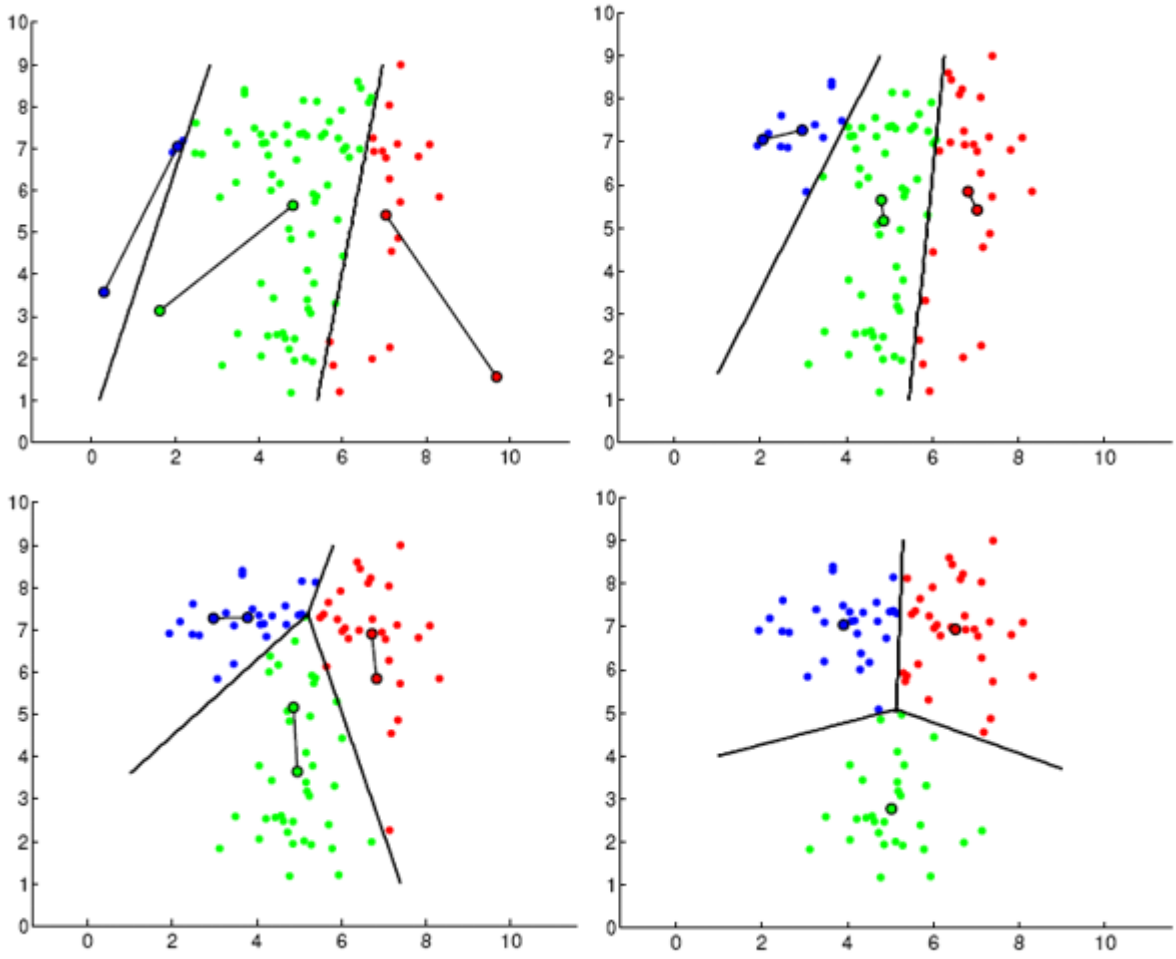


Obrázek 3.7: Shluková analýza pomocí K-means ve 3D [M3D]

Rozdělování je prováděno v iteračních krocích do té doby, dokud nedojde k přesunu žádného bodu mezi třídami a tím pádem nedojde k změně centrálních bodů. Algoritmus je možné popsat pomocí následujících kroků:

1. Nejprve je třeba zvolit počet shluků *k*
2. V prostoru je zvoleno *k* centrálních bodů
3. Body jsou přiřazeny k tomu centrálnímu bodu, ke kterému jsou nejblíže
4. Je spočítána nová pozice centrálních bodů na základě průměru k nim přiřazených bodů
5. Pokud se změnila pozice centroidů, je celý proces opakován od bodu 2

Výhodou této metody je rychlá konvergence k lokálnímu optimu ztrátové funkce, naopak její nevýhodou je nutnost zadat počet shluků, kdy je obtížné zvolit optimální počet.



Obrázek 3.8: Iterační kroky algoritmu K-means [KME]

3.2.2.2 C-means

C-means bývá označován jako fuzzy shlukovací algoritmus. Fuzzy, neboli neurčitost, spočívá v tom, že daný objekt může být zařazen do více shluků.

Mějme množinu *X* s *n* objekty, kde každý objekt je popsán *m* atributy. Mějme též množinu *C* s *k* třídami. Hlavním rysem fuzzy c-means algoritmu je, že každému objektu *x* z *X* přiřazuje algoritmus hodnost vyjadřující členství $\mu(x,c)$, takový, že $\mu(x,c)$: $X \times C \rightarrow [0,1]$. Algoritmus se

zastaví, jestliže změna hodností všech objektů se sníží pod definovaný práh. Pro měření vzdáleností se používá Eukleidovská vzdálenost. Algoritmus pracuje velice špatně, pokud se tvary shluků výrazně odlišují od sférických tvarů. [CME]

3.2.2.3 K-medoids

Metoda K-medoids, neboli metoda založená na reprezentujícím objektu, je stejně jako K-means metodou rozdělovací. Je obdobná jako K-means, modifikace spočívá v tom, že jednotlivé třídy nejsou reprezentovány fiktivním centrálním bodem, ale jedním z bodů z množiny analyzovaných dat. Takový bod je nazýván reprezentující bod (medoid).

3.2.2.4 Hierarchické metody

Hierarchické shlukovací metody fungují na principu seskupování skupin objektů do stromové struktury shluků. V závislosti na hierarchické dekompozici jsou tyto metody děleny na aglomerativní a dělící.

V případě aglomerativního shlukování se jedna o bottom-up metody shlukování, jejichž princip spočívá v tom, že je nejprve každý objekt přiřazen do zvláštní třídy a následně jsou podobně třídy shlukovány do větších celků. Celý proces je prováděn, dokud není dosaženo požadované úrovně shlukování.

Oproti tomu při dělícím shlukování (top-down metody) jsou všechny objekty umístěny do jedné třídy a následně jsou v iteračních krocích děleny do jednotlivých tříd.

4 Informační systém Belinda

4.1 Webové informační systémy

Technologie vývoje informačních systémů se posouvá směrem k internetu. Tyto informační systémy se stávají běžnou součástí firem či organizací působících v nejrůznějších oborech.

Informačním systémem je obecně myšlen počítačový systém pro sběr a zpracování informací a dat. Je to speciální typ komunikačního média, jehož cílem je odstranit bariéry v přístupu k informacím. Existuje mnoho typů informačních systémů, rozdělují se dle jejich využití. Informační systémy jsou být určeny například k podpoře provozu firem, podpoře plánování, řízení vztahů se zákazníky (CRM systémy) nebo k podpoře činností a služeb organizace.

Co rozumíme pod pojmem webový informační systém? Zjednodušeně by se dalo říci, že se jedná o klasický informační systém určený k provozu v podmínkách world-wide webu. Pokud bychom hledali přesnější definici, dalo by se říci, že webový informační systém (WIS) je definován jako takový parametrizovaný systém, který je provozován v nelineárním nestavovém síťovém prostředí (a nabízí tedy svým uživatelům v maximální míře přizpůsobení jeho chování pomocí nastavení parametru).

4.2 O systému Belinda

4.2.1 Vývoj systému

Systém Belinda je webový informační systém firmy Voxnet s.r.o. Systém byl vytvořen na míru společnosti v roce 2003 a je nadále vyvíjen. Systém slouží pro podporu činnosti vlastního call centra a pro zefektivnění práce obchodníků firmy. Hlavní vývojář firmy Voxnet s.r.o. Dušan Stloukal popisuje informační systém Belinda a jeho vývoj takto:

Informační systém Belinda byl vytvořen v jazycích PHP a Perl a využívá relační databázový systém PostgreSQL. Výše uvedené jazyky a databáze byly zvoleny kvůli jejich dostupnosti (jsou šířeny pod open source licencí) a velké rozšířenosti.

Současná Belinda je ve verzi 3.0. Mezi jednotlivými verzemi se od roku 2003 měnila jak funkčnost systému (některé moduly přibyly, jiné zase zmizely), tak hardwarové vybavení včetně databázových serverů, ale hlavně přístup k programátorské koncepci.

Verze 1.0 byla vytvořena tzv. „na koleni“ a celá aplikace je tvořena kalendářem a základním řízením workflow pro callcentrum. Hlavními programovacími jazyky jsou HTML a PHP v kombinaci

s databázovým systémem MySQL. S rostoucí firmou a zvětšujícím se počtem zaměstnanců se zvětšovaly také nároky na informační systém.

Od verze 1.5 přebírá zodpovědnost za vývoj systému Dušan Stloukal, současný hlavní vývojář. V této verzi jsou doplněny některé funkcionality kalendáře a správy call centra.

Ve verzi 2.0 dochází k radikálním změnám. Dochází k migraci na databázový systém PostgreSQL, původní jádro systému postavené na principu „PHP v HTML“ je od základu nahrazeno procedurálním programováním v PHP. Důležitým krokem je také zřízení vývojového prostředí, používání systému pro správu verzí CVS a změny ve struktuře databáze směrem k dodržování alespoň základních principu normálních forem.

Při vývoji současné verze 3.0 přestávají vyhovovat omezené možnosti nastavení hostingového serveru a je rozhodnuto o pořízení vlastního serveru. Procedurální PHP jádro informačního systému je kompletně přepsáno a nahrazeno objektovým, dochází také k zásadním změnám uživatelského rozhraní.

Roadmap

Nejpalčivějším problémem současného informačního systému Belinda je její nedostačující reflexe na požadavky uživatelů. V aktuálně vyvíjené nové verzi dojde k přechodu na databázový systém Oracle 10.2, který umožňuje programovat aplikační vrstvu na úrovni databáze (jazyk PL/SQL, případně volání Perlových funkcí). Front-end aplikace bude zajišťovat MVC framework (s největší pravděpodobností Catalyst nebo Zend).

Advanced queuing v Oracle umožní implementaci plnohodnotného a špičkového řízení workflow ticketu, možnost ukládání a fulltextového prohledávání dokumentů v binárních formátech (DOC, PDF) pak umožní snadnou implementaci document managementu.

4.2.2 Funkčnost systému

V této kapitole bude popsána základní funkčnost informačního systému Belinda. Pro lepší představu podoby aplikace bude popis doplněn náhledy, které jsou jako příloha této práce. Některé náhledy jsou retušovány z důvodu obsahu citlivých údajů o klientech firmy Voxnet s.r.o.

Úvodní stránka

Úvodní stránka aplikace s navigací v jednotlivých sekcích systému (obrázek P.5).

Vyhledávání

Podrobné vyhledávání ve službách a seznam služeb dle zvolené kategorie produktu (obrázek P.1).

Fakturační historie služby

Detailní náhled na fakturační historii služby včetně grafu (obrázek P.2). Zelenou barvou jsou označeny zaplacené faktury, žluté jsou neuhrazené faktur a červenou barvou jsou zvýrazněny faktury, o nichž nebyly dodány žádné informace (například faktura může být vystavena a zaplacena, ale poskytovatel služby tuto fakturu nepřiřadil danému partnerovi).

Kalendář

Jednoduchý kalendář obsahuje správu pracovních i nepracovních schůzek zobrazovaných po jednotlivých dnech. Uživatelé (obchodníci) si mohou zadávat své schůzky či svolávat schůzky s ostatními spolupracovníky, kteří mají možnost pozvání potvrdit či odmítnout. Nemalou část schůzek tvoří dojednané schůzky s klienty zadané pracovníky call centra (obrázek P.3).

Administrace oprávnění

Část systému přístupná pouze administrátorovi umožňující správu oprávnění uživatelů. Každému uživateli či skupině uživatelů lze nastavit přístupová práva zvlášť pro každou sekci systému. Práva jsou rozdělena do několika úrovní – čtení záznamů, čtení vlastních záznamů, vkládání záznamů a editace záznamů (obrázek P.4).

Tickety

Sekce systému umožňující kompletní správu ticketů v call centru. Tickety představují jednoduché workflow, kdy je možné zaznamenat a plánovat akce u firem (stávajících i potenciálních zákazníků). Každý ticket reprezentuje soubor jedné, typicky však více po sobě jdoucích činností. U každé je zaznamenáno, kdo ji naplánoval a především kdy a na jaký čas. Dále ticket může obsahovat informace o zákazníkovi, umožňuje založit novou schůzku a přiřadit ji obchodníkovi, odložit hovor či změnit prioritu. Příklad práce s ticketem může být následující:

Ticket je založen na základě informace, že v oblasti byl postaven nový DSLAM poskytovatele a je tak možné firmám na dostupných telefonních linkách nabídnout výhodné služby. Na základě externí databáze firem jsou vybrány vhodné firmy (především podle adresy a velikosti) a je založen ticket ve stavu „lustrace“. Pracovník následně zjistí, zda je na základě dostupných informací možné firmu kontaktovat a zda využívá telefonní čísla dostupná novému DSLAMu). Pokud ano, je stav ticketu přesunut na „hovor“ – tento stav je zachycen na náhledu (obrázek P.6). Následně operátoři call centra kontaktují firmy a nabízí služby. Na základě uskutečněného hovoru může být naplánována schůzka, hovor může být odložen či ukončen. V případě odložení je ticket za určitou dobu automaticky opět zařazen do fronty. Z historie ticketů lze poté získat údaje o klientech a jejich reakcích, uskutečněných hovorech či schůzkách.

Filtry

Každému operátorovi call centra jsou zobrazeny tickety ve stavu hovor. Na základě filtrů lze určit parametry, podle kterých budou každému z operátorů přiřazeny dané tickety. Lze například nastavit z jakého zdroje (databáze) budou použity kontakty firem či v kterém městě mají dané kontakty být (obrázek P.7).

Produkty

Umožňuje kompletní správu produktového portfolia. Nabízené produkty jsou rozděleny do skupin (hlas, internet, atd.), je možné je upravovat, nastavovat jako aktivní či neaktivní, vyhledávat atd. (obrázek P.8).

Sklad

Modul Sklad (obrázek P.9) umožňuje evidenci drobného zboží, které je koncipováno jako doplněk k prodeji telekomunikačních služeb (tzn. modemy, routery, síťové prvky, atd.). Hlavní myšlenkou skladu je, aby každý obchodník měl možnost přímo při návštěvě zákazníka zjistit, které zboží může nabídnout ihned, které později a za jakou cenu. Dále je pak evidováno, který zákazník jaké zboží zakoupil. Primitivní workflow umožňuje automatický návrat zboží ze stavu rezervováno do stavu na skladě v okamžiku dosazení data rezervace (a pokud je zboží stále ve stavu rezervováno).

5 Analýza dat v prostředí SAS Enterprise Miner

5.1 SAS Enterprise Miner

SAS Enterprise Miner firmy SAS Institute je produktem pro pokročilé analýzy dat, který umožňuje řešit problémy komunikace obchodníků, uživatelů a analytiků při definování cílů analýz i řízení projektů dolování dat - to vše při zachování otevřenosti vůči novým metodám analýz. [DM99]

5.2 Vstupní data

Vstupní data pro dolování jsou uložena v relační databázi PostgreSQL, import do aplikace SAS Enterprise Miner byl proveden ve formátu CSV. Tabulka s daty byla redukována jen na potřebné údaje. Na níže uvedeném obrázku 5.1 je struktura tabulky s vybraným vzorkem dat.

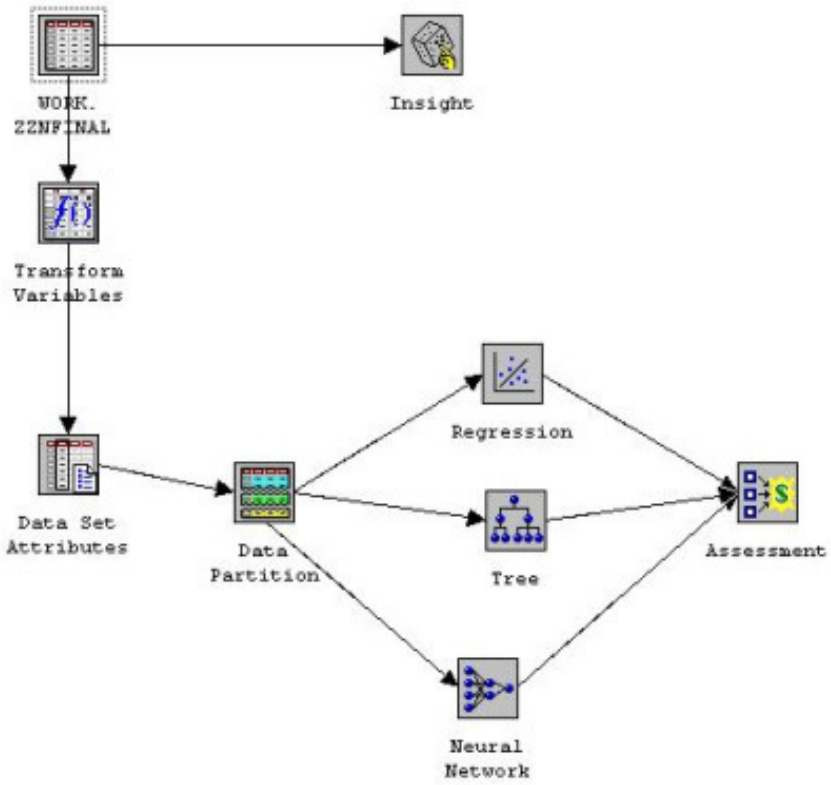
firma_id	pocet_zamestnancu_id	zakladni_kapital	obrat	sluzba_id	obchodnik_id	invoice_amount
23241	230	175000	19	18	264	1020.70
15964	120	600000	10	18	264	300.00
12095	120	200000	9	18	266	905.10
16870	220	100000	19	18	266	1200.00
16297	320	105259000	306	18	264	1474.48
23464	310	200000	173	18	264	8318.45
10637	240	105000	96	19	264	5000.00
5866	120	200000	1	18	264	938.28
5866	120	200000	1	18	264	796.35
26689	230	500000	51	17	276	745.00
18942	120	300000	9	17	261	945.00
25249	120	1250000	4	18	261	1877.82
26784	120	100000	9	31	274	290.00

Obrázek 5.1: Ukázka zdrojových dat

Tabulka obsahuje informace o jednotlivých firmách – obsahuje unikátní identifikátor firmy (firma_id), počet zaměstnanců (pocet_zamestnancu_id), základní kapitál (zakladni_kapital), obrat, identifikátor služby (sluzba_id), kterou si daná firma objednala, identifikátor obchodníka (obchodnik_id), který obchod realizoval a částku (invoice_amount), kterou firma platila za hlasové nebo datové služby před objednáním nové služby.

5.3 Blokové schéma

Nastavení a celý proces dolování dat v prostředí SAS Enterprise Miner se provádí v intuitivním a uživatelsky přívětivém rozhraní. Celý proces je sestaven z jednotlivých komponent, které jsou navzájem propojeny. Na obrázku 5.2 je blokové schéma znázorňující propojení jednotlivých předdefinovaných komponent systému použitých při dolování a toky dat mezi nimi.



Obrázek 5.2: Schéma komponent

Input Data Source

Komponenta sloužící k mapování importovaných dat ke konkrétní dolovací úloze.

Insight

Pomocí komponenty Insight lze prohlížet namapovaná data komponenty, která je asociována s Insight pomocí vazby. V případě uvedeného schématu je tedy možné prohlížet data, která jsou na vstupu této úlohy.

Transform Variables

Transformace proměnných slouží k modifikaci stávajících proměnných nebo k vytvoření nových, jejichž výpočet lze nadefinovat pomocí elementárních operací.

Data Set Attributes

Slouží k nastavení proměnných, které budou použity jako vstupy (input) a výstupy (target) pro dolovací metody a k určení, které proměnné nemá systém brát v úvahu při dolování (rejected). Vždy je nutné zvolit alespoň jednu vstupní proměnnou a právě jednu výstupní.

Data Partition

Pomocí komponenty Data partition jsou vstupní data rozdělena procentuálně do tří množin. První z nich je množina trénovací, na těchto datech se zvolená dolovací metoda bude učit. Na množině validačních dat bude testováno, zda je metoda dostatečně naučena. Data v poslední, testovací množině budou použita pro samotné dolování.

Regression, Tree, Neural Network

Tyto tři komponenty provádí dolování pomocí zvolených metod, tedy pomocí regrese, rozhodovacího stromu a neuronové sítě.

Assessment

Komponenta Assessment (zhodnocení analýzy) slouží k vyhodnocení výsledků použitých dolovacích metod a k zobrazení souhrnných informací.

5.4 Provedené testy

V této kapitole jsou uvedeny 2 vybrané testy provedené v prostředí SAS Enterprise Miner. Cílem dolovací úlohy bylo předpovězení nebo odhad služby na základě dostupných informací o firmě.

V prvním testu bylo provedeno předpovězení služby na základě obratu firmy, počtu zaměstnanců, základního kapitálu a faktury. Ve druhém testu byly jako vstupní data použity informace o faktuře firmy za telefonní služby a identifikátor obchodníka, který daný kontrakt realizoval.

Při testování bylo použito více kombinací jednotlivých údajů na vstupu, ale při použití uvedených kombinací bylo dosaženo nejlepších výsledků.

Jako vstupní hodnoty byl použit vzorek dat obsahující 1000 záznamů, rozdělení dat na vstupu bylo provedeno následovně:

Trénování	40% dat
Validace	30% dat
Testování	30% dat

Na obrázcích 5.3 a 5.4 je uvedeno nastavení proměnných v prostředí SAS Enterprise Miner pro Test 1 a pro Test 2. V prvním testu byly jako vstupní proměnné nastaveny pocet_zamestnancu_id, obrat a invoice_amount. Ve druhém testu byly na vstupu proměnné obchodnik_id a invoice_amount. V obou testech jsme se snažili odhadnout hodnotu sluzba_id.

Name	Model Role	Measurement	Type	Format	Informat
FIRMA_ID	id	nominal	char	$7.	$7.
POCET_ZAMESTNANCU_ID	input	nominal	char	$5.	$5.
ZAKLADNI_KAPITAL	rejected	nominal	char	$12.	$12.
OBRAT	input	nominal	char	$6.	$6.
SLUZBA_ID	target	ordinal	char	$4.	$4.
OBCHODNIK_ID	rejected	nominal	char	$5.	$5.
INVOICE_AMOUNT	input	nominal	char	$11.	$11.

Obrázek 3.3: Nastavení proměnných pro Test 1

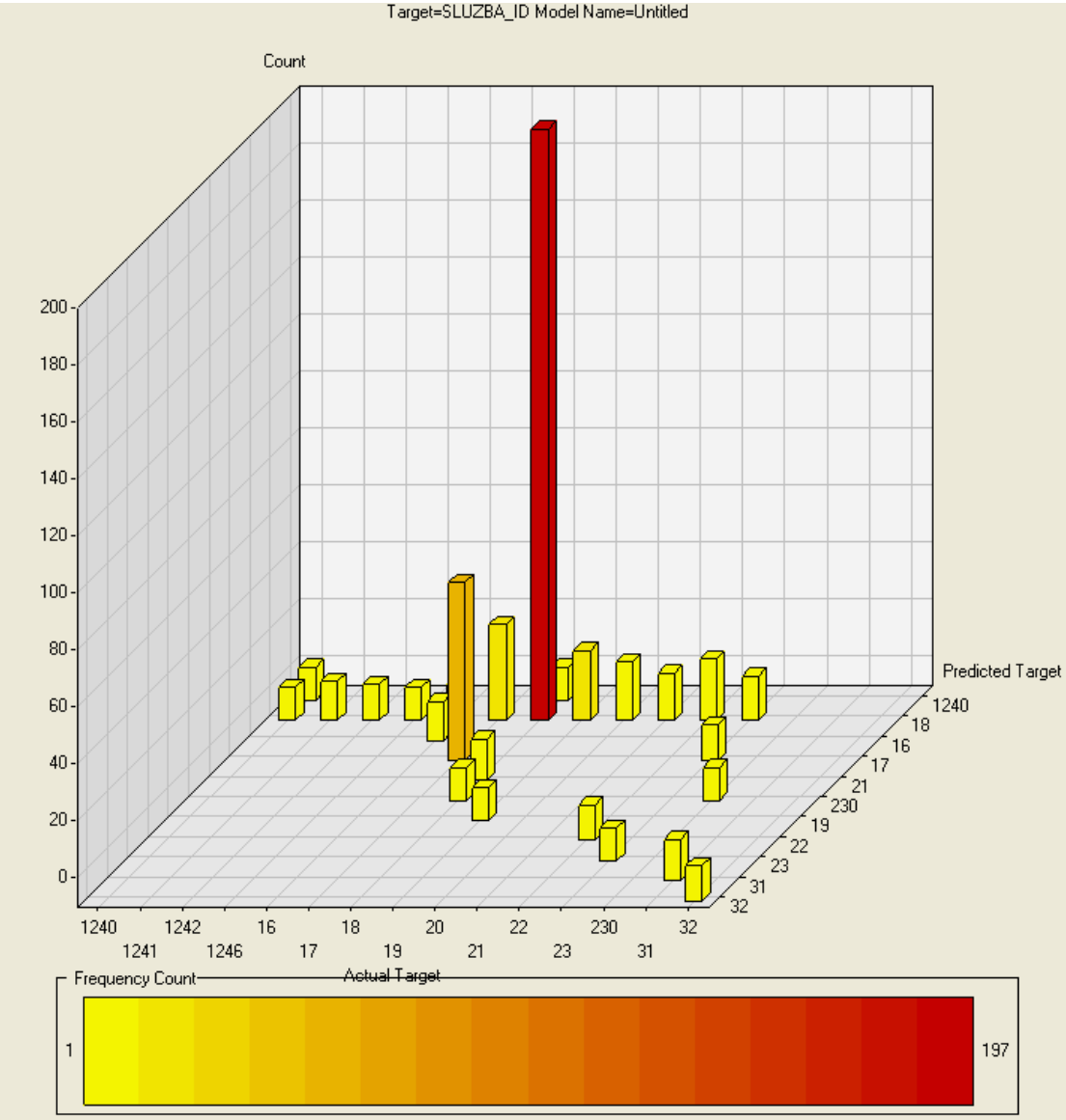
Name	Model Role	Measurement	Type	Format	Informat
FIRMA_ID	id	nominal	char	$7.	$7.
POCET_ZAMESTNANCU_ID	rejected	nominal	char	$5.	$5.
ZAKLADNI_KAPITAL	rejected	nominal	char	$12.	$12.
OBRAT	rejected	nominal	char	$6.	$6.
SLUZBA_ID	target	ordinal	char	$4.	$4.
OBCHODNIK_ID	input	nominal	char	$5.	$5.
INVOICE_AMOUNT	input	nominal	char	$11.	$11.

Obrázek 5.4: Nastavení proměnných pro Test 2

5.4.1 Test 1 – Odhad služby na základě obratu, počtu zaměstnanců a faktury

Při výše uvedené konfiguraci testů byly pomocí regrese, rozhodovacího stromu a neuronové sítě získány následující výsledky:

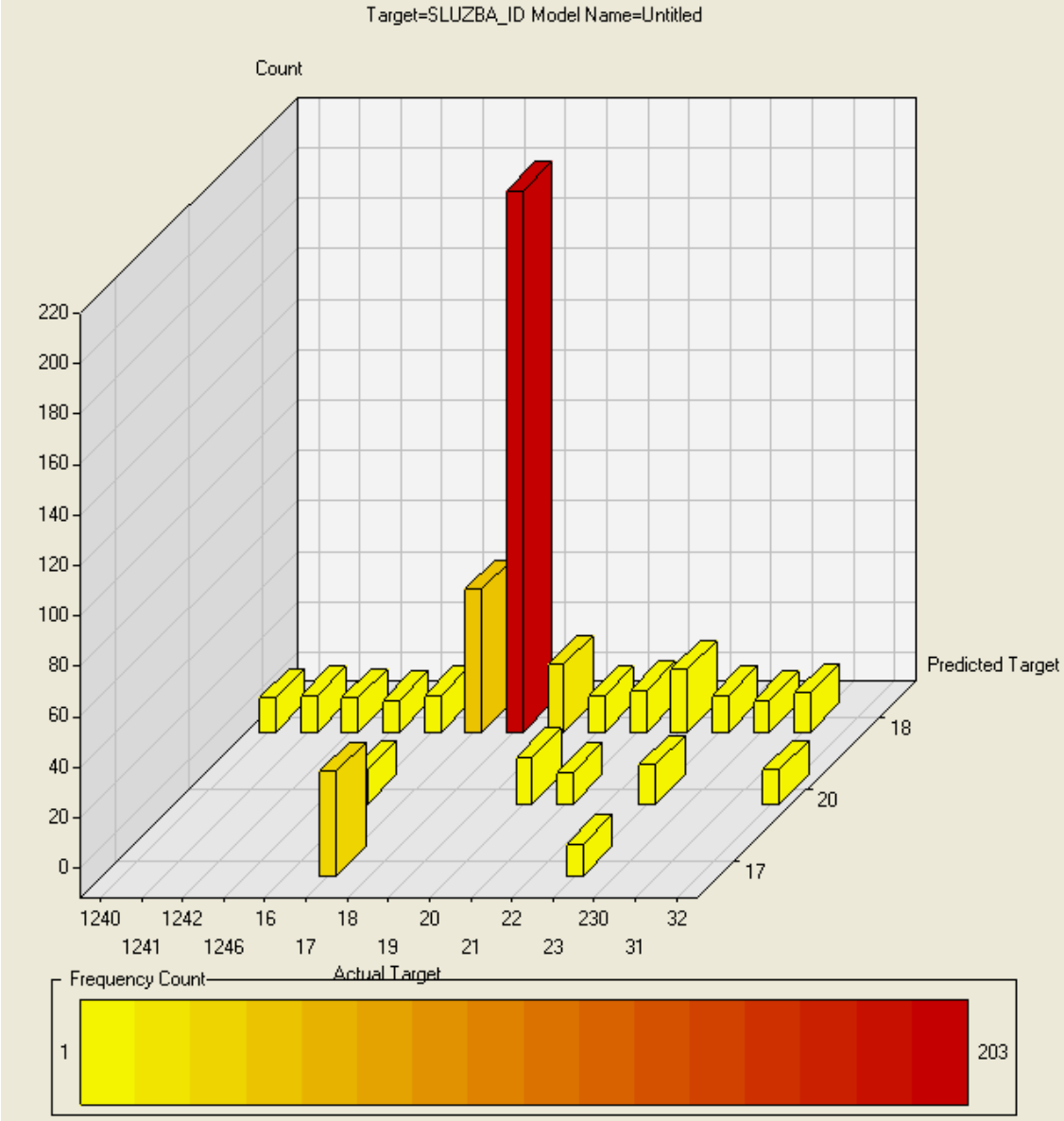
5.4.1.1 Regrese



Obrázek 5.5: Výsledek metody regrese Test 1

Na obrázku 5.5 je diagnostický graf metody regrese a lze vysledovat, že nejvíce předpovídanou službou je služba s identifikátorem 18. Rozložení hodnot v grafu není úplně optimální, ale v porovnání s jinými použitými metodami bylo pomocí regrese dosaženo nejlepších výsledků.

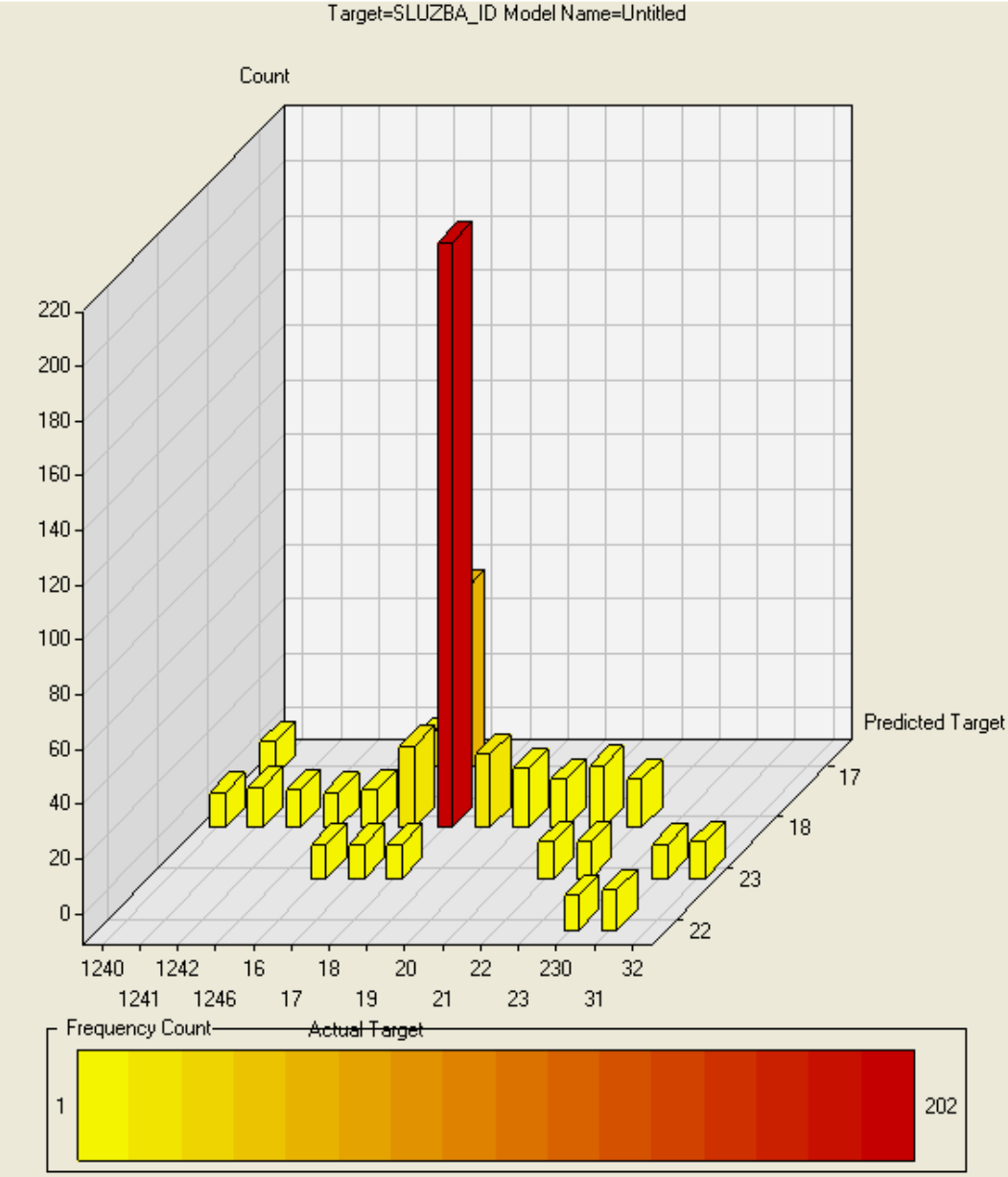
5.4.1.2 Rozhodovací strom



Obrázek 5.6: Výsledek metody rozhodovací strom Test 1

Obrázek 5.6 zachycuje diagnostický graf výsledku analýzy provedené pomocí rozhodovacího stromu. Jak je patrné z grafu, pomocí této metody nebylo dosaženo takových výsledků jako pomocí regrese, rozhodovací strom navíc výsledné hodnoty klasifikoval pouze do 3 tříd.

5.4.1.3 Neuronová síť



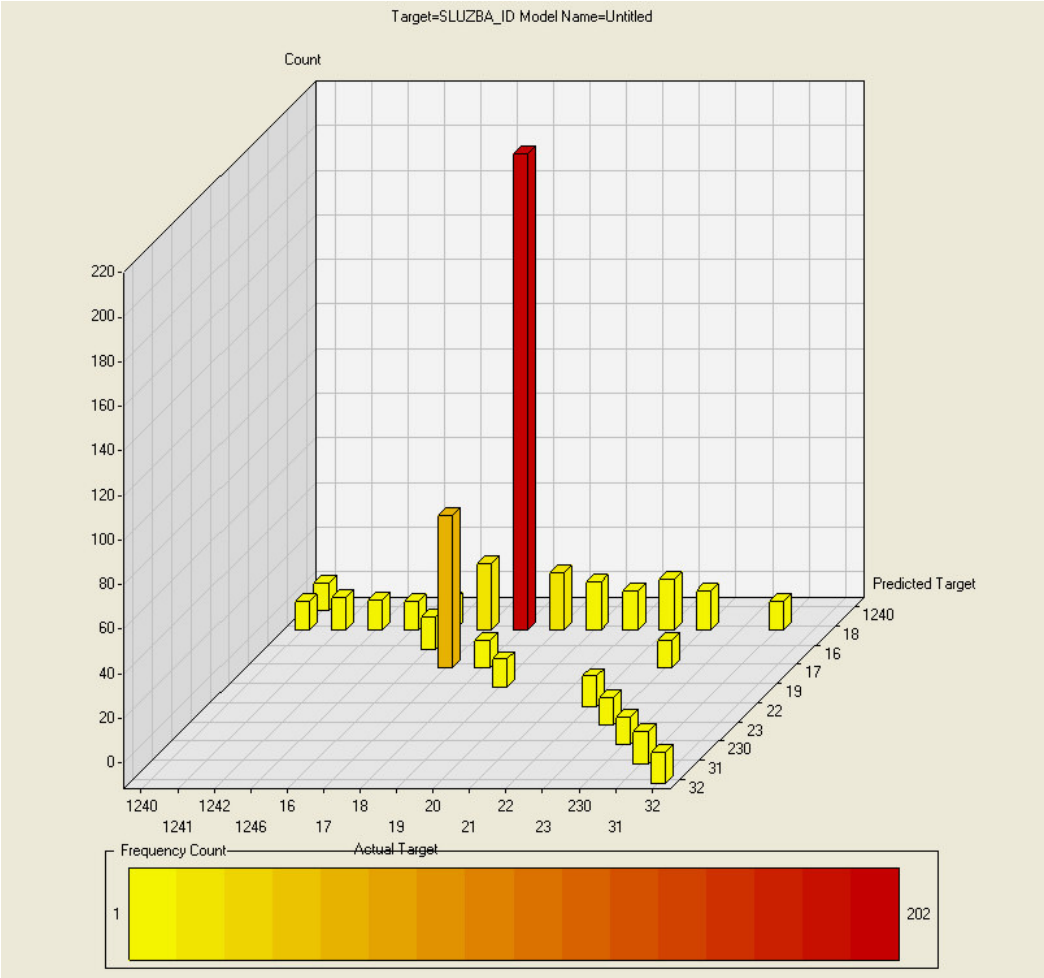
Obrázek 5.7: Výsledek metody neuronová síť Test 1

Pomocí neuronové sítě (obrázek 5.7) bylo dosaženo podobných výsledků jako v případě použití rozhodovacího stromu. Jak již bylo výše uvedeno, nejlepších výsledků bylo použito v případě použití metody regrese.

5.4.2 Test 2 – Odhad služby na základě faktury firmy a obchodníka

Při Testu 2 byly použity stejné metody jako v případě předchozího testování. Pomocí regrese, rozhodovacího stromu a neuronové sítě byly získány následující výsledky:

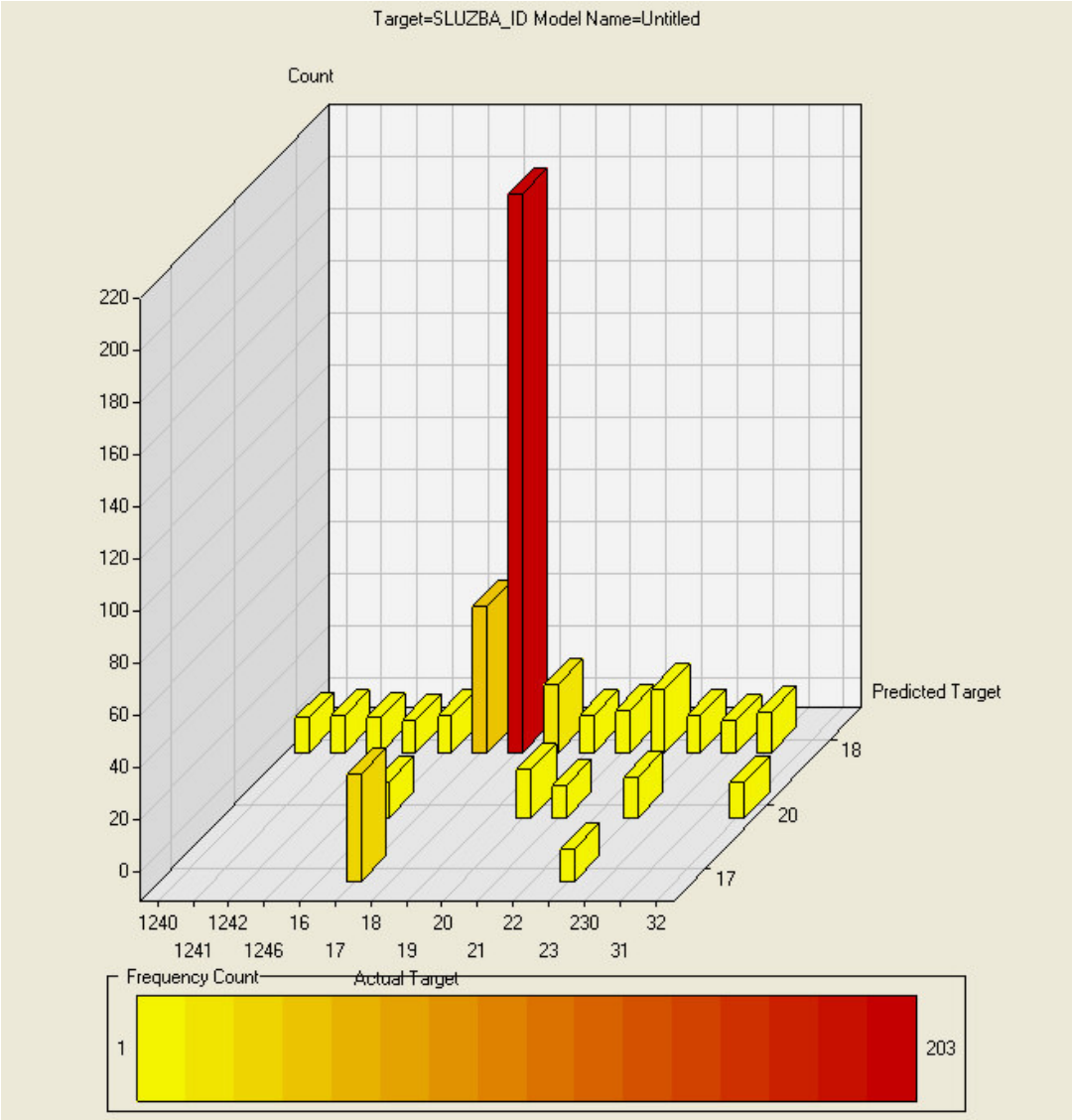
5.4.2.1 Regrese



Obrázek 5.8: Výsledek metody regrese Test 2

Na obrázku 5.8 je výsledek analýzy metodou regrese v Testu 2, stejně jako v předchozím testu i zde bylo pomocí této metody dosaženo nejlepších výsledků.

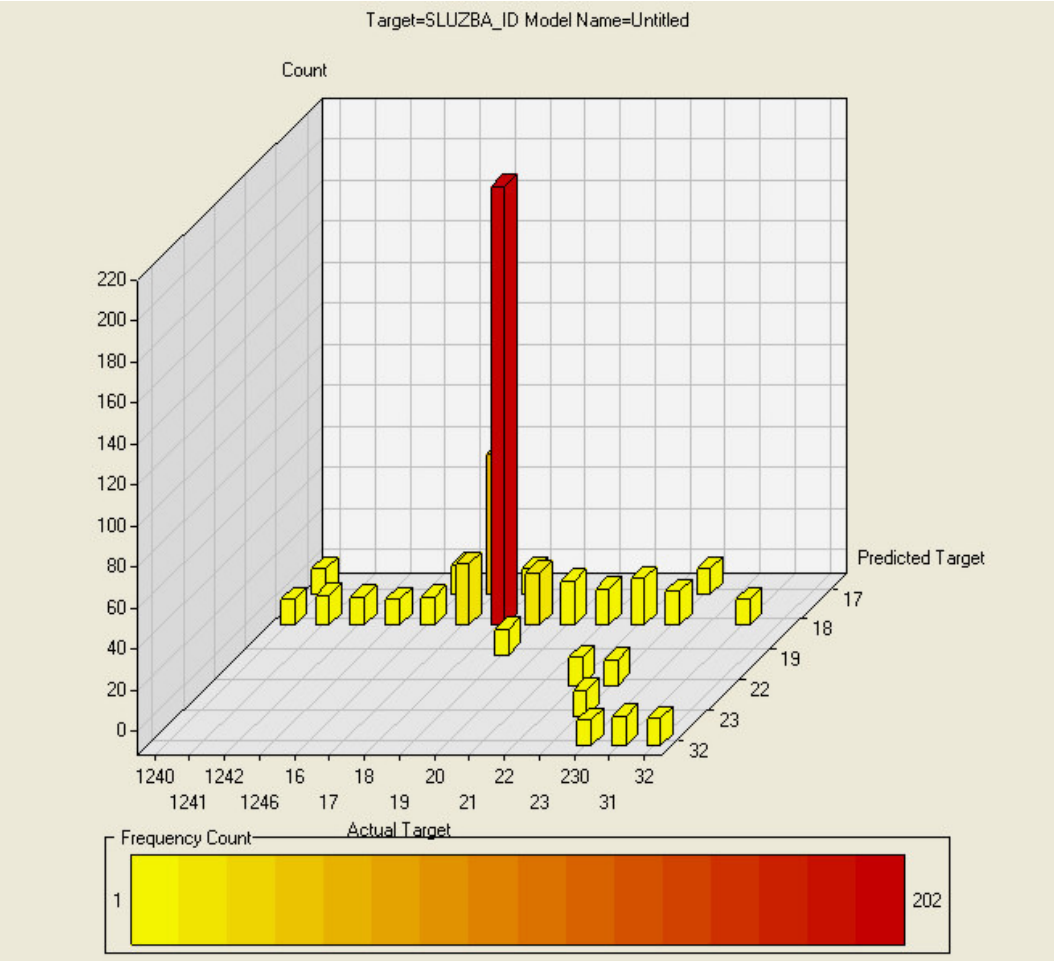
5.4.2.2 Rozhodovací strom



Obrázek 5.9: Výsledek metody rozhodovací strom Test 2

Stejně jako v předchozím testu byly nejhorší výsledky dosaženy pomocí metody rozhodovacího stromu (obrázek 5.9). Výsledné hodnoty byly opět klasifikovány pouze do tří nejčastěji se vyskytujících tříd.

5.4.2.3 Neuronová síť



Obrázek 5.10: Výsledek metody neuronová síť Test 2

Na obrázku 5.10 jsou výsledky Testu 2 pomocí neuronové sítě. Dosažené výsledky jsou lepší než u rozhodovacího stromu, ale stejně jako u ostatních metod je příliš často a nesprávně předpovídána hodnota 18.

5.5 Zhodnocení

Byla provedena analýza dat z databáze klientů firmy Voxnet s.r.o. v prostředí SAS Enterprise Miner. Cílem projektu byla snaha o odhadnutí služby z portfolia firmy Voxnet s.r.o., kterou si potenciální klient objedná.

Pro získání znalostí z dat byly použity metody regrese, rozhodovací strom a neuronová síť, přičemž nejlepších výsledků bylo dosaženo pomocí regrese. Při analýze získaných výsledků a

vstupních dat je patrné, že systém často nesprávně odhaduje služby, které se ve vstupních datech vyskytují zřídka. Místo nich předpovídá často se vyskytující služby. Tyto chyby v odhadu jsou způsobeny vzorkem dat a také tím, že služba a ostatní vstupní hodnoty jsou na sobě zcela nezávislé.

6 Implementace modulu

6.1.1 Implementace

Modul pro dolování dat byl dle zadání této práce a požadavků firmy Voxnet s.r.o. implementován za použití skriptovacího jazyka PHP a databázového systému PostgreSQL za účelem snadného napojení na stávající webový informační systém Belinda.

Jádro modulu tvoří dvě dolovací metody, a to regrese a neuronová síť. Tyto metody a konkrétní implementované algoritmy budou podrobněji popsány v následující kapitole.

Načítání vstupních dat pro dolování je prováděno z databázových tabulek. Formát vstupních dat může být měněn uživatelem, v nastavení modulu je možné nastavit název tabulky a jednotlivé sloupce tabulky, které budou dostupné pro dolování. Často používané nastavení lze uložit jako předdefinované hodnoty a urychlit tak práci s nastavováním vstupních dat.

Před započetím samotného dolování je možné zkontrolovat stav vstupních dat. Kontrolní skript upozorní na přítomnost záporných hodnot či na chybějící hodnoty a navrhne uživateli možné řešení nastalého problému. Neúplné záznamy lze odstranit z testovacího vzorku dat nebo chybějící hodnoty automaticky nahradit, aby negativně neovlivňovaly výsledek analýzy. Také je uživateli ponechána možnost upravit vzorek dat ručně.

Nastavení hlavní fáze dolování (náhled obrazovky je zachycen na obrázku 6.1) umožňuje výběr použité metody (regrese nebo neuronová síť) a nastavení atributů. Každý z atributů (sloupců tabulky) může být nastaven jako vstupní, cílový nebo jako nepoužívaný pro danou úlohu. Pro korektní spuštění úlohy je třeba zvolit alespoň jeden vstupní atribut a právě jeden cílový, jehož hodnoty budou predikovány. Dále je třeba vstupní data rozdělit na trénovací a testovací množinu. Uživatel může nastavit, jaké procento z celkového počtu záznamů se použije k natrénování zvolené metody na vstupní data. Na zbytku dat, označených jako testovací množina, poté probíhá samotná predikce pomocí naučené metody. Pomocí zbývajících voleb lze učit velikost vzorku vstupních dat nebo provést normalizaci dat pomocí min-max normalizace. Při použití neuronové sítě lze zvolit velikost sítě (počet neuronů ve skryté vrstvě) a počet iteračních kroků ve fázi učení.

Obrázek 6.1: Náhled nastavení dolovacích metod

Po dokončení procesu dolování jsou uživateli zobrazeny výsledky ve formě tabulky, která obsahuje vstupní a výstupní atributy a porovnání predikovaných a skutečných hodnot. Tyto výsledky může uživatel libovolně řadit, upravovat zobrazení atributů či provést export. Data jsou exportována ve formátu XLS ve dvou možných podobách. První z nich je ve formě tabulky totožné s tabulkou zobrazenou uživateli v aplikaci. V druhém případě jsou data upravena do tabulky tak, aby bylo možné snadno vytvořit graf znázorňující úspěšnost predikce.

6.1.2 Back-end aplikace

K implementaci byly zvoleny metody regrese a neuronová síť, konkrétně vícenásobná lineární regrese a neuronová síť backpropagation. Principy fungování těchto metod jsou popsány podrobněji v kapitole Procesy a algoritmy dolování dat. Zde bude popsána funkčnost a detaily implementace konkrétních algoritmů.

6.1.2.1 Vícenásobná lineární regrese

Vstupní data pro dolovací úlohu jsou dle nastavení uživatele procentuálně rozdělena do dvou množin. První množina dat, testovací, slouží k naučení algoritmu. V tomto případě jsou pomocí těchto dat

spočítány koeficienty aproximační rovnice $Y = a_0 + a_1X_1 + a_2X_2 + \ldots + a_nX_n$, kde X je množina n-tic vstupních dat, Y je množina výstupních dat a A je množina koeficientů $a_0, a_1, a_2, \ldots, a_n$.

Výpočet koeficientů probíhá dosazením do následujícího vztahu $A = (X^T X)^{-1} X^T Y$, kde X^T je matice transponovaná a $(X^T X)^{-1}$ je inverzní matice vzniklá součinem matic X^T a X.

Postup výpočtu

Vstupní data z trénovací množiny ve formě n-tic $(x_{11}, x_{12}, \ldots, x_{1n}, y_1), (x_{21}, x_{22}, \ldots, x_{2n}, y_2), \ldots, (x_{t1}, x_{t2}, \ldots, x_{tn}, y_n$ jsou uloženy do matice. Matice je doplněna sloupec 1 v prvním sloupci matice. Výstupní data jsou načtena do sloupcové matice.

$$X = \begin{pmatrix} 1 & x_{11} & x_{12} & \ldots & x_{1n} \\ 1 & x_{21} & x_{22} & \ldots & x_{2n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_{t1} & x_{t2} & \ldots & x_{tn} \end{pmatrix} \qquad Y = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix}$$

V následujícím kroku je vytvořena matice transponovaná matice *X*. Princip transpozice je triviální a spočívá v záměně řádků a sloupců matice tak, že pro všechny prvky matice platí $a_{ij}^T = a_{ji}$.

$$A = \begin{pmatrix} a_{11} & a_{12} & \ldots & a_{1n} \\ a_{21} & a_{22} & \ldots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \ldots & a_{mn} \end{pmatrix} \qquad A^T = \begin{pmatrix} a_{11} & a_{21} & \ldots & a_{m1} \\ a_{12} & a_{22} & \ldots & a_{m2} \\ \vdots & \vdots & \ddots & \vdots \\ a_{1n} & a_{2n} & \ldots & a_{mn} \end{pmatrix}$$

Po vytvoření matice transponované je tato matice vynásobená maticí X. Součin matic se provádí jako skalární součin vektoru řádku první matice s vektorem sloupce druhé matice. Tento výsledek se pak zapíše na pozici ve výsledné matici, jejíž index odpovídá číslu řádku první matice a číslu sloupce druhé matice.

V následující fázi dochází k výpočtu matice inverzní. Tento výpočet již není zcela triviální a vyžaduje použití sofistikovanějšího řešení. Inverzní matice k dané matici je taková matice, která po vynásobení s původní maticí dá jednotkovou matici. Tedy platí vztah $AA^{-1} = A^{-1}A = 1$, kde 1 je jednotková matice. Výpočet inverzní matice lze provést pomocí determinantů a subdeterminantů nebo Gaussovou eliminační metodou. Výpočet pomocí determinantů je velmi časově náročný [INV], například pro matici 4. řádu je třeba vypočítat 16 determinantů 3. řádu. Vzhledem k tomu, že se předpokládají výpočty výrazně větších matic, byla tato metoda výpočtu zavržena a byl implementován výpočet pomocí Gaussovy eliminační metody.

Princip Gaussovy eliminace spočívá v řešení problému vyjádřeného pomocí matice převedením dané matice na horní trojúhelníkovou nebo na diagonální matici.

Výpočet inverzní matice k matici *A* probíhá následujícím způsobem:

1. K matici *A* je připsána jednotková matice.

$$A = \begin{pmatrix} a_{11} & a_{12} & \dots & a_{1n} & 1 & 0 & \dots & 0 \\ a_{21} & a_{22} & \dots & a_{2n} & 0 & 1 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} & 0 & 0 & \dots & 1 \end{pmatrix}$$

2. Matici je upravována tak, aby na místě původní matice *A* vznikla jednotková matice. Po odebrání nově vzniklé jednotkové matice je získána matice inverzní k matici *A*. Úprava matice probíhá pomocí následujících kroků:
    a. libovolná záměna pořadí řádku v matici
    b. vynásobení řádku skalárem
    c. přičtení jednoho řádku k jinému

3. Kontrolu výpočtu lze provést vynásobením matice *A* s její inverzní maticí. Pokud vynásobením vznikne jednotková matice, proběhl výpočet inverzní matice správně.

Po výpočtu inverzní matice je pomocí násobení matic dle daného vztahu získána matice A obsahující koeficienty aproximační rovnice a_0, a_1, a_2, ..., a_n.

Nyní je možné sestavit aproximační rovnici a provést fázi testování. Data z testovací množiny jsou přiváděna na vstup aproximační rovnice a jsou získávány výstupní hodnoty. Tyto získané hodnoty jsou porovnány s reálnými hodnotami a je spočtena úspěšnost predikce.

Naprogramovaný modul umožňuje sledovat celý proces výpočtu po jednotlivých krocích při zapnutí volby „debug mode“ při zahájení dolování. Průběh výpočtu můžeme demonstrovat na následujícím jednoduchém příkladu:

1. Uživatelem je nastaveno následující nastavení – 3 vstupní atributy, 1 výstupní, velikost trénovací a testovací množiny 55%, resp. 45% a bude použit vzorek dat obsahující 11 záznamů. Takto malý vzorek dat je zvolen pouze pro demonstrační účely tohoto výpočtu, v praxi nemá dolování na tak malém vzorku dat žádný smysl.

2. Jsou načtena trénovací data do vstupní a výstupní matice.

$$X = \begin{pmatrix} 1 & 4 & 1 & 3 \\ 1 & 4 & 0 & 4 \\ 1 & 4 & 1 & 2 \\ 1 & 4 & 0 & 2 \\ 1 & 3 & 0 & 3 \\ 1 & 3 & 0 & 2 \end{pmatrix} \qquad Y = \begin{pmatrix} 3 \\ 2 \\ 3 \\ 4 \\ 3 \\ 1 \end{pmatrix}$$

3. Je získána transponovaná matice X^T a proveden součin matic $X^T X$.

$$X^T = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 \\ 4 & 4 & 4 & 4 & 3 & 3 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 3 & 4 & 2 & 2 & 3 & 2 \end{pmatrix} \qquad X^T X = \begin{pmatrix} 6 & 22 & 2 & 16 \\ 22 & 82 & 8 & 59 \\ 2 & 8 & 2 & 5 \\ 16 & 59 & 5 & 46 \end{pmatrix}$$

4. K získané matici je přidána jednotková matice a pomocí Gaussovy eliminační metody je postupnými úpravami spočtena inverzní matice k matici *X*.

$$X^T X \mid 1 = \begin{pmatrix} 6 & 22 & 2 & 16 & 1 & 0 & 0 & 0 \\ 22 & 82 & 8 & 59 & 0 & 1 & 0 & 0 \\ 2 & 8 & 2 & 5 & 0 & 0 & 1 & 0 \\ 16 & 59 & 5 & 46 & 0 & 0 & 0 & 1 \end{pmatrix}$$

$$(X^T X)^{-1} = \begin{pmatrix} 12{,}83 & -3{,}33 & 1{,}33 & -0{,}33 \\ -3{,}33 & 1{,}08 & -0{,}58 & -0{,}17 \\ 1{,}33 & -0{,}58 & 1{,}08 & 0{,}17 \\ -0{,}33 & -0{,}17 & 0{,}17 & 0{,}33 \end{pmatrix}$$

5. V následujícím kroku je proveden součin matic $A = (X^T X)^{-1} X^T Y$ a je získána výsledná matice *A* obsahující koeficienty aproximační rovnice.

$$A = \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{pmatrix} = \begin{pmatrix} -0{,}67 \\ 1{,}17 \\ -0{,}17 \\ -0{,}33 \end{pmatrix}$$

6. Aproximační rovnice má tedy tvar

$$Y = a_0 + a_1 X_1 + a_2 X_2 + a_3 X_3 = -0{,}67 + 1{,}17 X_1 + (-0{,}17) X_2 + (-0{,}33) X_3$$

7. V posledním kroku jsou testovací data přivedena na vstup aproximační rovnice a jsou vypočteny předpovídané hodnoty.

   První tři sloupce výsledná matice tvoří vstupní data testovací množiny, 4. sloupec potom skutečné výsledky a poslední sloupec predikované hodnoty. Z matice výsledných hodnot je patrné, že hodnota byla správně předpovězena pouze v případě posledního vzorku dat.

$$A = \begin{pmatrix} 3 & 1 & 3 & 3 & 2 \\ 4 & 1 & 1 & 4 & 3 \\ 4 & 0 & 2 & 4 & 3 \\ 4 & 1 & 3 & 4 & 3 \\ 4 & 0 & 1 & 4 & 4 \end{pmatrix}$$

6.1.2.2 Neuronová síť Backpropagation

Neuronová síť Backpropagation je klasifikační učící se algoritmus. Stejně jako v případě implementace vícenásobné lineární regrese jsou data rozdělena do dvou množin dle nastavení uživatele. Na první množině dat, trénovací, bude provedeno naučení algoritmu na daný vzorek dat. Na testovací množině dat potom bude provedena samotná klasifikace.

Sestavení neuronové sítě

Před započetím fáze učení neuronové sítě je třeba zvolit topologii sítě a určit počet neuronů v jednotlivých vrstvách. Počet neuronů ve vstupní vrstvě odpovídá počtu vstupních atributů. Skrytá vrstva je sestavena dle nastavení uživatele, dle nastavení je pak počet neuronů určen jako násobky neuronů ve vstupní vrstvě. Počet neuronů ve výstupní vrstvě potom odpovídá počtu tříd, do kterých jsou hodnoty klasifikovány.

Obecně nejsou dána přesná pravidla jak zvolit optimální počet neuronů ve skryté vrstvě. Návrh sítě se často provádí testováním různých nastavení. Navržení sítě však může (pozitivně i negativně) ovlivnit přesnost výsledku, stejně tak jako počáteční nastavení vah.

Učení neuronové sítě

Učení neuronové sítě je iterativní proces opakovaný do té doby, dokud není síť dostatečně naučena. V prvním kroku jsou inicializovány všechny váhy a biasy neuronů náhodnými malými hodnotami z intervalu *<0,1>*. Dále je sestavena množina výstupních hodnot, které zároveň tvoří klasifikační třídy. Tyto hodnoty jsou přepsány do matice tak, aby je bylo možné porovnávat s výslednou konfigurací neuronů ve výstupní vrstvě.

Proces učení neuronové sítě probíhá v následujících krocích:

1. Do neuronů ve vstupní vrstvě jsou načítána data z testovací množiny. Počet neuronů odpovídá počtu atributů n-tice dat.

2. Jsou spočítány hodnoty neuronů ve skryté vrstvě. Pro každý neuron je počítána hodnota podle následujícího vztahu:

$$I_j = \sum_i w_{ij} O_i + \theta_j$$

$$O_j = \frac{1}{1 + e^{-I_j}}$$

3. Stejně jako v předchozím bodě jsou spočteny hodnoty všech neuronů ve výstupní vrstvě.

4. Uvedený postup se provede pro všechny n-tice testovací množiny dat. Poté následuje fáze zpětného šíření chyby, kdy jsou na základě spočtené chyby modifikovány hodnoty vah a biasů. Šíření chyby se provádí od neuronů ve výstupní vrstvě k vrstvě vstupní. Nejprve je určena chyba pro každý neuron ve výstupní vrstvě:

$$Err_k = O_k(1 - O_k)(T_k - O_k),$$

kde *T* je skutečná hodnota.

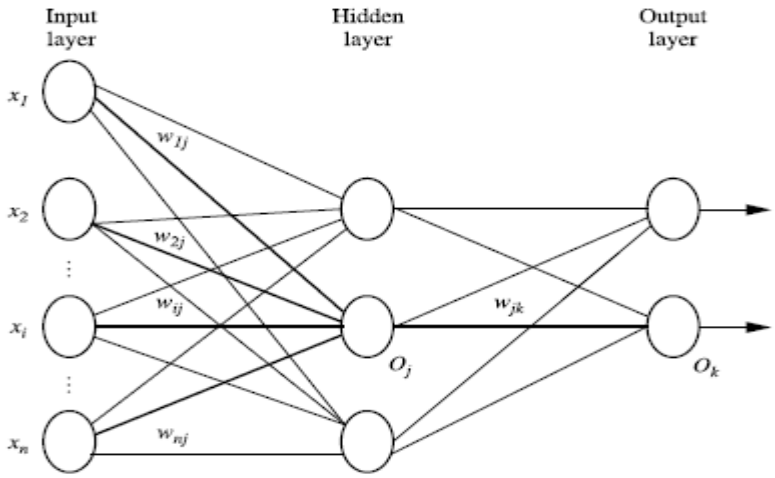
5. Následně je určena chyba neuronů ve skryté vrstvě:

$$Err_j = O_j(1 - O_j) \sum_k Err_k w_{jk}$$

6. Po vypočítání chyb jsou modifikovány váhy a biasy neuronů ve všech vrstvách. Každá váha w_{ij} a každý bias θ_j jsou modifikovány následujícím způsobem:

$$w_{ij} = w_{ij} + L(Err_j O_j),$$
$$\theta_j = \theta_j + L(Err_j),$$
$$L = 1/t,$$

kde *L* je koeficient učení a *t* je iterační krok.



Obrázek 6.2: Schéma neuronové sítě

Klasifikace

Po provedení daného počtu kroků učení je předpokládáno, že neuronová síť je dostatečné naučena. Poté se provede fáze samotné klasifikace. Na vstup neuronů ve vstupní vrstvě jsou nastavena data z testovací množiny. Klasifikace se provádí pomocí stejného šíření vstupu na výstup jako ve fázi učení, nedochází však k porovnávání se skutečnou hodnotou, k šíření chyby a tím i k modifikaci vah a biasů.

Výstup neuronové sítě tvoří vektor *0* a *1*, které jsou následně mapovány na matici možných výstupních hodnot, a tak dojde k určení klasifikační třídy. Průběh klasifikace lze snáze pochopit na následujícím příkladu.

1. Uživatelem je nastaveno následující nastavení – 3 vstupní atributy, 1 výstupní, velikost trénovací a testovací množiny 65%, resp. 35% a bude použit vzorek dat obsahující 20 záznamů.

2. Do matice jsou načtena vstupní data, první tři sloupce obsahují vstupní data, poslední sloupec potom data cílová (skutečné hodnoty). Dále je sestavena množina možných výstupních hodnot (klasifikačních tříd). Tato množina je přepsána do matice obsahující *0* a *1*, aby bylo možné tyto hodnoty porovnávat s výstupem neuronové sítě. Jedná se o jednotkovou matici a pozice hodnoty *1* ve sloupci udává index mapované hodnoty v matici *tSet*.

$$Input = \begin{pmatrix} 4 & 1 & 3 & 3 \\ 4 & 0 & 4 & 2 \\ 4 & 1 & 2 & 3 \\ 4 & 0 & 2 & 4 \\ 3 & 0 & 3 & 3 \\ 3 & 0 & 2 & 1 \\ 3 & 1 & 3 & 3 \\ 4 & 1 & 1 & 4 \\ 4 & 0 & 2 & 4 \\ 4 & 1 & 3 & 4 \\ 4 & 0 & 1 & 4 \\ 4 & 1 & 1 & 4 \\ 3 & 1 & 2 & 2 \end{pmatrix} \qquad tSet = \begin{pmatrix} 1 \\ 2 \\ 3 \\ 4 \end{pmatrix} \qquad tSetBin = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

3. Váhy a biasy všech neuronů jsou inicializovány náhodnými hodnotami z intervalu *<0,1>*. Hodnoty vah i biasů jsou uloženy v maticích. Mějme prvek *a[i,j]* v matici vah, potom *i* udává index neuronu v předchozí vrstvě a *j* index neuronu v aktuální vrstvě.

$$wHidden = \begin{pmatrix} 0{,}55 & -0{,}06 & 0{,}66 \\ 0{,}26 & 0{,}03 & -0{,}82 \\ 0{,}35 & -0{,}43 & -0{,}88 \end{pmatrix}$$

$$wOutput = \begin{pmatrix} 0{,}13 & -0{,}74 & 0{,}10 & 0{,}04 \\ -0{,}59 & 0{,}87 & -0{,}34 & 0{,}49 \\ -0{,}40 & 0{,}98 & -0{,}44 & 0{,}25 \end{pmatrix}$$

$$biasHidden = (-0{,}08 \quad 0{,}37 \quad -0{,}02)$$

$$biasOutput = (0{,}87 \quad 0{,}91 \quad -0{,}14 \quad 0{,}65)$$

4. V iteračních krocích je síť opakovaně přeučována na všech testovacích vzorcích dat. Výstupní hodnota z každého neuronu je upravena aktivační funkcí $y = \frac{1}{1+e^{-x}}$. Po provedení daného počtu iteračních kroků by měla být síť naučena (v případě správného nastavení). Jsou tak finálně modifikovány hodnoty vah a biasů.

$$wHiddenFinal = \begin{pmatrix} 0{,}37 & 0{,}58 & 0{,}34 \\ 0{,}62 & -0{,}70 & 0{,}32 \\ 0{,}79 & -1{,}43 & -0{,}06 \end{pmatrix}$$

$$wOutputFinal = \begin{pmatrix} -1{,}14 & -1{,}22 & 0{,}01 & -0{,}52 \\ 0{,}19 & 0{,}63 & -0{,}74 & 1{,}65 \\ 0{,}16 & -0{,}14 & -0{,}53 & -0{,}36 \end{pmatrix}$$

$$biasHiddenFinal = (0{,}06 \quad 0{,}97 \quad -0{,}50)$$

$$biasOutputFinal = (-1{,}38 \quad -0{,}81 \quad -0{,}13 \quad -0{,}04)$$

5. Nyní je neuronová síť naučena a lze tedy provést klasifikaci. Na vstup neuronů ve vstupní vrstvě jsou přivedena data z testovací množiny. Pomocí nastavených vah a biasů jsou spočítány hodnoty na výstupu neuronů ve výstupní vrstvě, tedy i výstupní hodnoty celé sítě. První čtyři sloupce v matici *result* udávají výstupní hodnoty jednotlivých neuronů ve výstupní vrstvě. V posledním sloupci jsou hodnoty odpovídající tomuto výstupu.

$$result = \begin{pmatrix} 0 & 0 & 0 & 1 & 4 \\ 0 & 0 & 1 & 0 & 3 \\ 0 & 0 & 1 & 0 & 3 \\ 0 & 0 & 0 & 1 & 4 \\ 0 & 0 & 0 & 1 & 4 \\ 0 & 0 & 1 & 0 & 3 \\ 0 & 0 & 0 & 1 & 4 \end{pmatrix}$$

6. Výsledná data jsou porovnána se skutečnými hodnotami a lze vidět, s jakou přesností byla klasifikace provedena. V matici *test* první tři sloupce udávají vstupní atributy, další sloupec potom skutečnou hodnotu a v posledním sloupci jsou hodnoty na výstupu neuronové sítě. V této testovací úloze byly tedy správně klasifikovány tři ze sedmi n-tic.

$$test = \begin{pmatrix} 4 & 1 & 3 & 3 & 4 \\ 2 & 1 & 3 & 1 & 3 \\ 3 & 1 & 3 & 3 & 3 \\ 4 & 1 & 1 & 4 & 4 \\ 4 & 1 & 3 & 4 & 4 \\ 3 & 1 & 3 & 2 & 3 \\ 4 & 1 & 2 & 3 & 4 \end{pmatrix}$$

Normalizace

V některých úlohách a při použití některých metod (např. neuronové sítě) je vhodné použít normalizaci vstupních hodnot. Tato úprava může urychlit proces učení. Volba použití normalizace byla ponechána na uživateli a je možné ji zvolit při konfiguraci dolovací úlohy.

Normalizace se provádí tak, že vstupní hodnoty jsou upraveny tak, aby po úpravě náležely do intervalu <-1.0,1.0>. Pro implementaci byla zvolena min-max normalizace prováděna pomocí následujícího vztahu

$$v' = \frac{v - minA}{\max A - minA}(new_{minA} - new_{maxA}) + new_{minA},$$

kde hodnota *v* atributu *A* je mapována na hodnotu *v'*, *minA* a *maxA* jsou minimální a maximální hodnoty atributu *A* a new_{minA} *a* new_{maxA} jsou nové minimum a maximum, v tomto případě hodnoty -1.0 a 1.0.

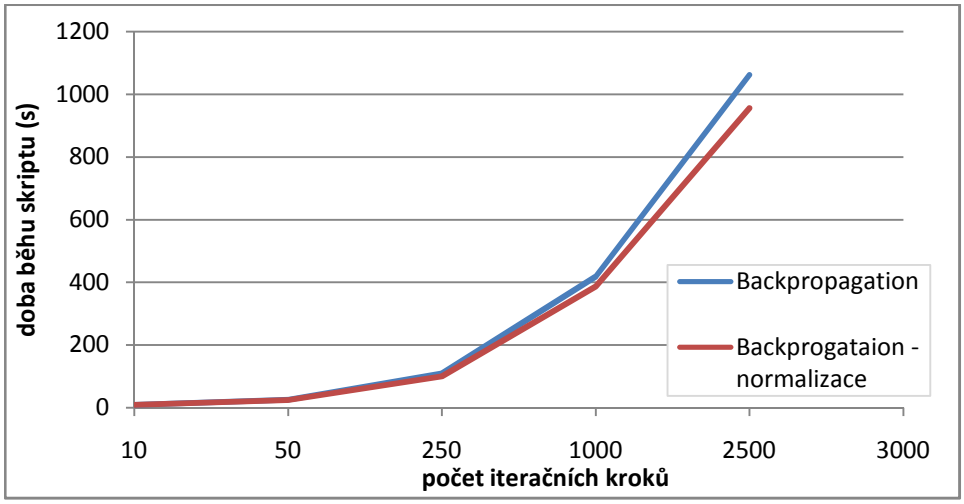
Jakým způsobem dokáže normalizace urychlit proces učení, bylo testováno na několika úlohách. Zde je uvedena jedna z nich:

Bylo použito následující nastavení – neuronová síť, 1400 záznamů rozdělených na trénovací (55%) a testovací (45%) množinu. Postupně byl zvětšován počet iteračních kroků ve fázi učení a byla zaznamenávána délka trvání výpočtu. Měření bylo prováděno v době, kdy použitý server byl minimálně vytížen, aby nedošlo k výraznému zkreslení výsledků. Získané výsledky jsou uvedeny v

následující tabulce, která zachycuje dobu běhu skriptu v sekundách ve dvou případech – když jsou vstupní data normalizována a bez normalizace.

Počet kroků	Bez normalizace	Normalizace dat
10	8,67	8,60
50	24,90	24,14
250	109,27	99,46
1000	418,54	386,34
2500	1061,83	955,97

Z tabulky i z grafické reprezentace naměřených údajů je patrné, že při nižším počtu iteračních kroků není rozdíl nijak výrazný. Se zvětšujícím se počtem kroků se však tento rozdíl zvětšuje a při použití normalizace vstupních dat může být doba výpočtu i o 10% kratší.



Obrázek 6.3: graf zachycující závislost délky trvání výpočtu na počtu iteračních kroků

6.1.2.3 Export dat

Export dat je řešen pomocí modifikované třídy volně dostupné na http://www.appservnetwork.com/modules.php?name=News&file=article&sid=8. Výsledná data dolovacích úloh je možné vyexportovat ve formátu XLS. V nastavení je možné zvolit, zda budou data exportována ve stejné podobě (ve formě tabulky obsahující vstupní data a porovnání skutečných a predikovaných hodnot) nebo v upravené formě vhodné pro vytvoření grafu.

6.1.3 Front-end aplikace

Front-end aplikace je vytvořen v jazyce XHTML dle norem W3C (www.w3.org). Při tvorbě uživatelského rozhraní byl kladen důraz na použitelnost aplikace a na snadné a intuitivní ovládání.

Část aplikační logiky (například kontrola formulářů) byla přesunuta na stranu klienta, z toho důvodu je při vstupu kontrolována podpora klientského skriptování. V případě, že uživatel nemá zapnutu podporu jazyka Javascript, není mu umožněno aplikaci využívat.

Pro tvorbu uživatelského rozhraní byl použit javascriptový framework ExtJS, který je volně dostupný pro nekomerční využití. Použitý framework umožňuje tvorbu efektního i efektivního uživatelského rozhraní a i přes svou robustnost je poměrně snadno modifikovatelný. Při implementaci se však vyskytly i problémy, například načítání většího množství dat do výsledné tabulky je poměrně dlouhé a spotřebovává značnou část výkonu procesoru.

6.1.4 Instalace a konfigurace modulu

Nasazení modulu k použití je snadné – stačí nahrát veškeré skripty na server podporující jazyk PHP, nejlépe ve verzi 5, a nastavit konfigurační soubor. Soubor s konfigurací (*configs/config.php*) obsahuje následující nastavení:

- připojení do databáze – dolovací modul importuje data z databázových tabulek, implicitně je připraven na práci s databázovým systémem PostgreSQL.
- konstanta *PATH* – obsahuje absolutní cestu k adresáři se skripty. Tato konstanta je potřebná pro export dat, kdy před samotným exportem jsou data uložena do souboru. Dočasné soubory jsou uloženy ve složce *export*, která musí mít nastavena práva pro zápis skupiny, no níž náleží webový server.
- volitelně je možno nastavit předdefinované dolovací úlohy. Toto nastavení se provádí v souboru *configs/configMining.js*.

6.1.5 Testování

Součástí této práce je i provedení různých dolovacích úloh, porovnání s výsledky provedenými v aplikaci SAS Enterprise Miner a jejich zhodnocení.

6.1.5.1 Odhad služby na základě obratu, počtu zaměstnanců a faktury

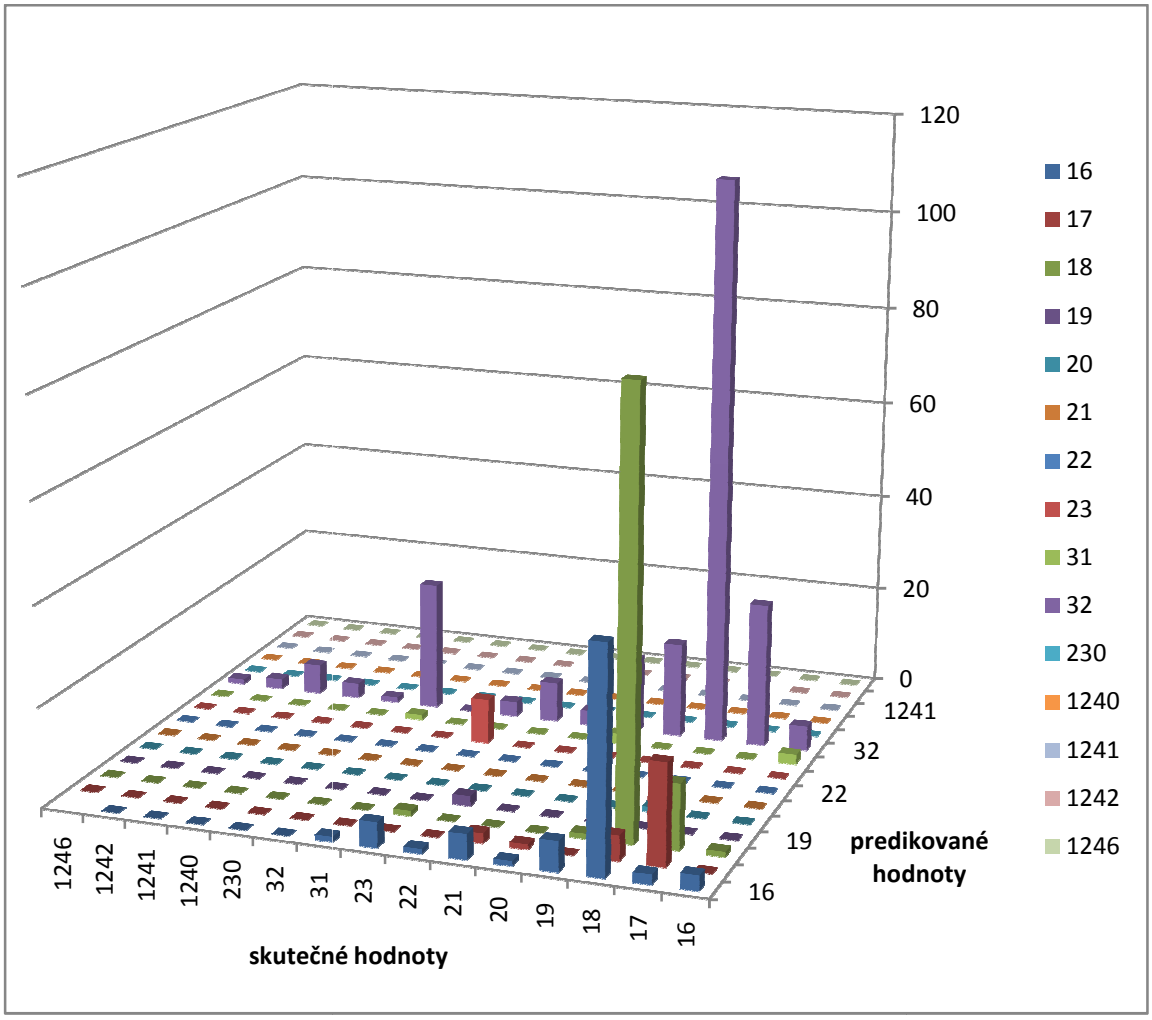
Při řešení této dolovací úlohy bylo použito stejné nastavení jako v prostředí SAS Enterprise Miner a byly použity metody vícenásobné lineární regrese a neuronové sítě. Cílem úlohy je pokusit se nalézt vztah mezi prodávanými službami a atributy firem v databázi, jako je počet zaměstnanců a roční obrat firmy.

Nastavení dolovacího modulu:

1. vstupní atributy: obrat, invoice_amount, pocet_zamestancu
2. cílový atribut: sluzba_id
3. počet záznamů: 1000
4. rozdělení trénovací a testovací množiny: 55% a 45%

Regrese

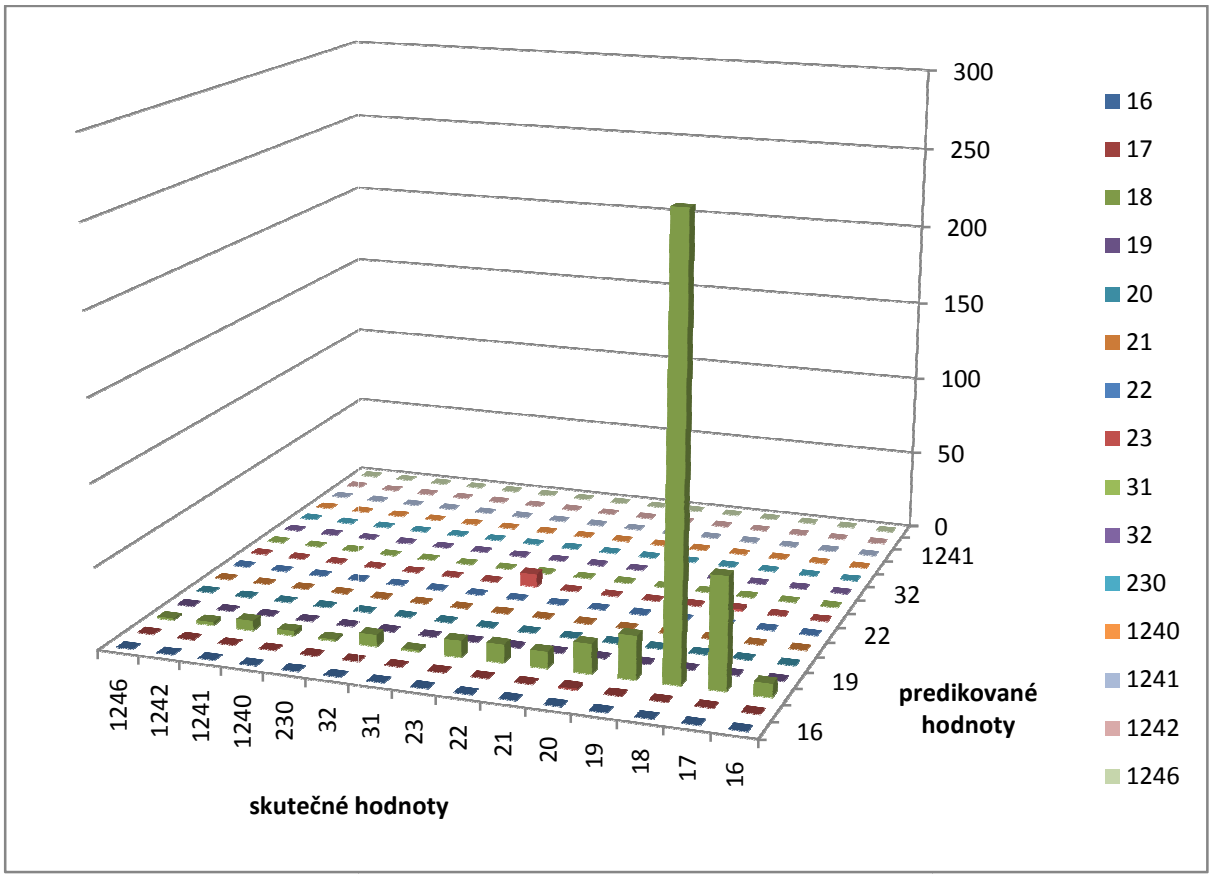
Na obrázku 6.4 je zachycen graf úspěšnosti predikce cílového atributu, v tomto případě id služby. Jak je patrné z grafu, predikce pomocí lineární regrese není příliš úspěšná. Stejně jako v případě použití aplikace SAS Enterprise Miner (obrázek 5.5) je často predikována služba s identifikátorem 18, jež se často vyskytuje ve zvoleném vzorku dat. Při použití implementovaného modulu je však tato služba predikována správně, ale chybně je často predikována služba s identifikátorem 32.



Obrázek 6.4: Graf úspěšnosti predikce id služby pomocí regrese

Neuronová síť

Stejný test byl proveden také pomocí neuronové sítě, jeho výsledky jsou zachyceny na obrázku 6.5. Při porovnání s grafem 5.7 je patrné, že bylo dosaženo velmi podobných, byť neúspěšných, výsledků. V této úloze se neuronová síť jeví jako nepoužitelná, jelikož takřka všechny hodnoty jsou zařazeny do stejné třídy.



Obrázek 6.5: Graf úspěšnosti predikce id služby pomocí neuronové sítě

6.1.5.2 Odhad služby na základě obchodníka a faktury

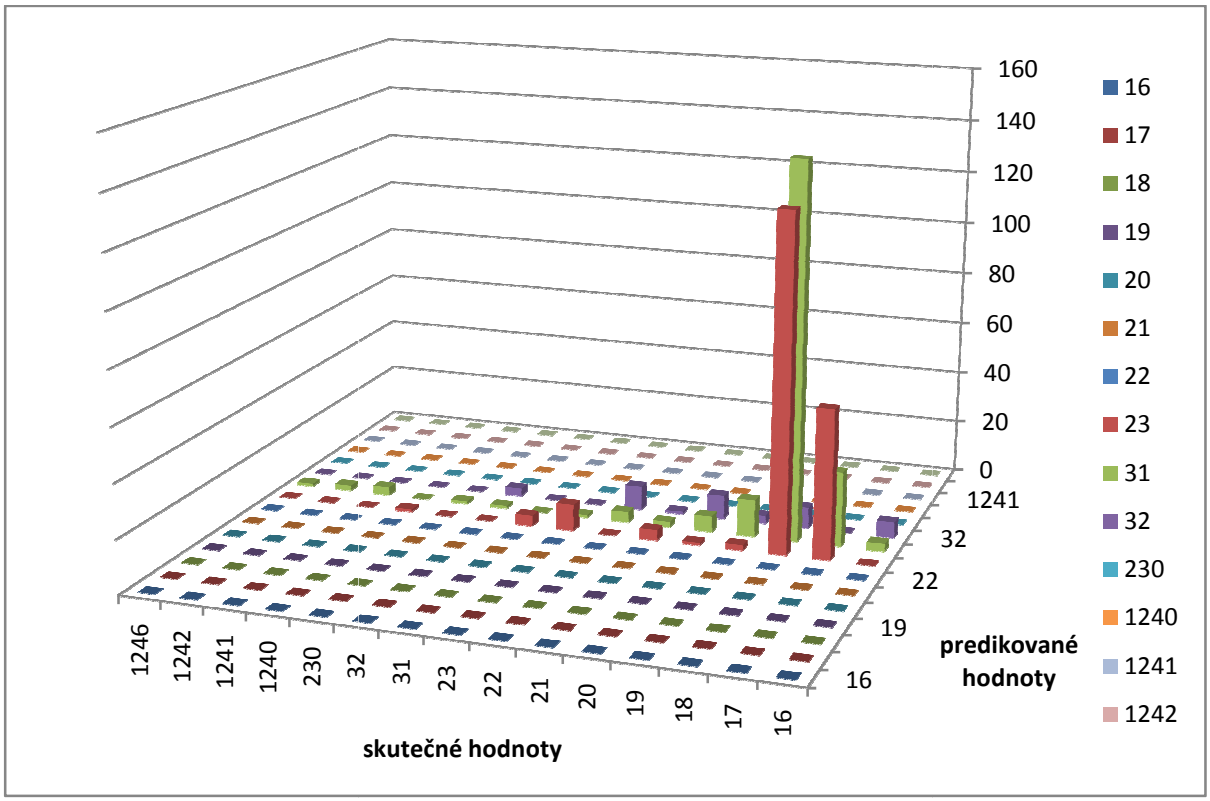
V další úloze byla predikována služba na základě obchodníka (prodejce) a faktury firmy za telefonní služby.

Pro test bylo použito následující nastavení:

1. vstupní atributy: obchodnik_id, invoice_amount
2. cílový atribut: sluzba_id
3. počet záznamů: 1000
4. rozdělení trénovací a testovací množiny: 55% a 45%

Regrese

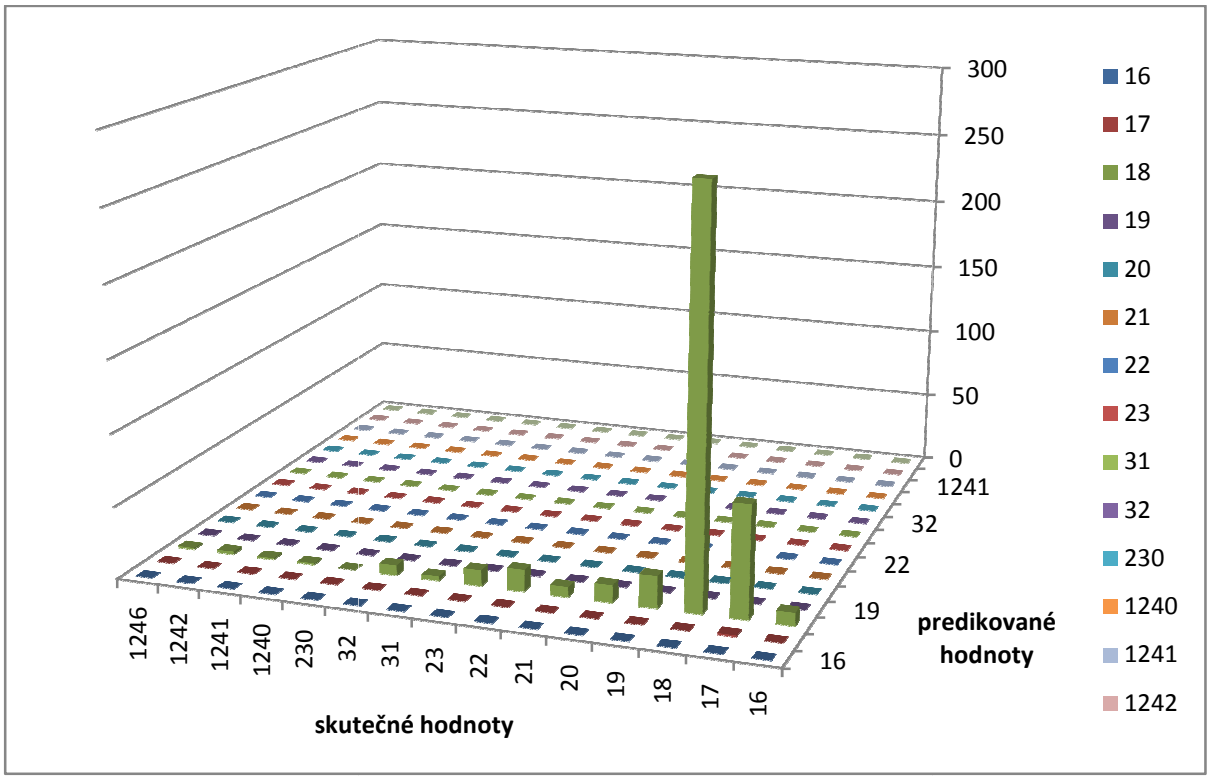
dle získaných výsledků je zřejmé, že lineární metoda není vhodná pro tuto dolovací úloha. Takřka všechny hodnoty zachycené v grafu 6.6 leží mimo diagonálu, a tedy byly predikovány chybně. Oproti výsledkům získaným pomocí aplikace SAS Enterprise Miner se nepodařilo ani správně predikovat nejčastěji se vyskytující hodnotu 18.



Obrázek 6.6: Graf úspěšnosti predikce id služby pomocí regrese

Neuronová síť

Pomocí neuronové sítě bylo dosaženo podobných výsledků jako při testování pomocí SAS Enterprise Miner. Takřka ve všech případech byly hodnoty klasifikovány do nejčastěji se vyskytující třídy 18. Zvolené metody tedy pro tuto úlohu nejsou vhodné nebo použitá data nejsou vhodná pro dolování.



Obrázek 6.7: Graf úspěšnosti predikce id služby pomocí neuronové sítě

6.1.5.3 Zhodnocení testování na datech z IS Belinda

V rámci této práce bylo provedeno testování na datech z databáze firmy Voxnet s.r.o., například predikce služby (uvedeno v předchozích podkapitolách), úspěšnosti hovorů pracovníků call centra, predikce tržeb atd. Jak je patrné z uvedených testů, úspěšnost predikce není nijak vysoká. Jelikož bylo pomocí vytvořeného modulu dosaženo obdobných výsledků jako pomocí aplikace SAS Enterprise Miner, usuzuji, že použitá data nejsou příliš vhodná pro dolování. Proto byla funkčnost implementovaného modulu ověřena i pomocí jiných dat.

6.1.5.4 Testování na antikoncepčních datech

Pro otestování funkčnosti implementovaného modulu byly použity statistická data pocházející z průzkumu používání antikoncepčních metod v Indonésii provedeného v roce 1987.

Data obsahují následující atributy: věk ženy, vzdělání ženy, vzdělání manžela, počet dětí, zda je žena věřící, zda je žena zaměstnaná, zda je manžel zaměstnaný, index životního standardu, zda je žena ovlivňována médii a zda používá antikoncepční metodu.

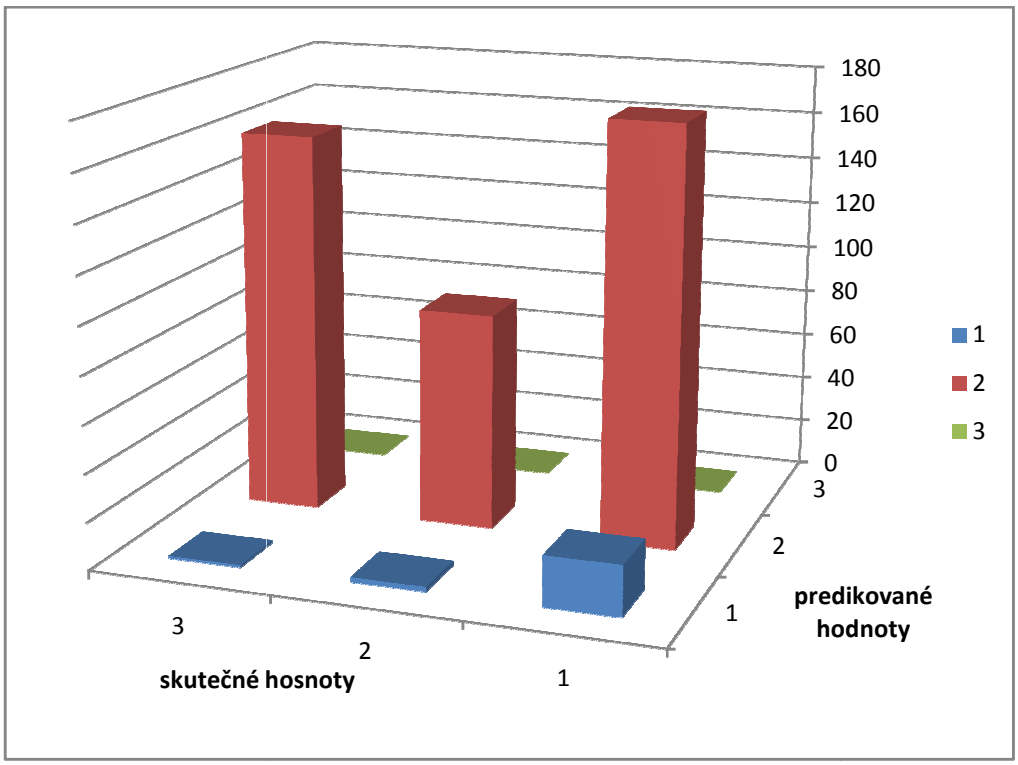
Pro test bylo použito následující nastavení:

1. vstupní atributy: vzdělání ženy, věřící, zaměstnaná, životní standard, ovlivnění médii
2. cílový atribut: antikoncepční metoda
3. počet záznamů: 1000
4. rozdělení trénovací a testovací množiny: 55% a 45%

Cílem úlohy je zjistit, zda ženy používají antikoncepci v závislosti na zvolených atributech. Zejména jakým způsobem ženu ovlivňuje náboženství (v tomto případě islám) a jiné atributy při volbě antikoncepce. Data byla klasifikována do 3 tříd – nepoužívají antikoncepci (1), používají dlouhodobé (2) a krátkodobé metody (3).

Regrese

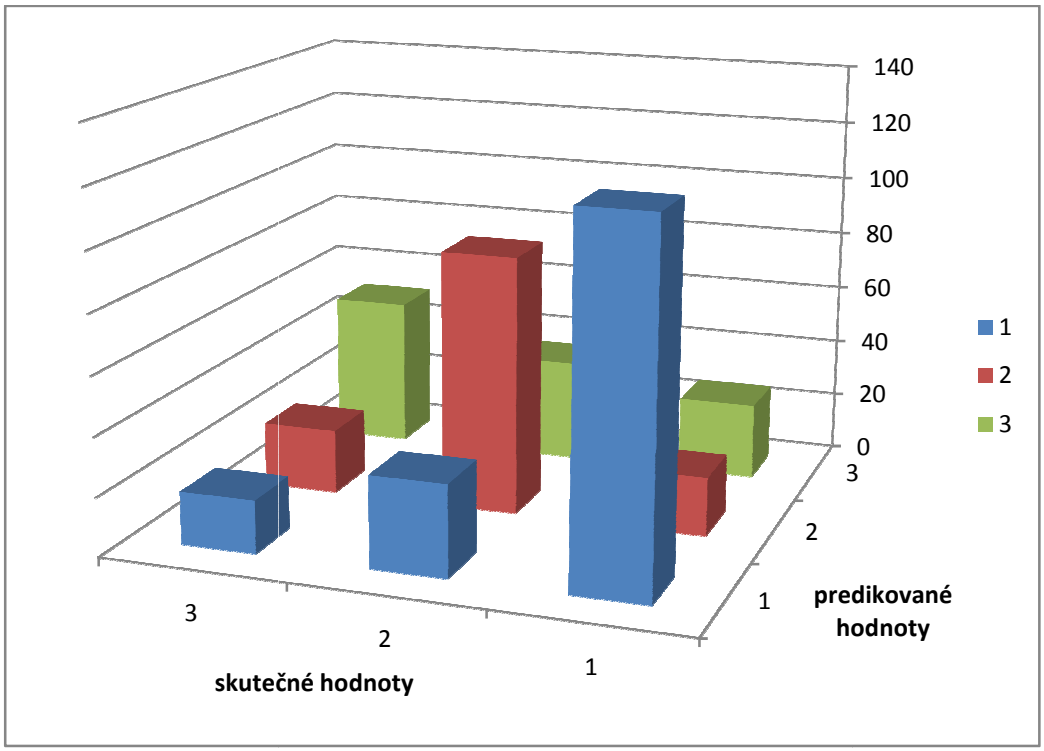
Z grafu predikce pomocí regrese je patrné, že tato metoda není vhodná pro danou dolovací úlohu. Až na výjimka byla predikována hodnota 2.



Obrázek 6.8: Graf úspěšnosti predikce typu antikoncepční metody pomocí regrese

Neuronová síť

Pomocí neuronové sítě bylo dosaženo výrazně lepších výsledků. Jak je znázorněno v grafu 6.9, většina hodnot leží na diagonále grafu, takže klasifikace do daných tříd byla poměrně úspěšná. Ze získaných výsledků je zřejmé, že nejvíce žen nevyužívá žádnou antikoncepční metodu.



Obrázek 6.9: Graf úspěšnosti predikce typu antikoncepční metody pomocí neuronové sítě

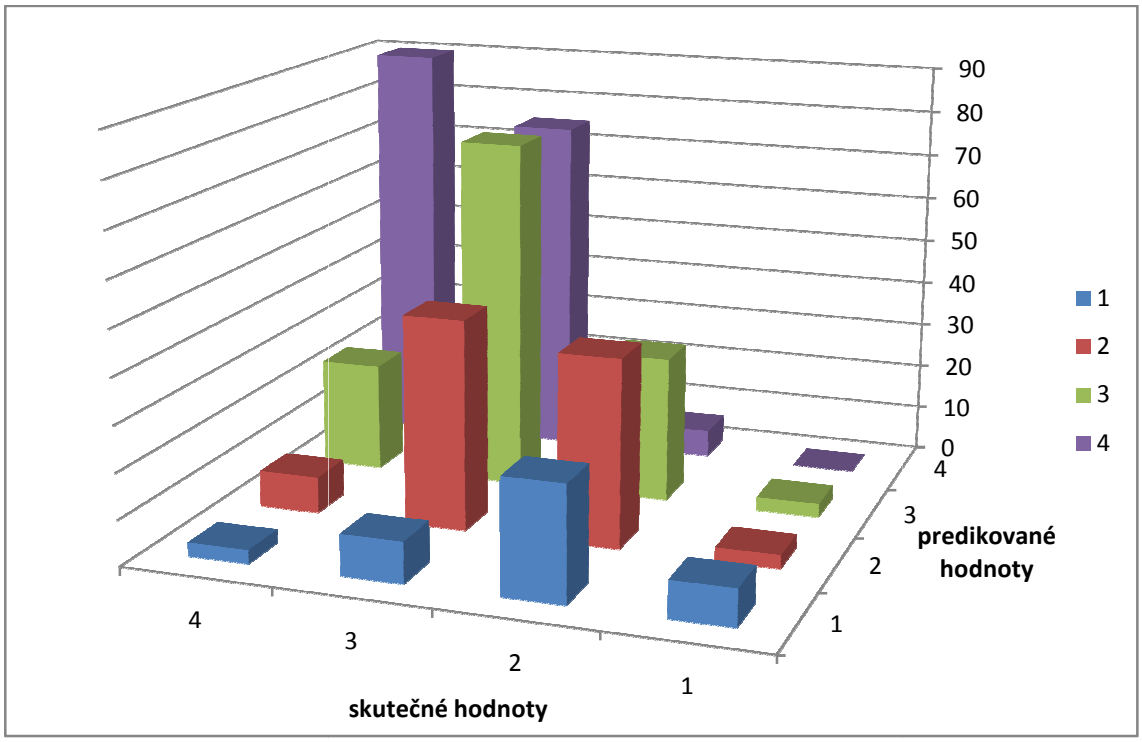
Na zvolených datech z průzkumu používání antikoncepčních metod byla provedena ještě úloha, jejímž cílem bylo predikovat typ vzdělání ženy na základě vzdělání a zaměstnání manžela. Typ vzdělání i zaměstnání je určen pomocí hodnot 1-4, kdy 1 značí nejnižší.

Nastavení dolovacího modulu:

1. vstupní atributy: vzdělání manžela, zaměstnání manžela
2. cílový atribut: vzdělání ženy
3. počet záznamů: 1000
4. rozdělení trénovací a testovací množiny: 55% a 45%

Regrese

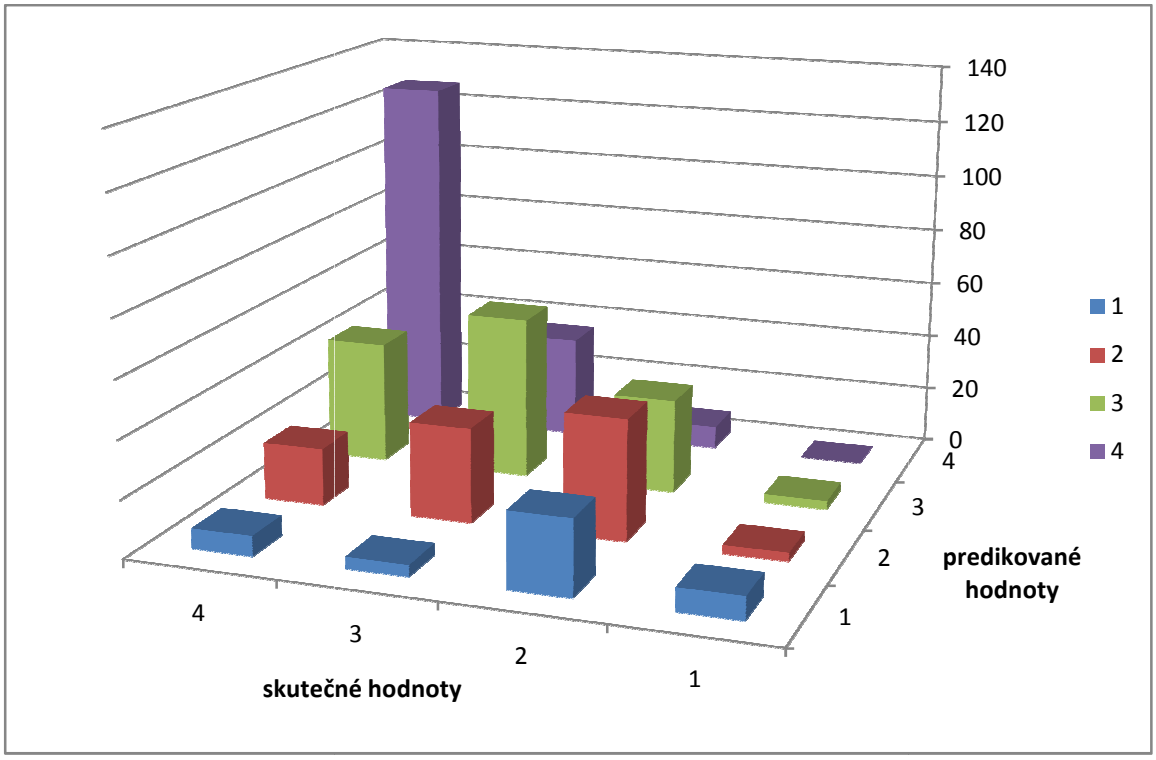
V této metodě se lineární regrese ukázala jako použitelná metoda, avšak jak je patrné z grafu 6.10, velké množství hodnot leží mimo diagonálu a tedy nebyly správně predikovány, ale odchylka od správné hodnoty byla minimální.



Obrázek 6.10: Graf úspěšnosti predikce typu vzdělání pomocí regrese

Neuronová síť

Pomocí neuronové sítě bylo dosaženo nepatrně lepších výsledků, které jsou zachyceny v grafu 6.11.



Obrázek 6.11: Graf úspěšnosti predikce typu vzdělání pomocí neuronové sítě

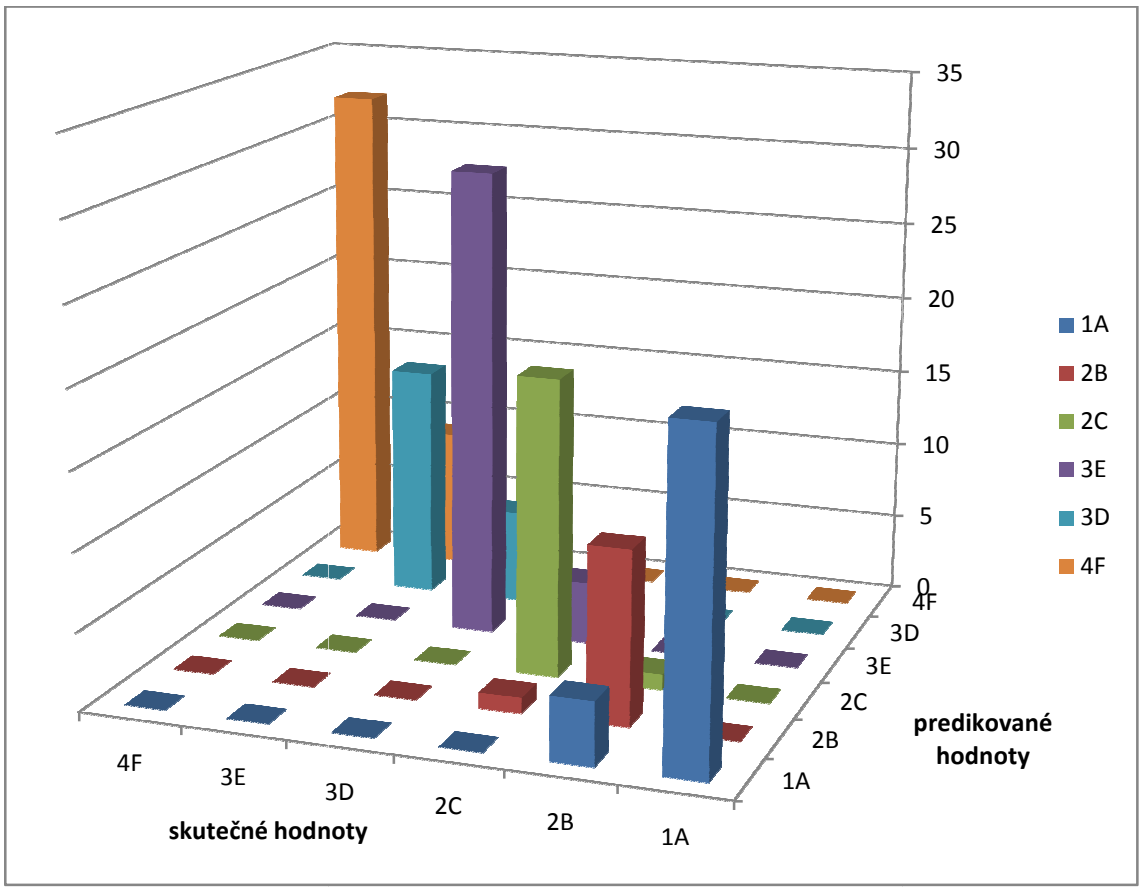
Při pohledu na testovací data je také patrné, že nejčastěji si lidé z průzkumu volí partnery se stejným stupněm vzdělání.

6.1.5.5 Testování na datech z předmětu IFJ

Pro poslední testovací úlohy byly použity data z předmětu Formální jazyky a překladače (IFJ) vyučovaného na FIT VUT. Tyto data obsahují získané body studentů během semestru, konkrétně body za projekt, půlsemestrální a semestrální zkoušku, udělený zápočet, celkový počet bodů a výslednou známku.

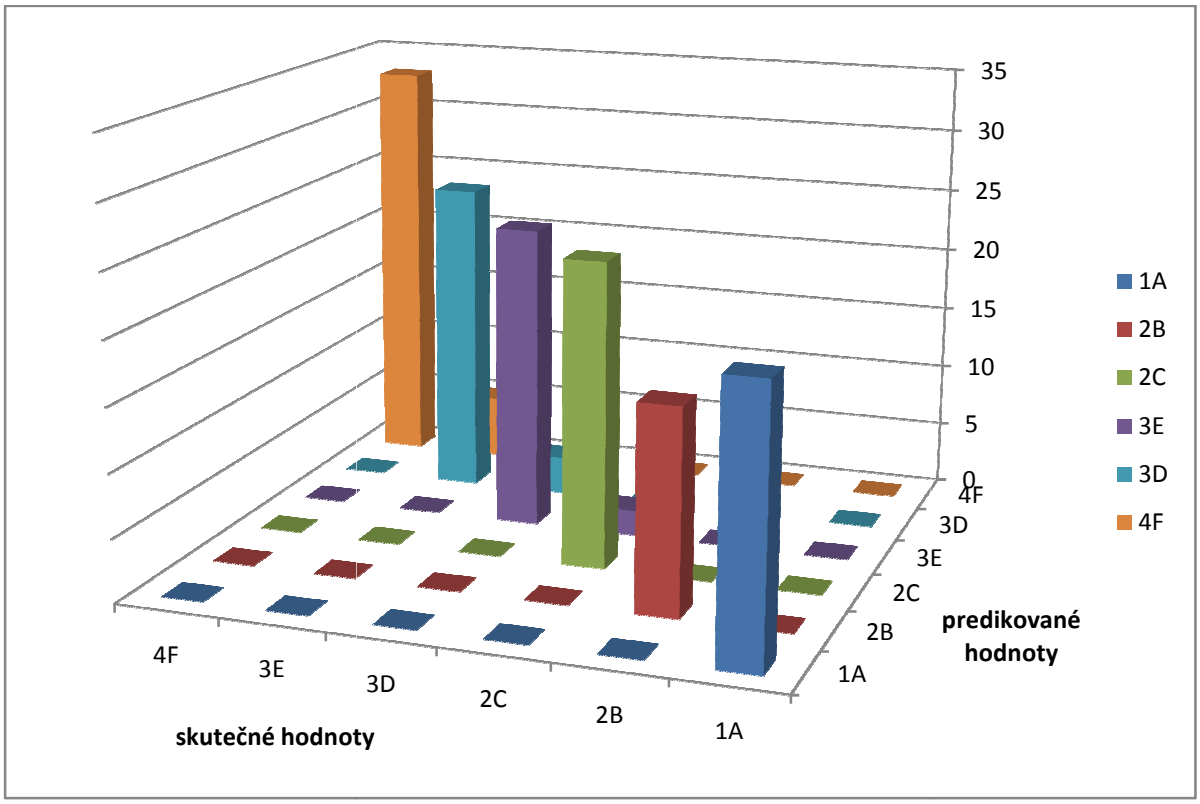
Nejprve bylo testováno, zda se podaří na základě získaných bodů predikovat výslednou známku. Jako vstupní hodnoty byly použity body za projekt a semestrální zkoušku, body za půlsemestrální zkoušku zůstaly skryty.

Pomocí lineární regrese byly získány poměrně dobré výsledky (obrázek 6.12), protože mezi zvolenými daty je lineární závislost. Horší výsledky byly získány pomocí neuronové sítě, kdy tato metoda je pro tento typ úlohy nepoužitelná. Data byla klasifikována pouze do dvou tříd.



Obrázek 6.12: Graf úspěšnosti predikce výsledné známky pomocí regrese

Následně byla metoda lineární regrese otestována na úloze, kdy na vstup byly zavedeny všechny atributy reprezentující získané body, tedy projekt, půlsemestrální a semestrální zkouška. Cílem úlohy je predikovat známku, která v tomto případě reprezentuje součet všech bodů a v optimálním případě by úspěšnost predikce měla být stoprocentní. Jak je patrné z obrázku 6.13, dosažené výsledky obsahují pouze minimální chybu predikce.



Obrázek 6.13: Graf úspěšnosti predikce výsledné známky pomocí regrese

7 Závěr

V rámci této práce byl vytvořen webový modul pro získávání znalostí z databází, který umožňuje analyzovat data pomocí lineární regrese a neuronové sítě Backpropagation. Modul je vytvořen tak, aby jej bylo možné jednoduše napojit na libovolnou databázi využívající PostgreSQL. Pro potřeby této práce je modul napojen na informační systém Belinda firmy Voxnet s.r.o.

Implementovaný modul byl otestován na několika dolovacích úlohách a výsledky byly porovnány s výstupy z aplikace SAS Enterprise Miner. Vytvořený dolovací modul je v praxi možné využít, ale pouze pro omezené typy dolovacích úloh. Pro lepší využitelnost by bylo třeba tento modul výrazně rozšířit o další dolovací metody a různé optimalizační algoritmy a vytvořit tak velmi robustní řešení, které by bylo obdobou některé z komerčních aplikací využívaných pro získávání znalostí z databází.

Hlavní výhody i nevýhody vyplývají z toho, že modul byl implementován jako on-line webová aplikace. Prostředí webu bylo pro implementaci zvoleno záměrně v souladu se současnými trendy přesouvání klasických desktopových aplikací do on-line verzí (kancelářské aplikace, grafické nástroje apod.). Další motivací pro toto rozhodnutí byl fakt, že nejznámější komerční aplikace pro dolování z dat jsou nabízeny pouze jako desktopové aplikace. Výhodou aplikace je její snadné nasazení na jakýkoliv webový informační systém a také její dostupnost, pro práci s dolovacím modulem stačí pouze internetový prohlížeč. Hlavní nevýhodou je časová i hardwarová náročnost při výpočtu některých dolovacích úloh, kdy v některých případech výpočty trvají i několik minut.

Literatura

[HAN2006] Han, J., Kamber, M.: *Data Mining: Concepts and Techniques. Second Edition.* Elsevier Inc., 2006, 770 p., ISBN 1-55860-901-3

[DM99] Půlpán, J.: *Dobývání znalostí z databází 99* [online], poslední modifikace 23.8.1999, [cit. 2007-12-21]. Dostupné na URL: < http://badame.vse.cz/dzd99/anotace-sas.html >

[DMGUIDE] Data-Mining-Guide.net: *Data Mining Overview* [online], [cit. 2007-12-27]. Dostupné na URL: <http://www.data-mining-guide.net/Data-Mining-Overview.html>

[SSCR] StatSoft CR s.r.o.: *STATISTICA a Data mining* [online], [cit. 2007-12-31]. Dostupné na URL: < http://www.statsoft.cz/page/index2.php?pg=navigace&nav=9>

[DMSOFT] Data-Mining-Software.com: *Data Mining History* [online], [cit. 2007-12-31]. Dostupuné na URL: < http://www.data-mining-software.com/data_mining_history.htm>

[BUSSW] IDG Czech, a.s.: *Bioinformatika – na půl cestě mezi algoritmy a životem* [online], [cit. 2007-12-31]. Dostupné na URL: <http://www.businessworld.cz/bw.nsf/ID/bioinformatikaI>

[SCART] INCOMA Consult: *Analýza nákupního košíku jako základ pro optimalizaci promocí* [online], [cit. 2007-12-31]. Dostupné na URL:
< http://www.marketingovenoviny.cz/index.php3?Action=View&ARTICLE_ID=301>

[KORDEF] Wikipedie: *Korelace* [online], [cit. 2007-05-01]. Dostupné na URL:
< http://cs.wikipedia.org/wiki/Korelace>

[ASOC] Burda, M.: *Získávání znalostí z databází - Asociační pravidla* [online], [cit. 2007-05-03]. Dostupné na URL: < http://www.cs.vsb.cz/burda/bibl/zzd/zzd.pdf>

[STECH] Husek, P.: *Neuronové sítě a principy umělé inteligence* [online], [cit. 2007-05-03]. Dostupné na URL: <http://www.stech.cz/articles_print.asp?idk=97&ida=132>

[ADA] Zbořil, F.: *Biologický a umělý neuron,neuronové sítě. Perceptron,Adaline, Madaline a BP (BackPropagation).* [online], [cit. 2007-05-05]. Dostupné na URL:
<https://www.fit.vutbr.cz/study/courses/SFC/private/07sfc_2.pdf>

[INV] Vondrák, V.: *Lineární algebra.* [online], [cit. 2007-05-10]. Dostupné na URL:
<http://vondrak.am.vsb.cz/LA-IT/la1.pdf>

[KR] Wikipedie: *Kvadratická regrese.* [online], [cit. 2007-05-13]. Dostupné na URL:
<http://cs.wikipedia.org/wiki/Kvadratick%C3%A1_regrese>

[PR] Wikipedie: *Polynomická regrese.* [online], [cit. 2007-05-13]. Dostupné na URL:
<http://cs.wikipedia.org/wiki/Polynomick%C3%A9_regrese>

[SVM] Sherrod, P.: *SVM - Support Vector Machines.* [online], [cit. 2007-05-13]. Dostupné na URL:
<http://www.dtreg.com/svm.htm>

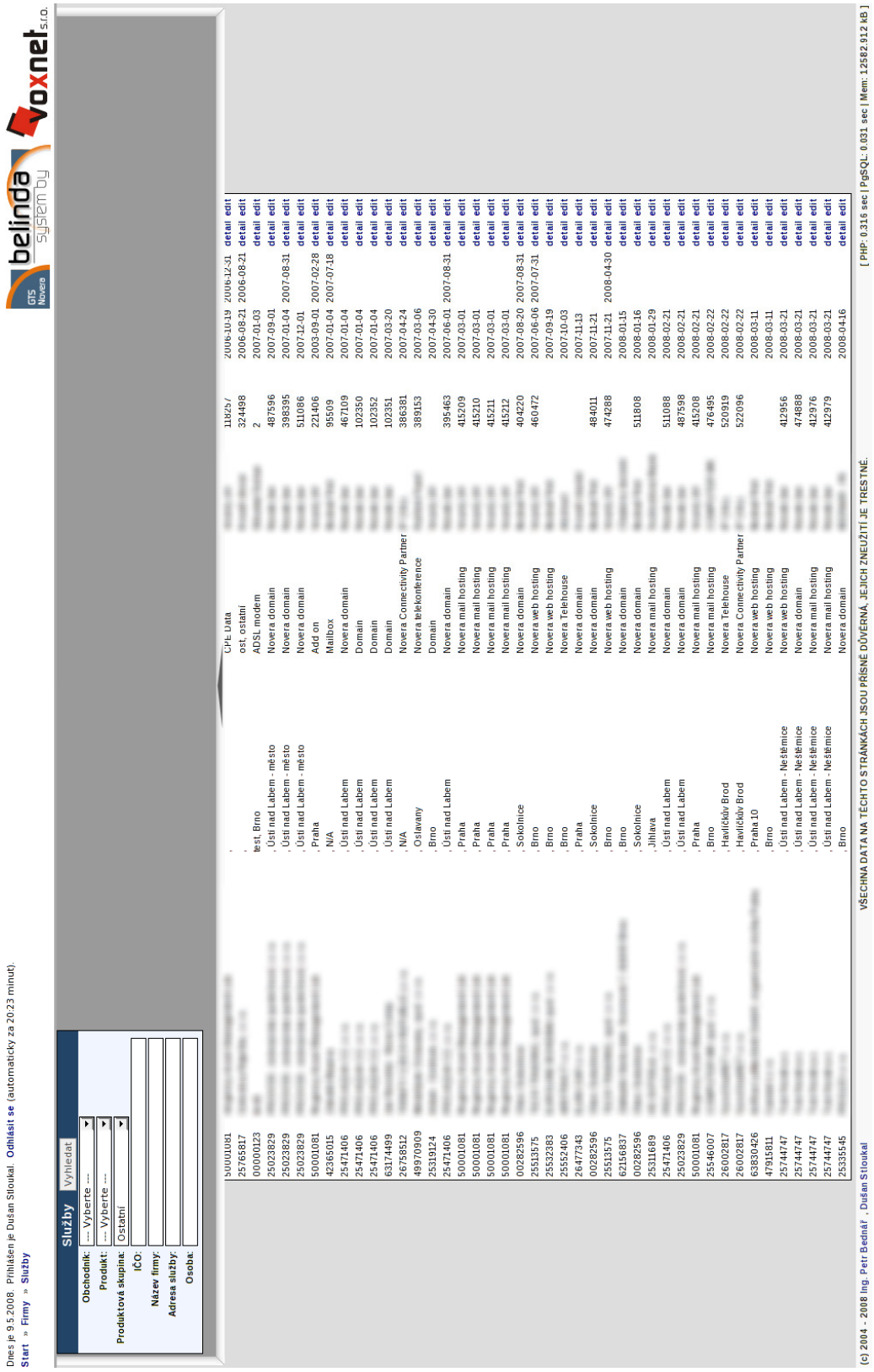
[KME] Pošík, P.: *Segmentace a shlukování.* [online], [cit. 2007-05-13]. Dostupné na URL:

<http://cyber.felk.cvut.cz/gerstner/teaching/zbd/DataMining4-hout.pdf>

[CME] Zeman, D.: *Aplikace procesu dolování dat v biologie - genetice.* [online], [cit. 2007-05-13]. Dostupné na URL: <http://www.fit.vutbr.cz/study/courses/VPD/public/0405VPD-Zeman.pdf>

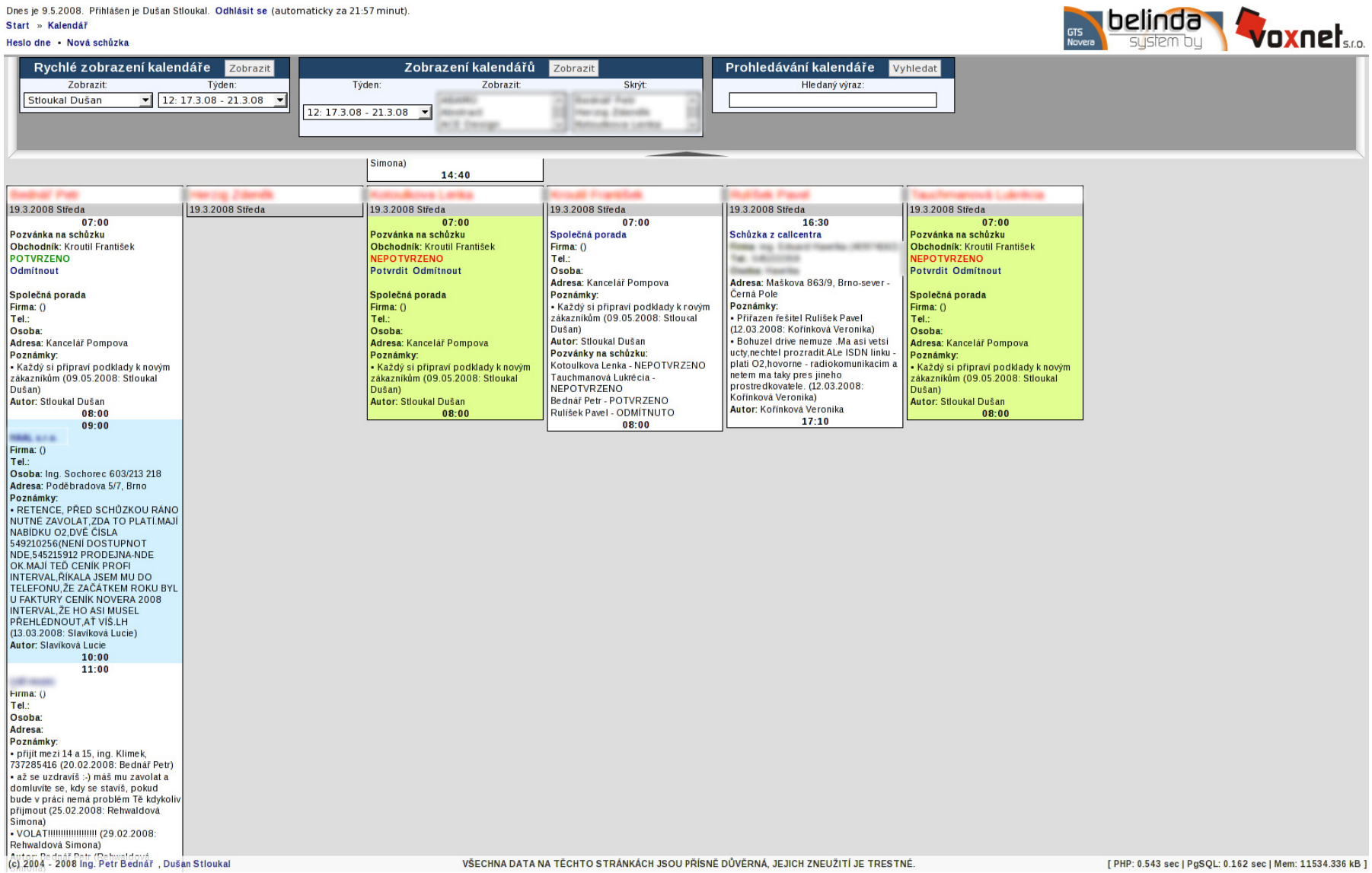
[M3D] Miner 3D, Inc..: *K-Means Clustering, Visual Clustering.* [online], [cit. 2007-05-13]. Dostupné na URL: <http://miner3d.com/products/kmeans_clustering.html>

Přílohy

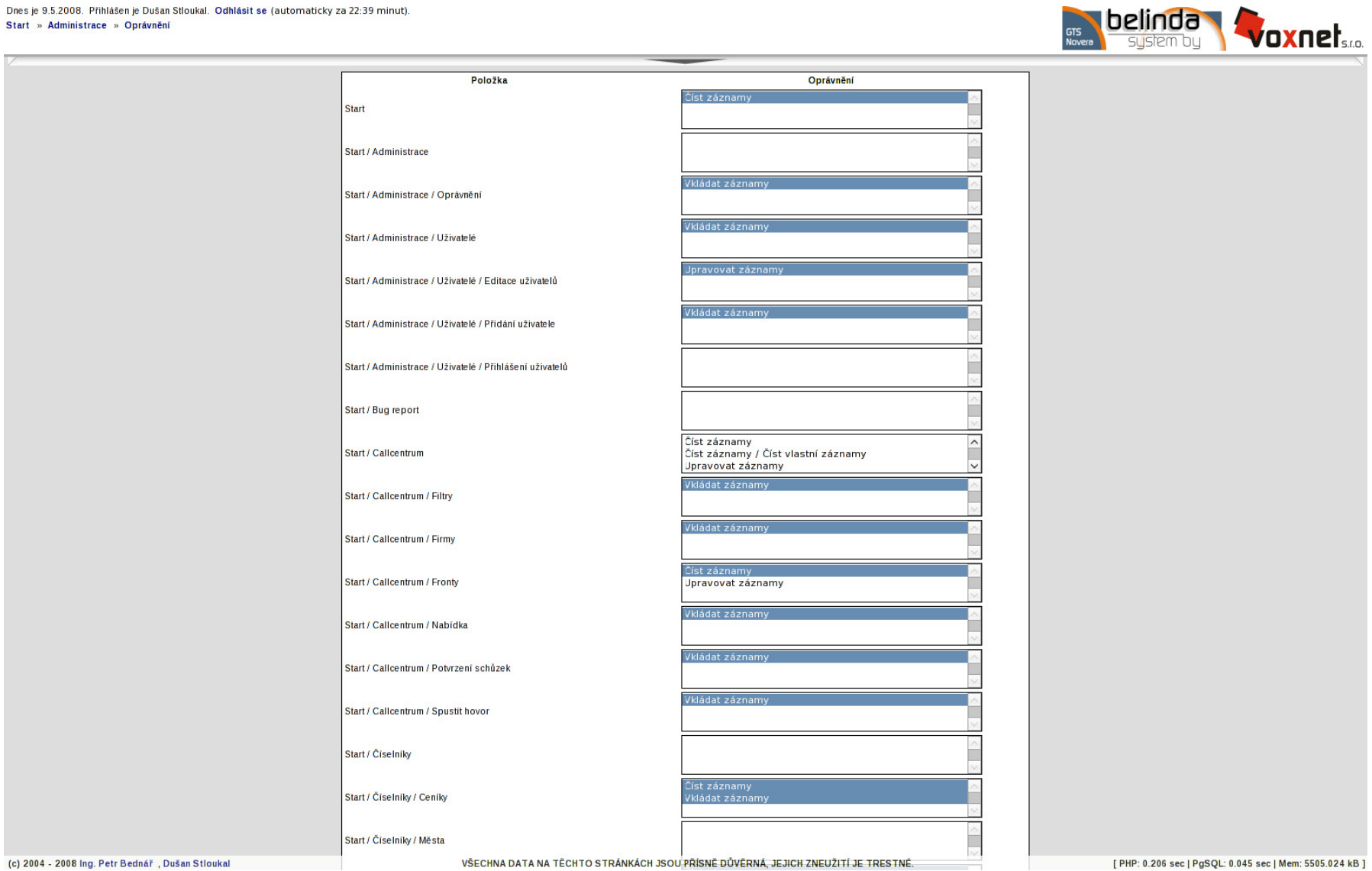


Obrázek P.1: Náhled obrazovky Informačního systému Belinda – vyhledávání služeb

Obrázek P.2: Náhled obrazovky Informačního systému Belinda – fakturační historie



Obrázek P.3: Náhled obrazovky Informačního systému Belinda – kalendář



Obrázek P.4: Náhled obrazovky Informačního systému Belinda – správa oprávnění

Obrázek P.5: Náhled obrazovky Informačního systému Belinda – úvodní strana

Obrázek P.6: Náhled obrazovky Informačního systému Belinda – tickety

Obrázek P.7: Náhled obrazovky Informačního systému Belinda – filtry

Obrázek P.8: Náhled obrazovky Informačního systému Belinda – produkty

Obrázek P.9: Náhled obrazovky Informačního systému Belinda – sklad