TECHNICKÁ UNIVERZITA V LIBERCI
Fakulta mechatroniky, informatiky
a mezioborových studií

KLASIFIKAČNÍ ALGORITMY V DATAMININGOVÝCH ÚLOHÁCH

Bakalářská práce

Studijní program: B2646 - Informační technologie
Studijní obor: 1802R007 - Informační technologie

Autor práce: **Petr Franz**
Vedoucí práce: RNDr. Klára Císařová, Ph.D.

TECHNICAL UNIVERSITY OF LIBEREC
Faculty of Mechatronics, Informatics
and Interdisciplinary Studies

CLASSIFICATION ALGORITHMS IN DATAMINING

Bachelor Thesis

Study programme: B2646 - Information Technology

Study branch: 1802R007 - Information Technology

Author: **Petr Franz**

Supervisor: RNDr. Klára Císařová, Ph.D.

TECHNICKÁ UNIVERZITA V LIBERCI
Fakulta mechatroniky, informatiky a mezioborových studií
Akademický rok: **2015/2016**

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**Petr Franz**
Osobní číslo:	**M12000127**
Studijní program:	**B2646 Informační technologie**
Studijní obor:	**Informační technologie**
Název tématu:	**Klasifikační algoritmy v dataminingových úlohách**
Zadávající katedra:	**Ústav mechatroniky a technické informatiky**

Zásady pro vypracování:

1. Seznamte se s dataminingem a nástrojem IBM SPSS Modeler.
2. Prostudujte klasifikační problém v dataminingových úlohách.
3. Vybraný algoritmus pro budování klasifikačního stromu naprogramujte tak, aby byl pomůckou studentům pro studium problému.
4. Algoritmus ověřte na různých souborech dat vhodných pro vybudovaný algoritmus.
5. Aplikaci a téma budování klasifikačního stromu zařaďte do kurzu DM na e-learningový portál ALS.

Rozsah grafických prací: **dle potřeby dokumentace**

Rozsah pracovní zprávy: **cca 30–40 stran**

Forma zpracování bakalářské práce: **tištěná/elektronická**

Seznam odborné literatury:

[1] **Yong Yin, Ikou Kaku, Jiafu Tang: Data Mining, Springer London Ltd, 2011**

[2] **Hendl J.: Přehled statistických metod zpracování dat, Portál, s.r.o. 2006**

[3] **Olivia Parr Rud: Datamining, Computer Press, a.s., 2006**

[4] **http://www.msps.cz/data-mining/**

Vedoucí bakalářské práce: **RNDr. Klára Císařová, Ph.D.**
Ústav mechatroniky a technické informatiky

Datum zadání bakalářské práce: **10. října 2015**

Termín odevzdání bakalářské práce: **16. května 2016**

prof. Ing. Václav Kopecký, CSc.
děkan

L.S.

doc. Ing. Milan Kolář, CSc.
vedoucí ústavu

V Liberci dne 10. října 2015

Prohlášení

Byl jsem seznámen s tím, že na mou bakalářskou práci se plně vztahuje zákon č. 121/2000 Sb. o právu autorském, zejména § 60 – školní dílo.

Beru na vědomí, že Technická univerzita v Liberci (TUL) nezasahuje do mých autorských práv užitím mé bakalářské práce pro vnitřní potřebu TUL.

Užiji-li bakalářskou práci nebo poskytnu-li licenci k jejímu využití, jsem si vědom povinnosti informovat o této skutečnosti TUL; v tomto případě má TUL právo ode mne požadovat úhradu nákladů, které vynaložila na vytvoření díla, až do jejich skutečné výše.

Bakalářskou práci jsem vypracoval samostatně s použitím uvedené literatury a na základě konzultací s vedoucím bakalářské práce a konzultantem.

Současně čestně prohlašuji, že tištěná verze práce se shoduje s elektronickou verzí, vloženou do IS STAG.

Datum: 4. 1. 2016

Podpis:

Poděkování

Rád bych na tomto místě poděkoval vedoucí práce, paní RNDr. Kláře Císařové, Ph.D. za veškerou podporu a pomoc při psaní a programování a za všechny užitečné rady během konzultací. Dále děkuji mé rodině za jejich veškerou podporu a trpělivost.

Abstrakt

Tato práce seznamuje čtenáře s pojmem datamining, SW nástrojem IBM SPSS Modeler a zaměřuje se na tvorbu rozhodovacích stromů a popisuje pro ně vybrané klasifikační algoritmy, jež jsou využívány při řešení různých reálných problémů. Vybrané algoritmy jsou pak demonstrovány na konkrétních datech.

Práce dále popisuje vznik desktopové aplikace pro vizualizaci vzniku rozhodovacího stromu, jakož to nástroje použitelného pro výuku.

Klíčová slova

Datamining, klasifikace, rozhodovací stromy, informační zisk, entropie, prediktor, predikovaný atribut.

Abstract

This bachelor thesis introduces the concept of data mining and software IBM SPSS Modeler. This thesis is focused on creating decision trees and classification algorithms, which are used for different real problems. The algorithms are showing on specific data.

This bachelor thesis describe creating desktop application for visualisation of a decision tree. This application is a tool for teaching.

Keywords

Datamining, decision trees, information gain, entropy, predictor, predicted attribute.

Obsah

1. Úvod

1.1 Historie

Již od vzniku prvního písma dochází ke sběru informací a vyhledávání v nich. Dlouhá léta se taková data psala ručně. Po vynálezu knihtisku se během padesáti let v Evropě zdvojnásobil počet knih. Dnes, v digitální době, podobný nárůst trvá přibližně 3 roky. [4]

S větším objemem dat, ale vzniká problém jak data archivovat, zpracovávat, jak v nich vyhledávat informace. Od 18. století vznikají první metody jak v datech vyhledávat určité „vzory." Z těch starších metod lze zmínit například Bayesův teorém, používání regresní analýzy a dalších statistických metod. V padesátých letech 20. století, kdy došlo k rozvoji výpočetní techniky, dochází s dalším zvětšováním objemu dat také k rozvoji nových technik, jak analyzovat data. Rozvíjely se například teorie neuronových sítí, shlukové analýzy, genetické algoritmy, rozhodovací stromy, strojové učení.

Teprve až v devadesátých letech se objevuje pojem Dobývání znalostí z databází (Knowledge discovery from databases, KDD) [5]. Na konferencích o umělé inteligenci v roce 1995 v Montreali se projednávalo mimo jiné právě spojení databází, statistických a analytických metod, využití teorie informace a metod umělé inteligence pro získávání skrytých informací z velkých dat. Pojmy datamining a KDD používá mnoho autorů jako synonymum, ale často je datamining chápán jen jako část KDD. Přes uvedené rozdíly, lze oblast zpracování dat v celé šíři naznačených problémů, jednoduše nazvat datamining, protože takto je většinově akceptován.

1.2 Datamining

Existuje mnoho definic, které se liší v detailech, většinou podle odborného zaměření autora. Lze shrnout, že datamining je analytická metodologie získávání netriviálních skrytých a potenciálně užitečných informací z dat. Dodnes se některé definice, tak jak je popsali světoví odborníci, od sebe liší. Například E. Brethnoux popsal datamining jako proces objevování nových významných vztahů, vzorů a trendů při zpracování velkých objemů dat z *datových skladů* pomocí metod automatické detekce závislostí, ale také matematických a statistických algoritmů. Nebo S. Wilson řekl o dataminingu, že pomocí pokročilých matematických metod

odkrývá vzory a vztahy ukryté v *databázích*. Zpravidla se jedná o vzory, které by nebyly detekovány běžnými postupy.

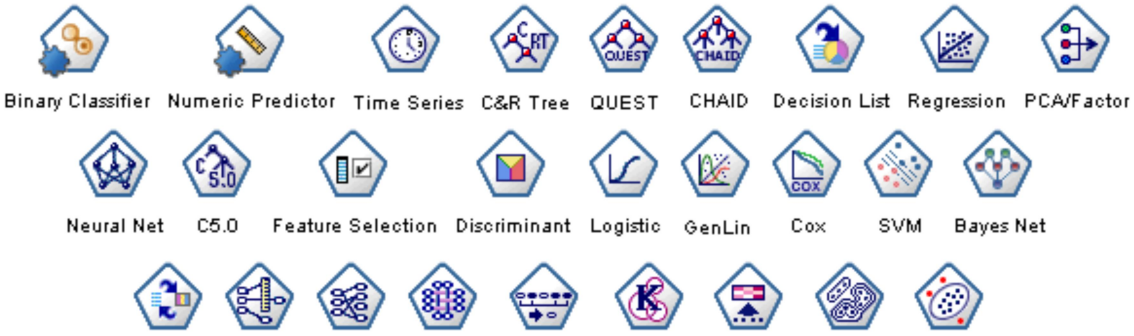
1.2.1 DM nástroje

V současné době pro využívání dataminingu byly vyvinuty různé nástroje a software, které dovolují zpracovávat dataminingový problém. Některé nástroje umožňují pracovat od porozumění datům, přes přípravu dat k modelování a od modelování problémů s nasazením různých postupů až k hodnocení výsledku a nasazení vytvořeného modelu do praxe. Výsledky interpretují uživateli v poměrně přehledných schématech a grafech, popřípadě v tabulkách. Mezi nejznámější aplikace patří například IBM SPSS Modeler, open-sourcová WEKA či Knime. Dnes je už obvyklé, že databázové systémy jako Oracle, ale i další, či statistické softwary jako IBM SPSS Statistics některé dataminingové postupy mají implementované jako nadstavbu.

Problémem je, že většina těchto nástrojů ještě nemá v sobě naimplementováno dostatečné množství statistických a analytických metod, které by pokryly DM úlohy komplexně. Obzvláště se to týká těch, co jsou open source. Jejich postupné přiblížení profesionálním dataminingovým softwarům se dá očekávat a to v souvislosti s aktuální poptávkou po levnějších nástrojích pro DM úlohy.

1.2.2 IBM SPSS Modeler

Software od IBM, IBM SPSS Modeler, je komplexním nástrojem, který dokonce splňuje možnost pracovat v obecně uznávaném postupu, kterým je metodika CRISP DM. Modeler vychází z dřívější verze SW nazývané Clementine a od té doby získal na robustnosti a funkčnosti. Obsahuje všechny funkce pro jednotlivé kroky dataminingu a především velké množství modelů pro DM modelování. Všechny funkce jsou přehledně zobrazeny jako uzly (moduly) různého zaměření. Graficky je jejich funkčnost vyjádřená tvarem. Například vstupní uzly jsou kulaté, modelovací uzly mají tvar pětiúhelníku, šestiúhelníky označují funkce pro práci s daty apod.

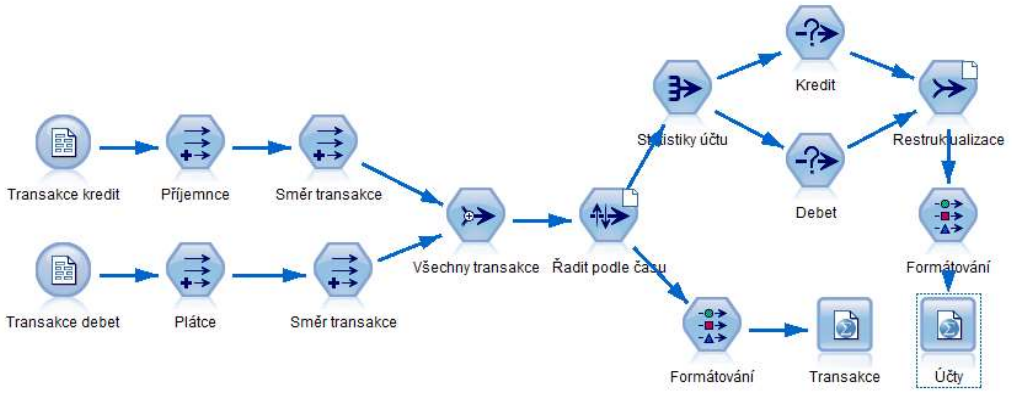


Obrázek 1 Uzly pro modelování

Na Obrázek 1 Uzly pro modelování je uveden přehled implementovaných dataminingových algoritmů, pomocí kterých se dataminingový model staví.

Z uzlů se vytváří sekvenční proud jednotlivých kroků dataminingového řešení. Uzly na sebe navazují spojnicemi, které symbolizují tok dat a jejich zpracování. Proud se značí anglickým „stream“. Každá DM úloha potřebuje jinak připravená data. To závisí nejen na tom, jaké modely hodlá pro danou úlohu dataminer použít, ale také na zdrojích dat. Data mohou mít různou podobu podle toho, jak vznikala. Mohou to být různé databázové zdroje, xls tabulky, statistické soubory typu sav, csv soubory a jiné. Datům je nutné porozumět a také je převést do takového typu a tvaru, který je pro vybraný algoritmus – ukrytý v modelovacím uzlu, vhodný. Pro tyto úkoly dává Modeler mnoho funkcí ukrytých v jednotlivých uzlech. Tato část řešení dataminingové úlohy je velmi náročná na kreativitu, znalosti i čas. V různých zdrojích se uvádí, že představuje až 70% času.

Obrázek 2 Příprava dat ilustruje použití uzlů pro práci s řádky a sloupci načtených dat. Lze vkládat nové atributy, spojovat soubory, vytvářet agregované hodnoty, restrukturalizovat data, seřadit je podle zvoleného kritéria, vybírat potřebné podmnožiny a zpracovaná data uložit do nových souborů různých typů. Pracovní uzly



Obrázek 2 Příprava dat

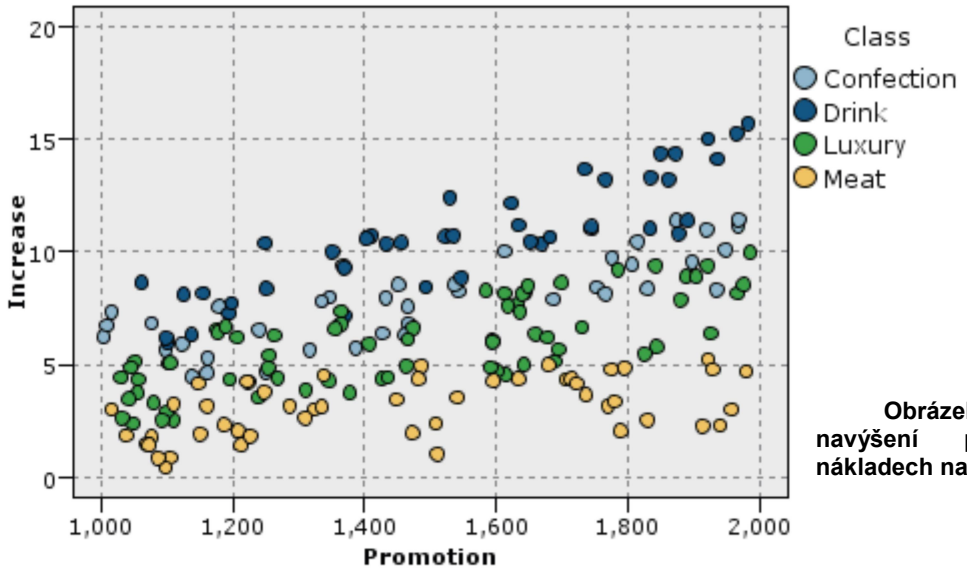
s řádky či sloupci jsou šestiúhelníkové grafické značky a výstup má tvar obdélníku. Každý uzel má mnoho nastavitelných parametrů a tak výsledný proud může být velmi složitý.

Data se načítají obvykle ze souboru do dvourozměrného pole, které si lze představit jako tabulku. Řádky představují záznam o objektu a sloupce jsou atributy, které objekt popisují. Za objekt může být považovaný například zákazník, pacient, ale i transakce bankovní operace aj. Řádek složený z hodnot pro jednotlivé atributy je vlastně vektor, kde definiční obory atributů jsou typově primárně různé. Mohou to být reálná čísla, číselníky, kategorie, stringy, podle typu objektu a dostupných informací o objektu.

Věk	Počet dětí	Pohlaví	Svobodný	Výše příjmu	Zaměstnaný	Výše konta	Úvěr
18	0	Muž	Ano	18 000	Ano	80 000	Ano
40	2	Muž	Ne	0	Ne	550 000	Ano
28	1	Žena	Ne	25 000	Ano	370 000	Ano
35	2	Muž	Ne	65 000	Ano	650 000	Ano
22	0	Muž	Ano	0	Ne	500 00	Ne

Tabulka 1 Možná reálná data

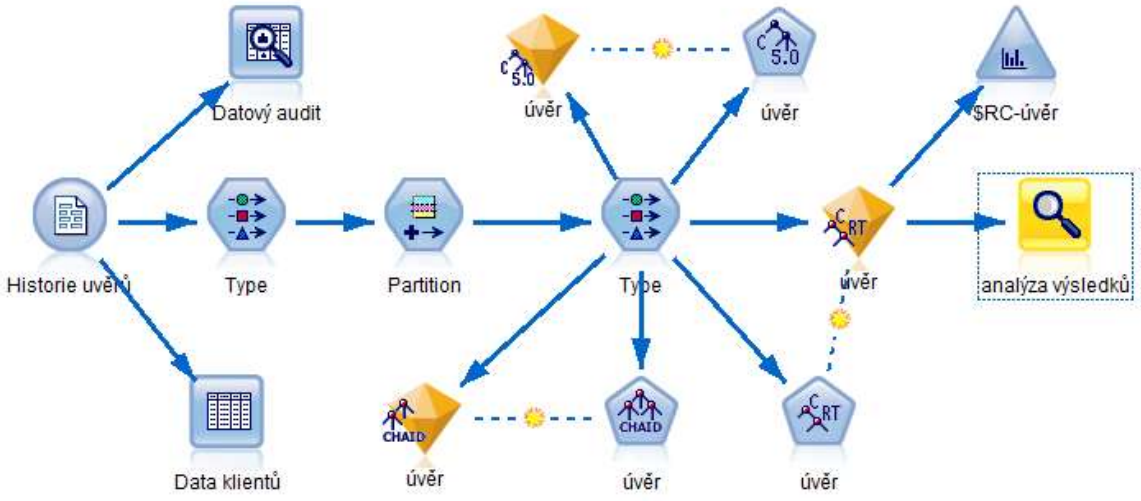
K lepšímu porozumění datům slouží mnoho různých typů grafů a analýz. Například už po načtení dat příslušným vstupním uzlem Modeler data analyzuje, předběžně určí typ a rozsah načtených hodnot a navrhne defaultně roli atributu v úloze. Pod pojmem role atributu se rozumí použití atributu v datamainingovém modelu. Atribut může být použit jako vstupní, výstupní, vstupní a výstupní zároveň nebo také může být bez role, pokud atribut bude ignorován. Dalšími speciálními uzly lze dělat hloubkovou analýzu. Například posoudit kvalitu dat lze uzlem „Data



Obrázek 3 Závislost navýšení prodeje na nákladech na reklamu

audit“. Výsledkem datového auditu jsou základní statistiky pro každý atribut podle typu atributu. Například pro číselné atributy výsledkem maximální, minimální hodnota, průměrná hodnota, směrodatná odchylka, interkvartilový rozptyl, graf rozdělení, zešikmení a další. Zároveň se kontroluje úplnost dat, tedy zjistí se chybějící hodnoty pro daný atribut, případně podle nastavení se hledá výskyt outlierů, pokud jsou data atributu číselná. Uzel „Type“ dovolí změnit typ, vybrat jen některé atributy, přejmenovat je, změnit roli. V průběhu proudu lze data kdykoliv zobrazovat jak pomocí tabulek, tak i grafů nebo matic. Dataminer tak vidí, co se s daty po aplikaci funkcí navržených do proudů, děje.

Samotné modelování v Modeleru umožňuje vyzkoušení několika souběžných modelů a následně jejich porovnání. Proud pro ilustraci rozhodování o udělení úvěru klientovi je „školní,“ nevychází z reálných dat, ale ilustruje použití několika modelů a slouží k porovnání výsledků s vlastním algoritmem, který byl v rámci zadání vytvořen k podpoře výuky v předmětu Datamining. Výsledky modelování pomocí klasifikačních a rozhodovacích stromů jsou dostupné v Modeleru v žlutých „diamantech.“

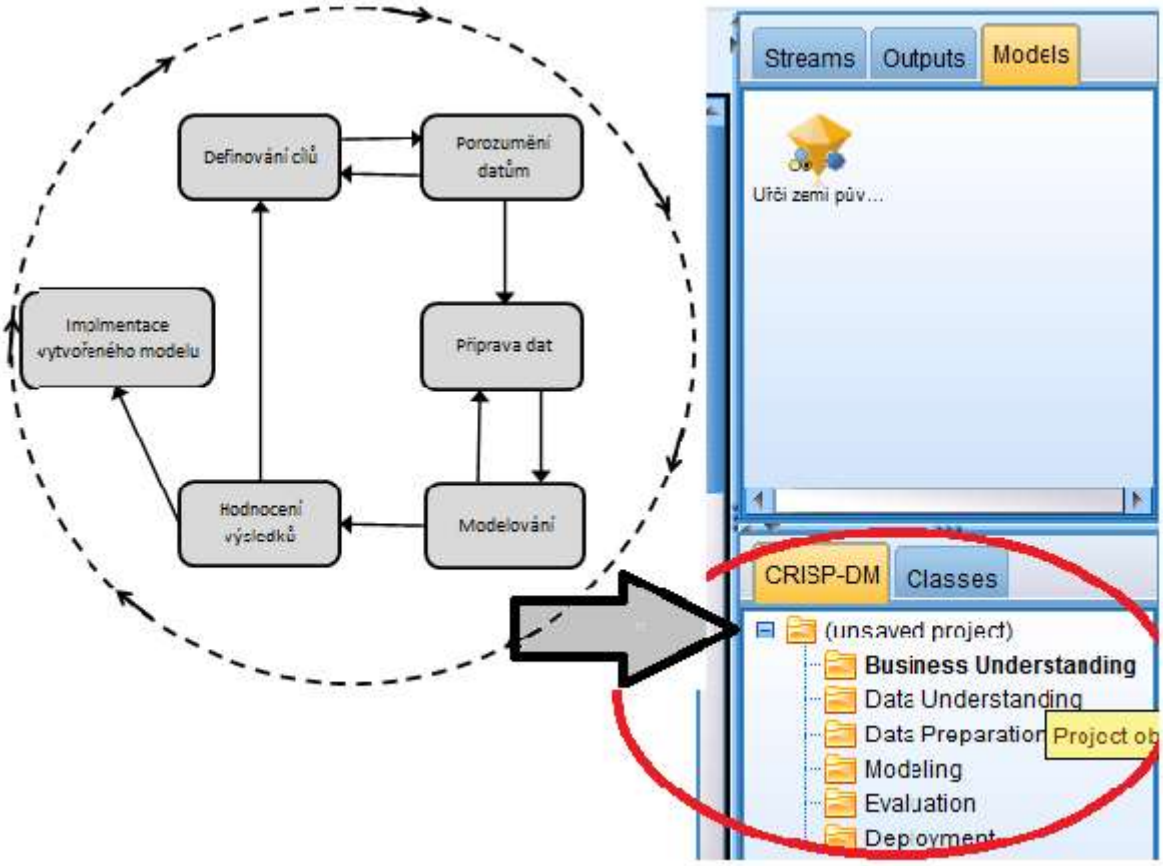


Obrázek 4 Úvěr v modeleru I.

Zmiňovaná metodika CRISP-DM (*Cross Industry Standard Process for Datamining*), definuje jednotlivé kroky a fáze při řešení úloh. Výsledek jednotlivých kroků je přímo ovlivňován stavem kroku minulého, avšak nejedná se vyloženě o cyklus, spíš o iterační proces, ve kterém je výsledek postupně optimalizován

Metodika CRISP-DM se skládá ze šesti kroků životního cyklu projektu:

1. Porozumění problematice – definování cílů (*Business Understanding*)
2. Porozumění datům (*Data Understanding*)
3. Příprava dat (*Data Preparation*)
4. Modelování (*Modeling*)
5. Vyhodnocení (*Evaluation*)
6. Nasazení (*Deployment*)



Obrázek 5 Zabudovaná technologie CRISP DM v Modeleru

1.2.3 Dataminingové úlohy

Dataminingové projekty se objevují všude tam, kde vznikají elektronická data a tyto se v čase hromadí. Je logické, snažit se o získání nějaké výhody z existence těchto dat. Náklady na archivaci a údržbu musí většinou majitel dat vynakládat tak jako tak, protože někdy musí plnit například zákonnou povinnost, jindy mu data slouží jako dokument pro reklamační řízení atd. Elektronická data máme dnes téměř ve všech oborech lidské činnosti a potřeba tyto data dále využívat a zpracovávat je stále naléhavější. Dříve byla v nasazení DM postupů mimořádná výhoda. Například

business pán obchodníka, který přišel s nápadem segmentovat zákazníky pomocí DM, dělat tak cílenou reklamu. Šetřil tím velké finanční částky oproti konkurenci, Dnes je nasazení DM postupů nutnost pro zajištění konkurenceschopnosti. V jiných oborech pomáhá DM chránit společnost před kriminalitou, různými podvody v bankách, při získávání zakázek či úvěrů. Významné jsou úlohy vědeckého charakteru, pomoc v diagnostice nemocí, v predikci šíření kritických nemocí. Textmining je dnes nasazován na webové prostředí a to z mnoha důvodů. První byly e-shopy. Naléhavý začíná být problém šíření nenávistných ideologií prostřednictvím počítačových sítí a i tady bude jistě pomáhat dataminingové postupy.

Přehled nejznámějších úloh aktuálně řešených DM postupy:

V sektoru finančnictví je typické

- **Skórování žádostí o úvěr**
- **Hodnocení chování při splácení úvěrů**
- **Hodnota (bonita) klienta**
- **Podvody při používání platebních karet**

Detekce podvodného jednání

- **Pojistné podvody**
    - Komerční pojišťovny
    - Zdravotní pojišťovny
- **Praní špinavých peněz**
- **Bankovní podvody**
- **Daňové úniky**
- **Korupce**

V sektoru prodej a marketing

- **Predikce úspěšnosti prospektů pro obchodní nabídky**

- **Určení kombinací produktů, které se kupují společně pro cílenou nabídku**
- **Identifikace zákazníků, kteří chtějí rozvázat smlouvu nebo přejít ke konkurenci**

V dalších oborech

- **Genové inženýrství**
- **Medicína – diagnostika chorob**
- **Personalistika – přijímání pracovníků**
- **Školství – udělování stipendií**
- **Obchodní řetězce – nákupní košík**
- **Státní sféra – cílená kontrolní činnost**
- **Logistika – rozmístění skladů**
- **Marketing - segmentace**

Z výše uvedeného je vidět, že typů úloh řešených DM postupy bude velké množství a budou zasahovat mnoha sfér a do budoucna bude jejich počet růst.

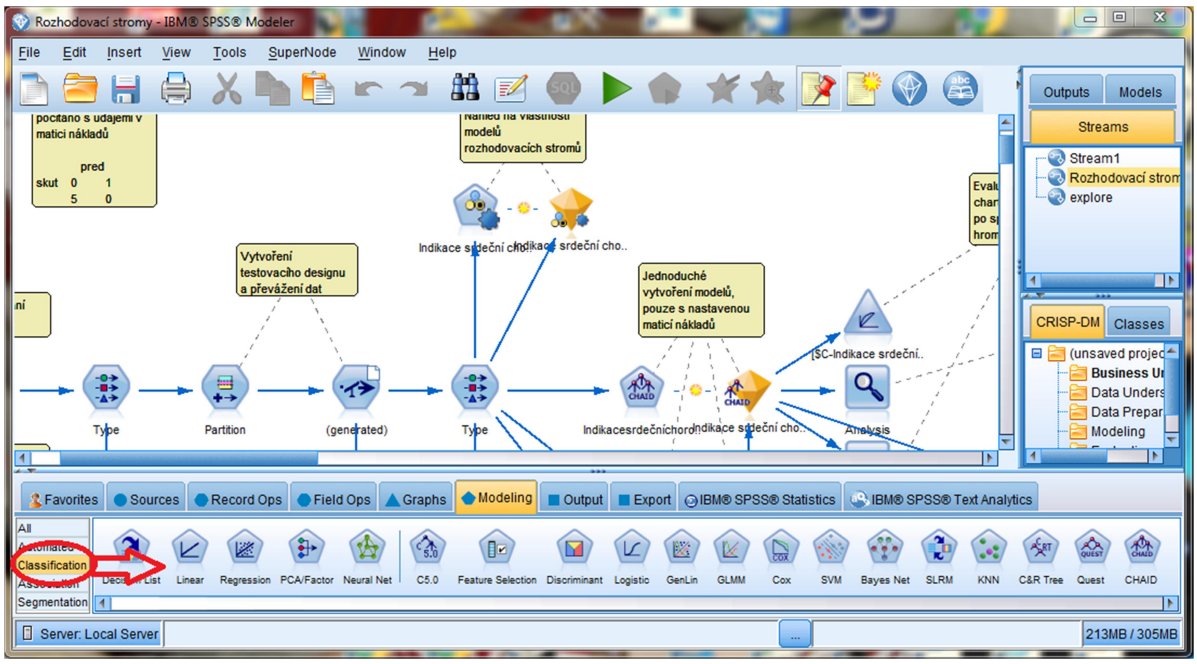
Zajímavé je rozdělení dataminingových úloh podle použitých statistických, matematických postupů, či nasazení algoritmů umělé inteligence. Z tohoto úhlu pohledu lze DM úlohy dělit na *klasifikační, segmentační, úlohy na odhadování hodnot vysvětlované proměnné pro rozhodování, úlohy analýz vztahů, úlohy predikce v časových řadách a úlohy detekce odchylek*.

2. Klasifikační úlohy v DM

V překladu přesně znamená třídění, hodnocení. V některých případech lze najít záměnu klasifikace za diskriminační analýzu, která je ale ve skutečnosti jen podskupinou klasifikace. Podstatou klasifikace je rozdělování objektů, které mají určité charakteristické rysy, do jednotlivých tříd. V dataminingových úlohách to závisí na zvoleném modelu. V některých modelech se vychází ze známé klasifikace a model pak trénuje na tréninkové množině dat. Jsou i jiné postupy, kde se vychází z podobnosti objektů a klasifikace je daná mírou podobnosti objektů. Míra podobnosti se posuzuje různě. Pro klasifikační a rozhodovací stromy, co je v zadání této práce, je typické, že cílová proměnná je známá v datech o historii klasifikace nebo rozhodování. Vybudování rozhodovacího stromu vychází z hledání vhodného predikátoru pro predikovaný atribut a danou množinu. Výběr predikátoru vychází z posouzení závislostí možných predikátorů s cílovou proměnnou pomocí různých statistik či jiných charakteristik. S klasifikací úzce souvisí predikce cílové hodnoty navrženým modelem pro nový objekt, u kterého klasifikace není známá a rozhoduje model. Je to předpověď neboli odhad následujících hodnot cílové proměnné při znalosti hodnot předchozích. Dohromady predikce a klasifikace pracují tak, že pro jeden konkrétní predikovaný atribut A (někdy též zvaný cílový atribut) se provádí modelování vlivu ostatních atributů na atribut A.

Klasifikaci lze rozdělit do několika skupin. Například podle typu učení na učení se s učitelem a bez učitele. A dále podle rozdělení náhodného vektoru na klasifikační a regresní. Anebo podle reprezentace dat na příznakové klasifikátory, sekvenční klasifikátory, strukturální klasifikátory a kombinované klasifikátory. V našem případě jsme se zabývali rozhodovacími stromy, které spadají do kategorie učení se s učitelem a do strukturálních klasifikátorů. Shluková analýza, která spadá do učení se bez učitele, je v práci zmíněná jako jedna z mnoha dalších možností klasifikace.

Na Obrázek 6 Možnosti klasifikace v Modeleru je rozhraní modeleru a v dolní části je vidět mnoho dalších uzlů, pomocí kterých lze řešit klasifikační problém. Kromě rozhodovacích stromů a shlukové analýzy, lze použít například neuronovou síť, lineární a logistickou regresi, diskriminační analýzu, faktorovou analýzu a další. Výběr modelovacích uzlů závisí na zadání a datech, které má dataminer k dispozici.



Obrázek 6 Možnosti klasifikace v Modeleru

Před klasifikací probíhá vždy proces poznávání dat a předzpracování dat. Mimo jiné se z dat odstraňují z dat. Nejčastěji se jedná o chybějící hodnoty, odlehlé hodnoty nebo jde o transformaci dat pro konkrétní algoritmus. Některé klasifikační algoritmy se a chybějícími daty dokáží vypořádat sami. Pokud to je možné, doplní hodnotu podle zvolené metody dataminerem, nebo zavedou novou kategorii, jedná-li se o data kategoriální.

V Modeleru mají právě moduly pro klasifikaci největší zastoupení. Z těch zajímavých lze zmínit například C&RT, CHAID, QUEST, C5.0, Linear regression a neuronové sítě neural Net.

Úlohy pro klasifikaci zabírají širokou škálu dataminingových úloh. Může to být například problém přidělení úvěru v bance, určení botanického druhu, určení diagnózy pacienta či vyhledávání rizikových míst v prevenci kriminality.

2.1 Rozhodovací a klasifikační stromy

Rozhodovací a klasifikační strom je soubor po sobě jdoucích pravidel, která určují a zařazují objekty nějaké množiny. Často je reprezentován grafem, kde uzel (vrchol) reprezentuje danou množinu a větev (hrana) reprezentuje rozpad množiny v uzlu, kde začíná. Listem je pak nazván takový uzel, který se už dál nevětví. Klasifikační strom rozhoduje o kategoriální cílové proměnné u rozhodovacího stromu, kde může být predikovaná hodnota i číselná.

Jedná se o jednu z nejrozšířenějších metod, jak rozdělovat množinu dat. Využívá se v běžném životě, nejen v dataminingu, například pro určování neznámých rostlin podle klíče a podobně.

V praxi se pracuje tak, že se nejprve vytvoří rozhodovací strom z trénovacích dat, která mají jasně určený predikovaný atribut a následně pomocí tohoto stromu zařazují predikovaný atribut testovacích dat. Testovacími daty se posuzuje kvalita navrženého modelu.

Rozhodovací stromy lze rozdělit podle počtu uzlů na binární stromy a obecné stromy, dále podle typu zpracování dat na klasifikační stromy a regresní stromy.

Pro všechny rozhodovací stromy platí obecný algoritmus, TDIDT.

2.1.1 Algoritmus TDIDT [5]

Top Down Induction Decision Trees je základním algoritmem, podle kterého se tvoří všechny rozhodovací stromy. Využívá metodu „rozděl a panuj“, kdy trénovací data rozděluje do menších a menších množin. Tyto množiny by měli být vždy tvořeny jednou převládající třídou nad ostatními, či ideálně pouze jednou třídou. Algoritmus pracuje ve třech krocích:

1. Zvolit jeden atribut jako kořen dílčího stromu.
2. Rozdělit data v tomto uzlu podle hodnot zvoleného atributu a přidat uzel pro každou třídu.
3. Pokud existuje uzel, pro který nepatří všechna data do téže třídy, pro tento uzel opakovat body 1 a 2, jinak skončit.

2.1.2 Obecný rozhodovací strom

Obecný strom je nejjednodušší pro tvorbu a také nejlepším příkladem pro vysvětlení, jak stromy fungují.

Na začátku máme nějaká vstupní data (uvažujeme kategoriální data). Můžeme si je představit jako velkou tabulku s hlavičkou, ve které jsou názvy objektů (sloupců). Pro to, abychom mohli data rozdělit, musíme určit, který z objektů je prediktor, a který je predikovaný atribut. Prediktor je takový objekt, na kterém závisí hodnota predikovaného atributu. Všechny objekty tabulky mají své třídy. Třída je nějaká konkrétní hodnota pro daný atribut.

Cílem je rozdělit data tak, abychom v listech stromu získali pro predikovaný atribut vždy jen jednu jeho třídu v závislosti na jedné třídě prediktora. Někdy to není hned možné, proto je prediktorů vždy více. Pro každou další vzniklou tabulku se provádí postup rozdělení znovu, ale už ne pro stejného prediktora jako v předchozím případě. Výběr vhodného prediktora se provádí daným výpočtem ať už z oblasti statistiky, či z teorie informace a dalších.

Konec růstu u všech stromů je dán takzvanými stop kritérii. Mezi nejčastější podmínky patří Minimální počet tříd pro predikovaný atribut, Poměr četností jednotlivých tříd predikovaného atributu, Hloubka stromu, Šířka stromu. Růst přestává samozřejmě i v případě, když už není k dispozici žádný další predikátor nebo je množina dat malá.

Typickým znakem obecného stromu je mnoho uzlů podle počtu kategorií vybraného prediktora a tím i jeho větší šířka.

2.1.3 Binární rozhodovací strom

Jak napovídá název, v tomto případě se jedná o strom, ve kterém jsou pro každý uzel nejvíce dvě větve. Vznikne, když máme pro každý prediktor, tj. atribut, pomocí kterého se štěpí v uzlu stromu data jen dvě třídy, nebo vzniká z obecného stromu pomocí převodu, nebo vzniká slučováním kategorií pro každý prediktor tak, aby nakonec zbyly kategorie dvě.

Důležitá vlastnost binárních stromů je jejich rychlost průchodu. Jsou daleko rychlejší než obecné stromy. Na druhou stranu mohou být velmi hluboké.

2.1.4 Prořezávání a vyvažování stromu

Prořezávání (tree-pruning) je způsob, jak z obecného stromu vytvoříme strom jednodušší. Děje se tak proto, aby se zrychlil proces klasifikace a kvalita klasifikace. Existují dva způsoby.

- Prořezávání při konstrukci (pre-pruning)
- Prořezávání po konstrukci (post-pruning)

První postup pracuje na principu předčasného ukončení některých větví při tvorbě stromu. Důvody mohou být různé. Například výrazně vyšší počet zastoupení

jedné třídy predikovaného atributu oproti ostatním. V tu chvíli je ale nutné vyřešit, jaká bude hraniční hodnota.

V druhém případě se odstraňují větve již z hotového stromu. Takové větve jsou nahrazeny listy s takovou hodnotou, která se nejčastěji vyskytuje v nahrazované větvi. Postup se opakuje, dokud je určitá míra chyby menší než zadaná hranice. Opět je nutné vyřešit problém, jaká bude hraniční hodnota.

V porovnání obou postupů vychází, že první postup je méně časově náročný než druhý, ale poskytuje o něco horší výsledky.

2.1.5 Porovnání klasifikačních a regresních stromů

Hlavní rozdíl mezi těmito stromy je v pohledu na predikovaný atribut. Klasifikační stromy pracují s kvalitativními daty a regresní stromy s kvantitativními daty.

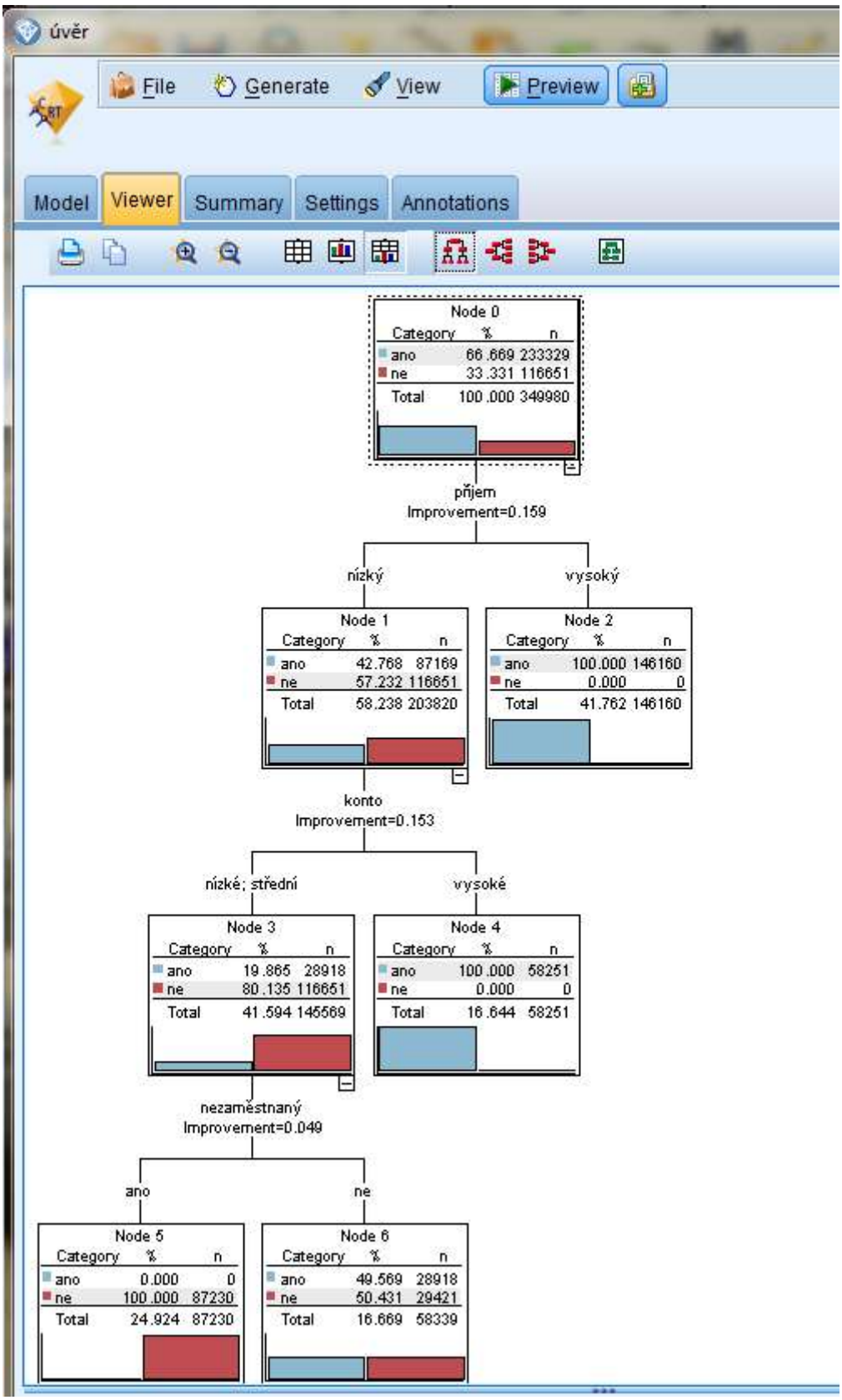
Klasifikační stromy pracují tedy především s kategoriálními proměnnými. Mohou pracovat i s daty, která jsou numerického charakteru. Taková data se pomocí různých statistických výpočtů převádějí na kategorie (kvartil, percentil, medián a podobně).

Kategoriální proměnou se myslí proměnná hodnota, která má nějaký význam. Zastupuje například nějaký číselný interval, nebo jinou skupinu hodnot (tvar okvětí, vysoký příjem, jedovatá houba…).

Regresní stromy pracují jedině s numerickými hodnotami. Například se využívají u internetových vyhledávačů. Ve vstupní tabulce pro regresní strom jsou pak zapsány například četnosti výskytů na konkrétní stránce hledaného slova, page-rank a podobně.

Pro ilustraci jak může vypadat výsledný klasifikační strom je vložený Obrázek 7 Vybudovaný binární strom v Modeleru.

Strom byl vytvořený modelem CART a jedná se o binární strom. Testovali jsme tímto modelem možnosti, které modeler v modelovacích uzlech poskytuje. Data tvořilo milion případů poskytnutí úvěrů, jednalo se o umělá školní data. Na naše testování výpočtů v Modeleru byla množina dostatečná. Na obrázku je jen část stromu. V uzlu 6 je vidět, že odpovídající množina je skoro rovnoměrně zastoupena vzhledem k cílovému atributu. A tedy strom musí pokračovat, aby bylo možno rozhodnout.



Obrázek 7 Vybudovaný binární strom v Modeleru

2.2 Shluková analýza

Jedná se o mnohorozměrné statistické metody, které klasifikují objekty. Objekty třídí do shluků, které obsahují vždy objekty k sobě navzájem nejvíce podobné. Podobnost objektů lze definovat různě a volba metriky či jiného statistického hodnocení podobnosti, je pro řešení dataminingového zadání zásadní. Typově řadíme shlukovou analýzu k metodám učení bez učitele. Vstupní data jsou tedy rovnou zpracovávána, není definovaná cílová proměnná a shluky jsou vytvářeny na základě matematických výpočtů. Shlukování je reprezentace dat, která zmenšuje objem dat za určité ztráty informací. Shlukovací algoritmy rozdělujeme na hierarchické aglomerativní a hierarchické divizní a nehierarchické optimalizační a nehierarchické analýzy módů.

2.2.1 Hierarchické shlukování

Postupy jsou založeny na hierarchickém uspořádání objektů a jejich shluků. Graficky se hierarchicky uspořádané shluky zobrazují formou vývojového stromu nebo dendrogramu. U aglomeračního shlukování na počátku platí, že co objekt to jeden shluk. V prvním kroku se do shluku spojí dva objekty, jejichž vzdálenost je nejmenší, spojí a vypočte se nová matice vzdáleností, v níž jsou vynechány objekty z prvního shluku a naopak tento shluk je zařazen jako objekt. Celý postup se opakuje tak dlouho, dokud všechny objekty netvoří jeden velký shluk nebo dokud nezůstane určitý, předem zadaný počet shluků. Divizní postup je obrácený. Vychází se z množiny všech objektů jako jediného shluku a jeho postupným dělením získáme systém shluků, až skončíme ve stadiu jednotlivých objektů. Otázka výpočtu vzdálenosti dvou více rozměrných objektů, či výpočtu jejich podobnosti je v kapitole 2.2.3

2.2.2 Nehierarchické shlukování

Uživatel na základě svých věcných znalostí, někdy však náhodně, určí, které objekty mají tvořit zárodky nově vytvořených shluků a systém rozdělí objekty do shluků podle jejich vzdálenosti od těchto typických objektů. Existuje několik postupů zadávání zárodků shluku a zařazování objektů do shluku. Vzdálenost mnohorozměrných objektů (řádkových vektorů ve vstupních datech) se dá posoudit různě a to i v závislosti na datových typech jednotlivých atributů shlukovaných objektů. Vzdálenost budeme považovat za míru podobnosti a některé míry budou dále zmíněny. Pro dobré rozložení se musí hodnotit kvalita vznikajících shluků a

případně jejich složení modifikovat přeskupením. Kvalita se hodnotí „funkcionálem kvality rozkladu“, který může být reprezentován:

- Vzdáleností objektů shluku od těžiště
- Vnitroshlukovým rozptylem
- Podobností objektů v shluku
- Mírou separace shluku
- Rovnoměrností rozložení objektů v shluku

Nehierarchické metody mají dvě fáze: určit ideální počet shluků a provést samotné shlukování

2.2.3 Vybrané míry podobnosti pro číselná data

Shluková analýza pracuje na principu porovnávání objektů a zjišťování jak moc si jsou blízko. Pokud bychom měli jednotlivá data reprezentovány například pomocí číselných vektorů, tak bychom hledali vzdálenosti mezi nimi. Vzdálenost ρ(A,B) pro objekty A, B je definovaná metrika v mnohorozměrném prostoru, pokud platí:

1. $\rho(A,B)=0 \Leftrightarrow A=B$
2. $\rho(A,B)\geq 0$
3. $\rho(A,B)=\rho(B,A)$
4. $\rho(A,C)\leq \rho(A,B)+\rho(B,C)$

Takovému měření říkáme také míra podobnosti daná metrikou. Metrik je celá řada, nejznámější jsou

Euklidovská vzdálenost: $$E_u = d(A,B) = \sqrt{\sum_{i=1}^{n}(a_i - b_i)^2}$$

Čebyševova vzdálenost: $$d(A,B) = max|A - B|$$

Manhattanskou vzdálenost: $$d(A,B) = \sum_{i=1}^{n}|A_i - B_i|$$

Tyto metriky pracují s numerickými daty. Problém tedy nastává s daty kategoriálními.

2.2.4 Vybrané míry podobnosti pro kategoriální data

Ideální případem pro taková data jsou binární proměnné. Tedy hodnoty, které nabývají pouze dvou hodnot (například Pohlaví – muž, žena). Vektory takových objektů pak mají na svých pozicích buď 1, nebo 0. Problém nastává u kategoriálních proměnných s více hodnotami. Z takových proměnných lze udělat transformaci na *Indikátorové proměnné.*

id_pacienta	Pohlaví	Věk	diagnoza	doporučený lék
132	M	47	120	L1
201	Ž	35	120	L2
345	M	28	100	L1

Tabulka 2 Původní data

id_pacienta	Pohlaví	Věk	diagnoza	L1	L2	L3	L4
132	M	47	120	1	0	0	0
201	Ž	35	120	0	1	0	0
345	M	28	100	1	0	0	0

Tabulka 3 Po transformaci atributu doporučený lék

Pokud se povede převod všech atributů na dichotomické atributy, tak hledáme vzdálenosti (podobnost) objektů, ale už ne pomocí metrik, nýbrž pomocí Koeficientů asociace. Pro tyto platí

- Používají se výhradně pro objekty reprezentované dichotomickými atributy
- Využívají se asociační tabulky, se kterými koeficienty pracují. Viz Tabulka 4
- Počet atributů, kde oba mají 1 -> a
- Počet atributů, kde oba mají 0 -> d
- Počet atributů, kde první má 0 a druhý 1 -> b
- Počet atributů, kde první má 1 a druhý 0 -> c

Například pro vektor A (1, 0, 1, 1, 0) a pro vektor B (0, 1, 0, 0, 1) bude taková asociační tabulka vypadat následovně:

A / B	1	0
1	A	B
0	C	D

=

A / B	1	0
1	0	3
0	2	0

Tabulka 4 Tabulka shod pro následný výpočet koef. asociace

Z této tabulky pak lze různými způsoby spočítat koeficienty asociace a vytvářet shluky. Některé výpočty operují pouze s *ab, b, c,* a úplně tak ignorují *d* (například Jaccardův koeficient, Diceův koeficient a další), jiné operují se všemi koeficienty (například Sokalův Michenerův).

Jaccardův	$S_j = a/(a+b+c)$
Sokalův a Michenerův koeficient	$S_{sm} = (a+d)/(a+b+c+d)$
Diceův	$S_d = 2*a/(2*a+b+c)$

Příklad v příloze 1 ukazuje, že několik různých koeficientů poskytuje podobné posouzení vzdáleností mezi objekty.

2.2.5 Koeficienty (ne)podobnosti shluků

Shluková analýza poskytuje mnoho způsobů jak nacházet shluky podobných dat a dále se shluky pracovat. Jejich základní dělení je uvedeno v 2.1 a 2.2.2

Podobnost shluků lze posuzovat například podle koeficientů (ne)podobnosti shluků, v práci jsou uvedené některé z metrik používaných pro číselná data a koeficienty asociace pro binární (dichotomická) data. Pro počítání příslušnosti objektů do shluku pro ilustraci uvádíme tři nejintuitivnější metody:

1. Nejbližšího souseda
2. Nejvzdálenějšího souseda
3. Centroidní metoda

Metoda nejbližšího souseda: Zde je nepodobnost shluků vyjádřena (ne)podobností dvou objektů. První objekt je součástí prvního shluku, druhý objekt náleží do shluku druhého a hledá se ten pár, který si je nejvíce podobný, což pro číselná data vyjádříme pomocí metrik a spočítanou vzdálenost chápeme jako koeficient podobnosti objektů, vzdálenost shluků určuje ten pár, jenž má nejmenší vzájemnou vzdálenost.

Metoda nejvzdálenějšího souseda shlukuje objekty tříděné množiny, které jsou nejdále od sebe. To znamená, že za vzdálenost dvou shluků se bere největší možná vzdálenost ze vzdáleností každých dvou objektů z dvou různých shluků. Z takto vypočítaných vzdáleností se pak vybere nejkratší a spojí odpovídající objekty.

Třetí metodou jak určit míru (ne)podobnosti shluků je centroidní metoda. V tomto případě se pro daný shluk vypočítá centroid (těžiště), což je pomocný objekt, který lze chápat jako typický objekt pro shluk, jehož hodnoty jednotlivých atributů jsou tvořeny středními hodnotami daného atributu pro všechny objekty daného shluku. Poté, co se vypočítají centroidy, jsou na ně aplikovány koeficienty podobnosti objektů.

2.2.6 Algoritmus K-Means

Algoritmus K-Means se řadí mezi nehierarchické metody zachovávající počet shluků v průběhu algoritmu. Známá je také jeho metoda *MacQueenova k-průměrů*.

V algoritmu K-Means je shluk reprezentovaný pomocí těžiště. Těžiště, nebo-li centroid, lze chápat jako typický objekt pro shluk. Tyto typické objekty mohou být buď čistě vypočítány, anebo to mohou být existující objekty ve vstupní množině. Podobnost mezi shluky se pro číselná data počítá některou z používaných metrik právě pro centroidy porovnávaných shluků. Podstatné pro tuto metodu je, že počet shluků je na počátku dán a v průběhu přerozdělování objektů do shluků, se nemění.

Počáteční typické objekty, tedy těžiště, mohou být zadány několika způsoby. Buď se může jednat o prvních k objektům z množiny vstupních objektů, nebo se může jednat o náhodný výběr, případně se mohou použít metody, které se snaží o co nejrozptýlenější pozice typických objektů.

K-Means v krocích:

1. Určení počtu shluků *N*
2. Výběr *N* počátečních objektů, které budou reprezentovat první iteraci centroidů. Výběr může být proveden mnoha postupy, některé jsou zmíněny výše.
3. Dále se všechny objekty postupně přiřazují k těm shlukům, k jejichž typickým bodům – centroidům či těžištím, mají nejblíže.
4. Typické body shluku jsou přepočítané pro nové rozmístění objektů

do shluků.

5. Pokud nedošlo k žádnému přeřazení objektů, algoritmus se ukončí. Jinak se opakují kroky 3 – 5.

Ukončení algoritmu může být i jiné. Především proto, že na mnohorozměrných velkých datech může být čas řešení neúměrný potřebě využití výsledku, nebo se výpočet zacyklí. Proto jako stop kritérium se často přidává počet iterací, případně další podmínky. Výsledek pro stejnou množinu dat se může měnit v závislosti na několika faktorech. Například výsledek ovlivní volba prvních *N* typických objektů, dokonce pořadí zpracovávaných objektů. Odlehlé objekty mohou taky zásadně ovlivnit výpočet typických objektů pro jednotlivé shluky. Vychýlí těžiště a tím výrazně ovlivní výpočet. Na druhou stranu, odlehlé objekty mohou ukrývat v sobě něco netypické, podezřelé a naopak dataminer bude právě toto chtít objevit. Pro první přiblížení lze použít K-Means s nastavením N=1 a výpočtem vzdáleností všech objektů od centroidu. Odlehlý objekt, objekt s významně velkou vzdáleností od centroidu – typického objektu, je pak předmětem dalšího zkoumání.

3. Budování klasifikačního stromu

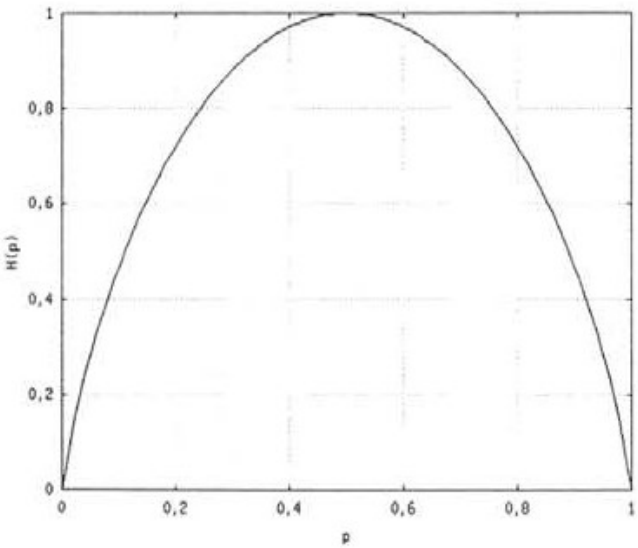
Základní principy klasifikačních a rozhodovacích stromů byly popsány v předešlých kapitolách. Podstatné pro tuto práci je jak vybrat prediktora z možných kandidátů, které jsou v datové v datové matici označené jako vstupní. Pro náš algoritmus byla vybrána podmíněná entropie a informační zisk jako kritérium rozhodující o kvalitě možných prediktorů. Pochopitelně existuje mnoho dalších možností jak postupovat, tedy jak zkoumat závislost či korelaci mezi předikovanou proměnnou a možnými atributy v roli prediktora.

3.1 Entropie jako vhodná charakteristika pro výběr prediktorů

V přírodních vědách udává pojem entropie míru neuspořádanosti nějakého systému. V teorii informace je definována funkcí (Berka, 2004)

$$H = -\sum_{t=1}^{T}(p_t * \log_2[p_t)]$$

Kde p_t udává pravděpodobnost výskytu třídy *t*. Jde o relativní četnost, která je počítaná na určité množině. *T* pak udává celkový počet tříd. Entropii můžeme chápat také jako míru nejistoty spojené s náhodnou proměnnou. Čím je tato hodnota menší,



Obrázek 8 Entropie pro náhodnou proměnnou se dvěmi třídami

tím jsme si jistější v predikci očekávaného výsledku spojeného s náhodnou proměnnou.

Na obrázku je vidět graf průběhu entropie v závislosti na pravděpodobnosti *p* v případě dvou tříd. Je-li *p* jedna a tedy všechny příklady patří do této třídy nebo je-li *p* nula a tedy žádný příklad nepatří do této třídy, pak je entropie nulová. V případě, že

je *p* rovno jedné polovině, a to znamená, že jsou obě třídy zastoupeny stejným počtem příkladů, je entropie maximální.

3.1.1 Obecný výpočet entropie

Pro každou třídu *v*, kterou nabývá atribut *A,* je podle vzorce na skupině příkladů spočítána entropie $H(A_{(v)})$. Skupina příkladů je pokrytá kategorií $A_{(v)}$.

$$H(A(v)) = -\sum_{t=1}^{T} \frac{n_t(A(v))}{n(A(v))} * \log_2 \frac{n_t(A(v))}{n(A(v))}$$

$$H(A) = -\sum_{v \in Val(A)} \frac{n(A(v))}{n} H(A(v))$$

.Následně je spočítána střední entropie atributu *A* jako vážený součet entropií $H(A_{(v)})$, kde váhy v součtu jsou četnosti kategorií $A_{(v)}$ v datech.

.Pro větvení stromu je pak vybrán atribut s nejmenší entropií *H(A)*.

3.1.2 Podmíněná entropie

Říkáme podmíněná entropie, protože počítáme četnosti tříd v závislosti na predikovaném atributu. Počítá se nejprve entropie třídy prediktora vůči predikovanému atributu. Tedy pravděpodobnost p_t se spočítá jako poměr četností třídy predikovaného atributu pro konkrétní třídu prediktora na celkový počet výskytů té stejné třídy prediktora.

Příklad:

Mějme příjem vysoký, nízký a střední a na základě toho udělujeme úvěr. Příjem vysoký má počet výskytů 4, přičemž 3x byl úvěr udělen a 1 nebyl. Příjem nízký má 7 výskytů a z toho 2x byl udělen úvěr a 5x nebyl. Příjem střední má 5 výskytů a z toho 2x byl udělen úvěr a 3x nebyl.

Výpočet:

$$H\left(\frac{P_v}{\acute{U}}\right) = -\left(\frac{3}{4} * \log_2 \frac{3}{4} + \frac{1}{4} * \log_2 \frac{1}{4}\right)$$

$$H\left(\frac{P_n}{\acute{U}}\right) = -\left(\frac{2}{7} * \log_2 \frac{2}{7} + \frac{5}{7} * \log_2 \frac{5}{7}\right)$$

$$H\left(\frac{P_s}{\acute{U}}\right) = -\left(\frac{2}{5} * \log_2 \frac{2}{5} + \frac{3}{5} * \log_2 \frac{3}{5}\right)$$

Abychom získali celkovou entropii prediktora, musíme jednotlivé entropie tříd vynásobit podílem počtu zastoupení třídy a celkovým počtem záznamů.

Pro předchozí případ bude konečný výpočet vypadat takto:

$$H\left(\frac{P}{\acute{U}}\right) = H\left(\frac{P_v}{\acute{U}}\right) * \frac{4}{16} + H\left(\frac{P_n}{\acute{U}}\right) * \frac{7}{16} + H\left(\frac{P_s}{\acute{U}}\right) * \frac{5}{16}$$

3.2 Informační zisk další vhodná charakteristika pro výběr prediktorů

Informační zisk je odvozen od entropie. Jde o rozdíl entropie predikovaného neboli cílového atributu *H(C) a uvažovaného* atributu, též prediktora, *H(A)*. Je tak měřena redukce entropie, která je způsobena volbou atributu *A*. Protože počítáme s predikovaným atributem, jsou všechny entropie prediktoru počítány podmíněně, v závislosti právě na predikovaném atributu. H(C) je tedy entropie predikovaného atributu a H(A) je podmíněná entropie prediktora. (Berka, 2004)

$$Zisk(A) = H(C) - H(A)$$, kde $$H(C) = -\sum_{t=1}^{T} \frac{n_t}{n} * \log_2 \frac{n_t}{n}$$.

Na rozdíl od entropie, ale informační zisk hledá atribut s maximální hodnotou. Je to dáno tím, že entropie pro celá data není závislá na atributu. Z toho plyne, že první člen rozdílu je konstantní a tedy maximální rozdíl nastane tehdy, pokud druhý člen rozdílu bude minimální.

Nevýhodou ale je, že se nezahrnuje do úvahy počet hodnot daného atributu. Jde pouze o odlišení příkladů různých tříd na základě vybraného atributu. Problém nastane například v případě, že bychom pro další větvení jako atribut vybrali pořadové číslo příkladu. Tento atribut by sice umožnil bezchybně rozdělit a klasifikovat data, ale byl by zcela nepoužitelný pro klasifikaci dalších příkladů. Z toho důvodu byl zaveden *poměrný informační zisk*.

$$Poměrný\ zisk(A) = \frac{Zisk(A)}{Větvení(A)}$$

.

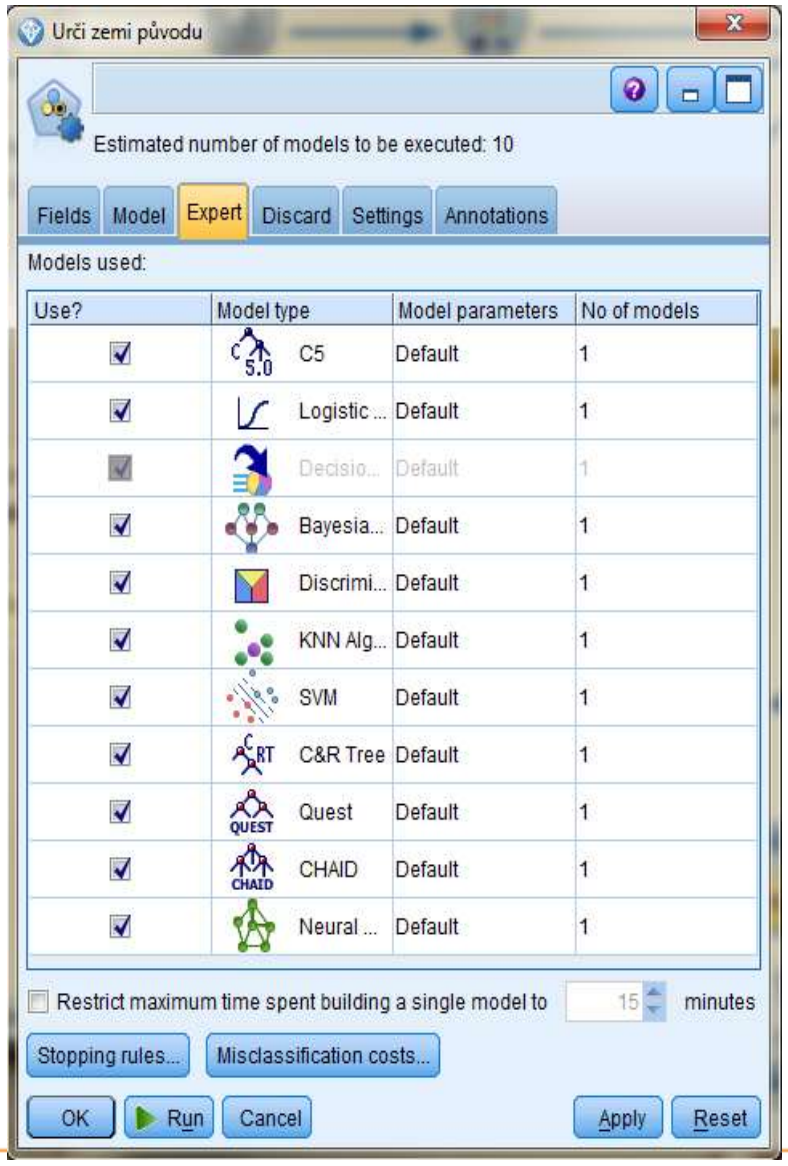
Větvení(A) je vlastně entropie dat k hodnotám atributu *A*.

$$Větvení(A) = -\sum_{v \in Val(A)} \frac{n(A(v))}{n} \log_2 \frac{n(A(v))}{n}$$

.

3.3 Klasifikační úlohy v Modeleru

Naznačené postupy pro klasifikační úlohu nejsou úplným výčtem všech možností jak klasifikaci řešit. To by přesahovalo rozsah zadané bakalářské práce. Proto jen pro ilustraci uvádíme některé další možné postupy jak klasifikační problém řešit. Typicky lze použít neuronové sítě, lineární a logistickou regresi, diskriminační analýzu a další. IBM SPSS Modeler nabízí pro klasifikační úlohu uzel *Automatický výběr*. Uživatel může jednoduchým způsobem navolit několik implementovaných modelů najednou a výsledky pak porovnat. Případně vybrat jeden nejvhodnější model. Modely pak mohou pracovat každý sám za sebe, nebo vytvářet navazující sekvence a dokonce spolupracovat na základě různých scénářů. Tato problematika je nad rámec bakalářské práce, nic méně obrázek ilustruje možnosti klasifikace v Modeleru, se kterým jsem se měl seznámit a své výsledky vybudování klasifikačního stromu porovnat se stromem, který na stejných datech postaví Modeler.



Obrázek 9 Další možné postupy

4. Vlastní aplikace MyTree

Cílem bakalářské práce bylo vytvořit aplikaci, která by přehledně zobrazovala tvorbu klasifikačních stromů a byla by tak pomůckou studentům ke studiu dataminingu. Zároveň má aplikace poskytovat základní informace k problému budování klasifikačních stromů a to tak, aby si jednotlivé kroky na malém souboru student mohl ověřit, přepočítat si ručně výběr prediktoru na dané množině. Aplikace studentovi dovolí i experiment s velkými daty, tam už ruční výpočet nepřichází do úvahy, ale je zobrazen výsledek.

Aplikace byla napsána v jazyce C#, kromě úvodní obrazovky, která je webovou stránkou v desktopové aplikaci, kterou má student dostupnou z e-learningového portálu Fakulty mechatroniky. Aplikaci jsem nazval MyTree.

4.1 Vstupy

Nejprve byla vybrána vstupní vzorová data, o nichž jsem výpočtem v Microsoft Excelu zjistil, jak bude vypadat výsledný klasifikační strom. Výběr atributu, který je na dané množině nejlepší prediktor, jsem počítal na základě podmíněné entropie atributu v závislosti na cílovém, tedy predikovaném atributu.

Tato vzorová data pak byla zadána do aplikace jako testovací a kontroloval jsem tak, jestli mnou navržený algoritmus pracuje správně. Testovací data jsou podle [5].

Klient	Příjem	Konto	Pohlaví	Nezaměstnaný	Úvěr
1	vysoký	vysoké	žena	ne	ANO
2	vysoký	vysoké	muž	ne	ANO
3	nízký	nízké	muž	ne	NE
4	nízký	vysoké	žena	ano	ANO
5	nízký	vysoké	muž	ano	ANO
6	nízký	nízké	žena	ano	NE
7	vysoký	nízké	muž	ne	ANO
8	vysoký	nízké	žena	ano	ANO
9	nízký	střední	muž	ano	NE
10	vysoký	střední	žena	ne	ANO
11	nízký	střední	žena	ano	NE
12	nízký	střední	muž	ne	ANO

Tabulka 5 Vstupní vzorová data

Data pro aplikaci jsou uložena v csv souboru a od sebe oddělena středníkem.

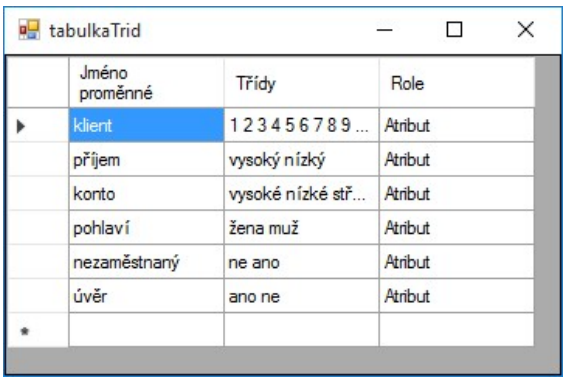
Vlastní vstupní data mohou být jiná než výše popisovaný vzorový soubor, ale s omezením, které vyplývá z metod budování klasifikačních stromů, které jsem podle

zadání programoval. Pravidla pro jiné datové vstupy jsou popsané v následující kapitole. Aplikace má ještě jeden vstupní soubor, ve kterém se ukládají některé parametry rozhraní „nastaveni.ini," pro zvýšení komfortu uživatele. Například se zde ukládá hodnoty o tom, jak velké bude písmo v aplikaci, jakou barvou bude obarven který prediktor

4.1.1 Načtení dat

Data jsou do aplikace nahrávána pomocí dialogového okna, pro které je nastaven filtr pro csv soubory. Nelze tedy nahrát jiné typy souborů. Každý soubor musí mít hlavičku s označením pro jednotlivé sloupce tabulky. Pokud takovou hlavičku soubor má, nastaví do hlavičky tabulky na první řádek dat. Jednotlivé sloupce musí obsahovat pouze „významová" data, tedy ty, které se v dataminingových algoritmech označují jako kategoriální. Mohou být nominální, to znamená, že jednotlivé hodnoty nejde seřadit jako u atributu ***pohlaví*** nebo ordinální, které naopak seřadit lze, například atribut ***konto.***

Nejprve je tedy přečten první řádek souboru a nastavena hlavička tabulky. Pokračuje se čtením souboru do konce, a zároveň se hodnoty uloží do tabulky. Viz Obrázek 10 Načtené hodnoty. Tato tabulka je následně nastavena vstupnímu **Uzlu** (kořenovému uzlu) pro vznikající strom. Na poklikání na uzel se otevře dialogové okno s tabulkou, kde si může student data prohlédnout a zároveň se otevře informační okno, které uživateli říká, jaké třídy obsahuje daný atribut a jakou má v současné době roli



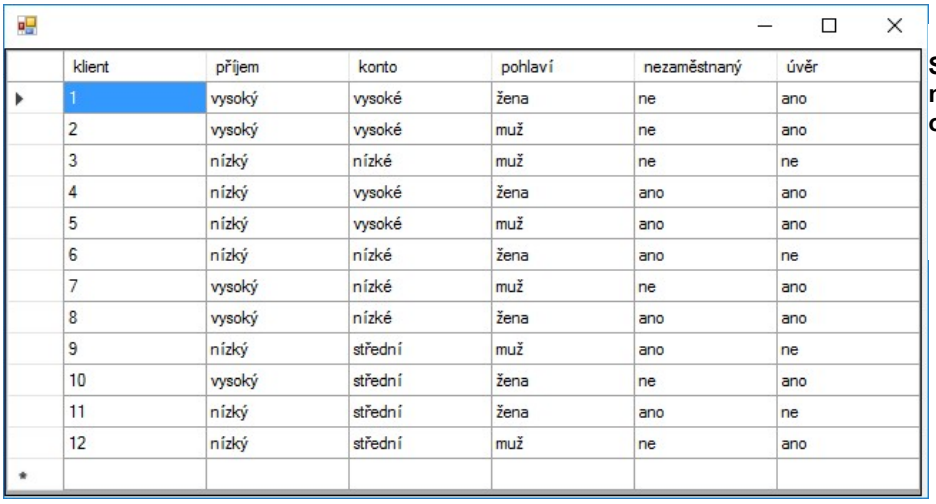
Obrázek 10 Načtené hodnoty

4.2 Struktura Uzel a design

Byla zmíněna struktura Uzel. Jedná se o třídu, která v sobě uchovává veškeré potřebné informace. Vedle třídy Uzel pracuje třída UzelDesign, která říká konkrétnímu Uzlu, jak bude vypadat a jak se bude chovat jeho vzhled. Obě třídy dohromady tedy tvoří to, jak bude vypadat uzel stromu a jak se bude chovat.

Na kliknutí na jakýkoliv uzel ve vytvořeném stromu nám kromě svých dat, která obsahuje, zobrazí ještě četnosti jednotlivých tříd pro všechny prediktory, které ještě nebyli použiti. Navíc se ještě v datech obarví sloupec vybraného prediktora a sloupec predikovaného atributu obarví na zeleno nebo červeno podle toho, zda je cílová hodnota v datové množině uzlu jednoznačně určená, či nikoliv.

Asi nejdůležitější částí pro funkčnost aplikace jsou zde dvě metody. První počítá četnosti tříd pro konkrétní atribut nad celými daty. Druhá počítá četnosti tříd v závislosti na predikovaném atributu.

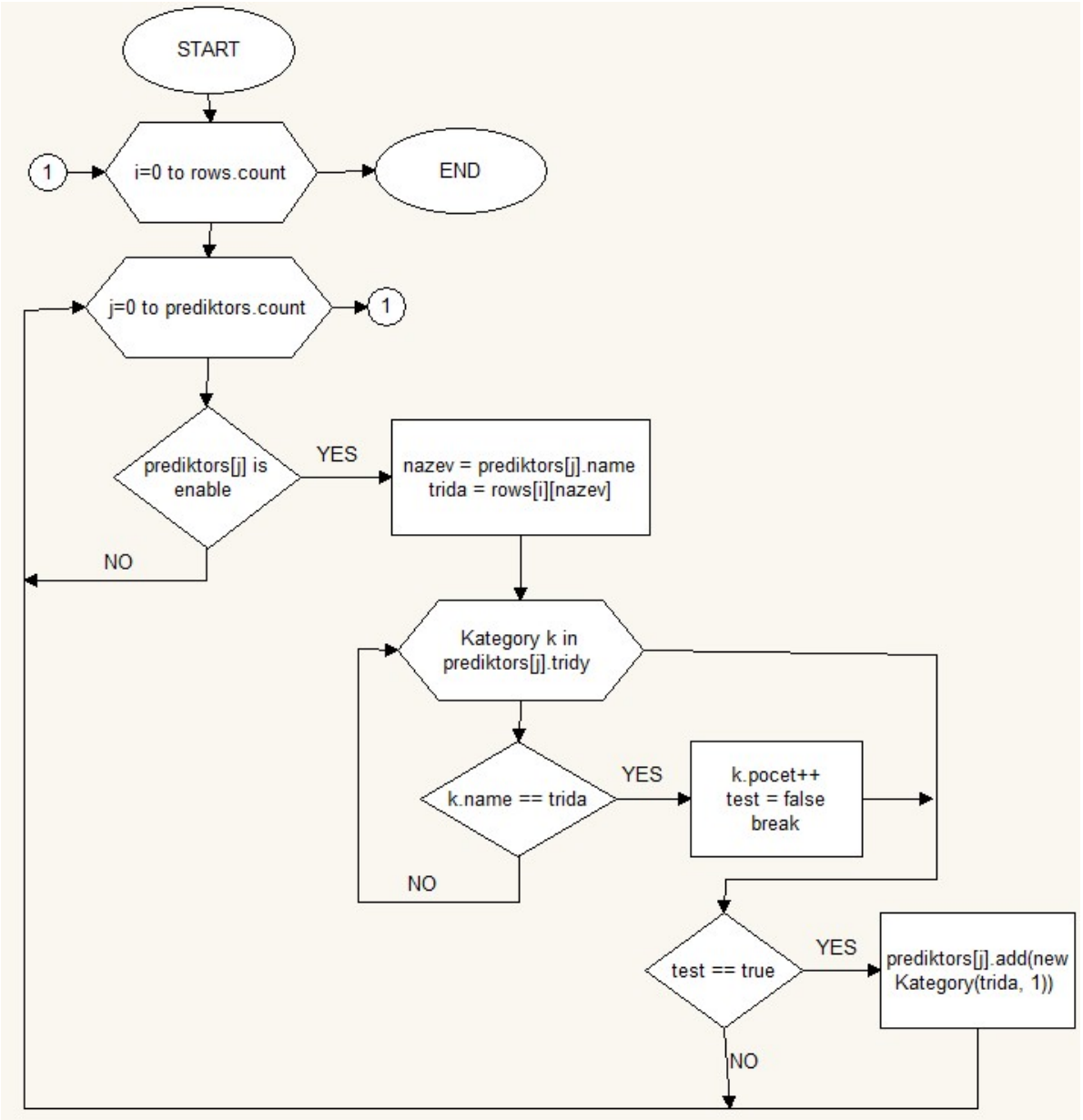


klient	příjem	konto	pohlaví	nezaměstnaný	úvěr
1	vysoký	vysoké	žena	ne	ano
2	vysoký	vysoké	muž	ne	ano
3	nízký	nízké	muž	ne	ne
4	nízký	vysoké	žena	ano	ano
5	nízký	vysoké	muž	ano	ano
6	nízký	nízké	žena	ano	ne
7	vysoký	nízké	muž	ne	ano
8	vysoký	nízké	žena	ano	ano
9	nízký	střední	muž	ano	ne
10	vysoký	střední	žena	ne	ano
11	nízký	střední	žena	ano	ne
12	nízký	střední	muž	ne	ano

Obrázek 11 Struktura načtených dat v okně aplikace

4.2.1 Metoda pro obyčejné počítání četností

Proto, abychom spočítali četnosti všech tříd pro každý atribut z dat, musíme projít všechny řádky tabulky. Pro každý atribut kontrolujeme, zda je prediktorem, a pokud ano, tak ukládáme hodnotu z konkrétní buňky. Tuto hodnotu porovnáváme s hodnotami v listu tříd pro daného prediktora a pokud je nalezena shoda, tak zvýšíme počet záznamů o jedna a pokud ne tak vytvoříme novou třídu o počtu záznamů jedna. Viz obrázek.

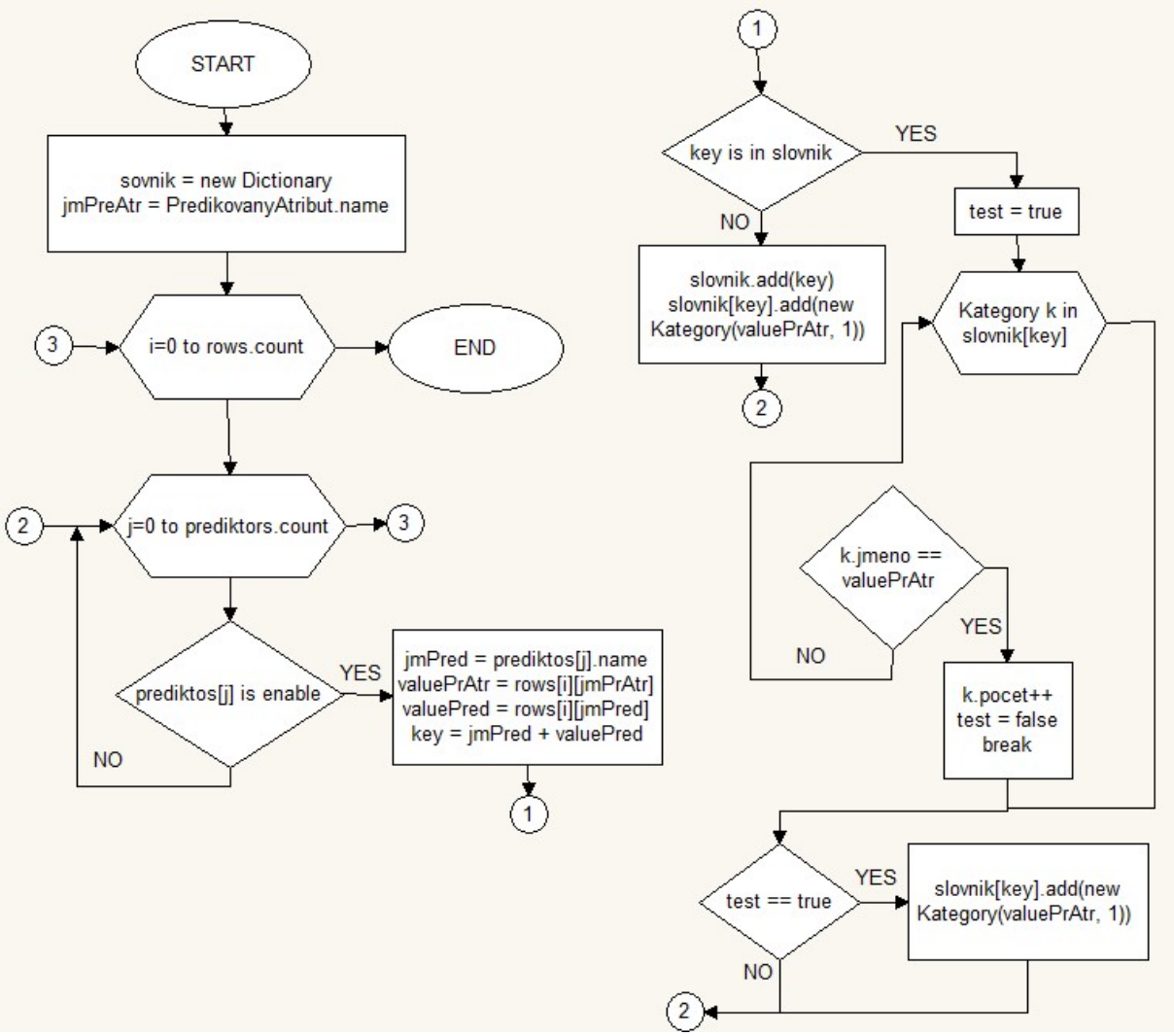


Obrázek 12 Vývojový diagram pro metodu 4.2.1

4.2.2 Metoda pro počítání četností v závislosti na predikovaném atributu

Už na začátku metody musíme vědět, jaký atribut je predikovaným. Pro tento problém byl vytvořen slovník s řetězcovým klíčem a jako hodnotu byl zvolen list tříd. Opět musíme projít data přes všechny řádky a přes všechny aktuální prediktory. Pro každou procházenou buňku vytváříme klíč spojením jména prediktora a konkrétní hodnoty v buňce. Ten pak porovnáváme s klíči ve slovníku, zda se zde už náhodou nevyskytuje. Pokud ne, tak je vytvořen nový záznam ve slovníku s hodnotami třída predikovaného atributu s počtem záznamů jedna. Pokud se klíč ve slovníku již

nachází, tak procházíme všechny hodnoty ve slovníku pro tento klíč a pokud se zde aktuální hodnota nenachází tak přidáme do listu pro tento klíč nový záznam. Pokud se zde nachází tak je pouze zvýšen počet záznamů o jedna. Vzniklý slovník je návratovou hodnotou celé metody. Viz obrázek.



Obrázek 13 Vývojový diagram pro metodu 4.2.2

4.3 Struktura Prediktor a Kategorie

Pro zobrazení atributu jako prediktora byly vytvořena struktura jménem Prediktor. Ta uchovává o kromě svého jména ještě všechny své třídy v Listu, dále si pamatuje, zda byl prediktor již použit pro rozhodování a také svojí spočítanou hodnotu. Pro reprezentaci tříd prediktora byla vytvořena třída Kategorie, která si pamatuje pouze své jméno a počet výskytů a ve třídě Prediktor je právě ona datovým typem pro List tříd prediktora.

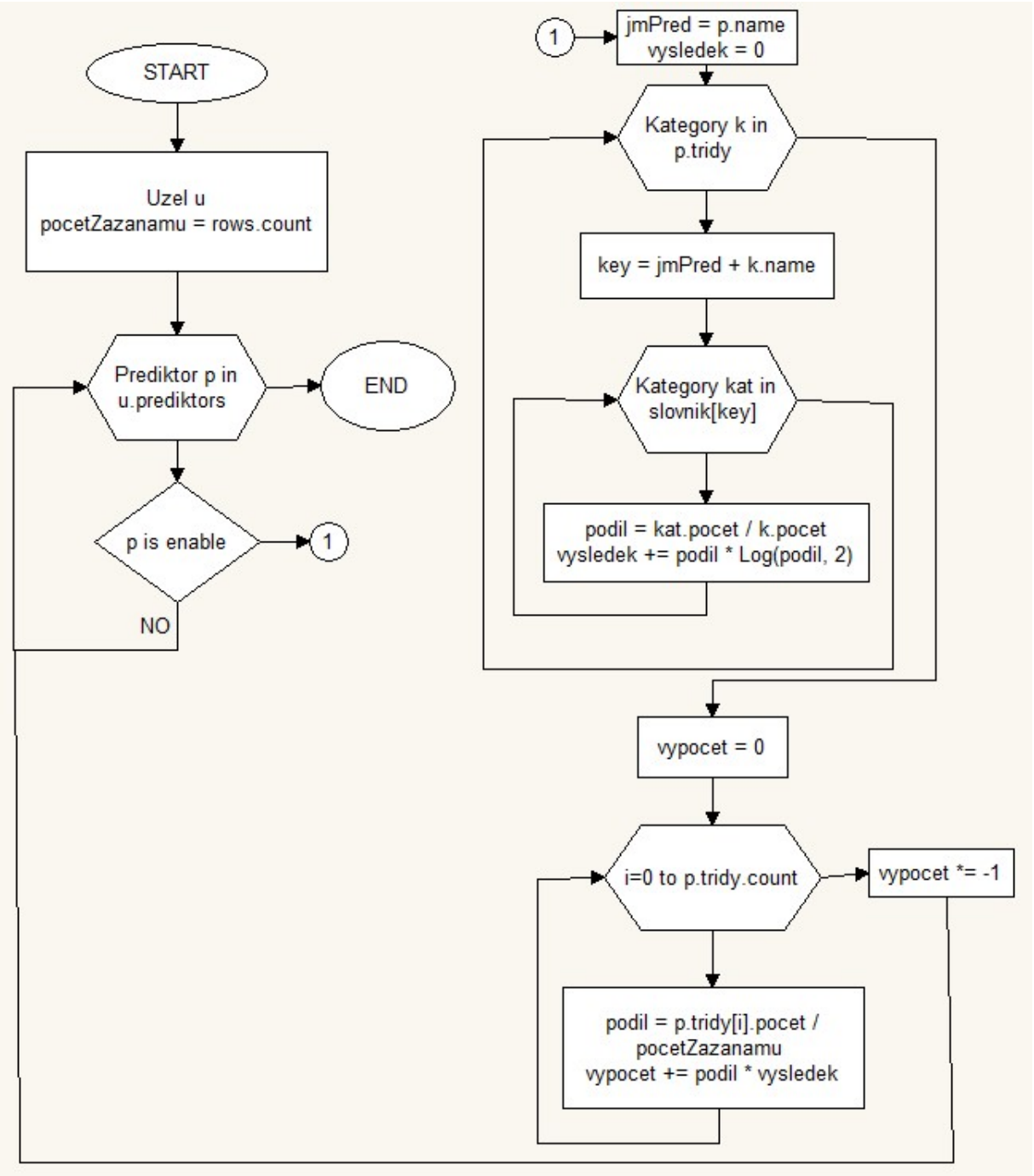
Dohromady tyto dvě třídy tvoří základní stavební prvky pro počítání četností a výpočty entropie i informačního zisku.

4.4 Tvorba stromu

Metody pro tvorbu stromu jsou zapsány v hlavní třídě Datamining. Kromě metody pro vykreslování se jedná o čtyři základní metody. Metoda pro průchod do šířky (Projdi ()), která uvnitř spouští počítání četností a zbylé metod. První je buď metoda pro výpočet entropie (spočítejEntropy()), nebo metoda pro výpočet informačního zisku (spočítejIFZ()). To záleží na tom, kterou metodu si uživatel vybere v combo boxu. Následuje nejdříve metoda pro nalezení nejlepšího prediktora (vyberNej()), a pak metoda pro rozdělení aktuálních dat podle nejlepšího prediktora (rozdělpodlenej()). Průchod pokračuje do té doby, dokud vznikají stále nové uzly.

4.4.1 Metoda pro Entropii a metoda pro Informační zisk

Teorie pro výpočet podmíněné entropie je popsána v kapitole 3.1.2. Pracuje na základě průchodu všech ještě nepoužitých prediktorů pro konkrétní uzel, který je uvažován, a pokračuje průchodem všech tříd prediktora. Pro každou třídu se podívá do slovníku vzniklý z počítání četností v kapitole 4.2.2 a s konkrétní hodnotou spočítá podle vzorce entropie výslednou hodnotu. Říkejme jí například mezivýsledek. Metoda pokračuje dalším průchodem všech tříd prediktora, ale počítá z nich už jen podíl mezi celkovým počtem výskytů třídy a celkovým počtem záznamů. Tento podíl se rovnou vynásobí vypočteným mezivýsledkem a přičte se ke konečnému výsledku, který je na počátku nulový a je uložen ve struktuře Prediktor. VIZ OBRÁZEK



Obrázek 14 Vývojový diagram podmíněné entropie

Metoda pro výpočet informačního zisku má stejný základ, jako předchozí výpočet. Předně je zde volána metoda pro výpočet entropie, ale dále se ještě pokračuje průchodem všech prediktorů a počítá se zde rozdíl mezi vypočtenou hodnotou predikovaného atributu a prediktorem. Výsledek je opět uložen v struktuře Prediktor.

4.4.2 Výběr nejlepšího prediktora a rozdělení uzlu

Jedná se zde o dvě metody. První pro nalezení nejlepšího prediktora, kterého určuje podle napočítaných hodnot a druhá metoda, která podle konkrétního vybraného prediktora rozdělí uzel na další uzly podle tříd tohoto prediktora.

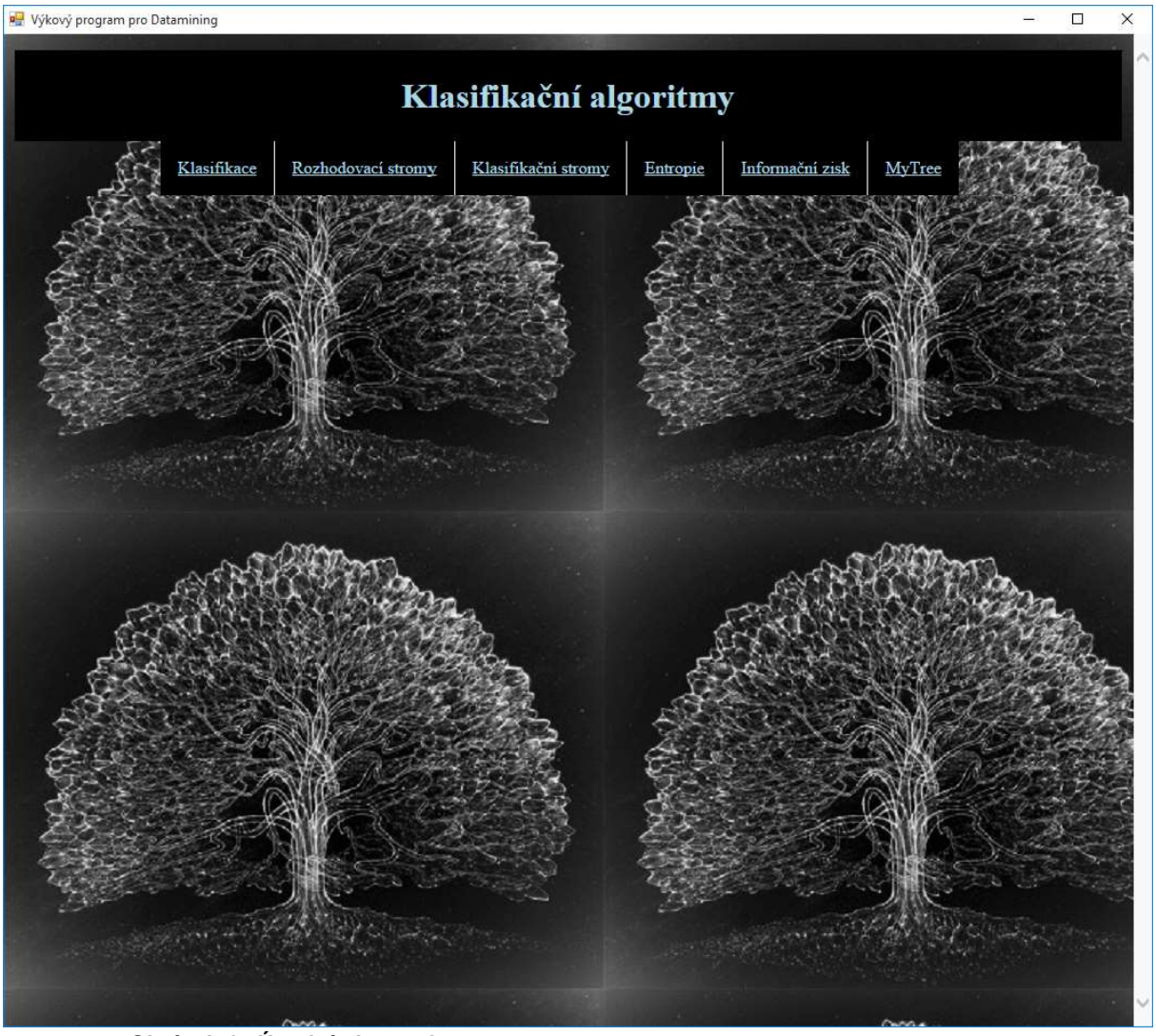
Pro nalezení nejlepšího prediktora je nutné projít všechny dosud nepoužité prediktory pro konkrétní uzel a přitom se dívat na jejich napočítané hodnoty z předchozích metod. Je nutné vědět, že nejlepší prediktor po výpočtu entropie je ten, který má nejmenší hodnotu, protože hledáme prediktora s nejmenší mírou neuspořádanost. Naopak pro výběr nejlepšího prediktora po výpočtu informačního zisku, hledáme takového prediktora, který má hodnotu nejvyšší. Je to tedy ten, který nás nejvíce informuje.

V průběhu cyklu je tedy vybraným prediktorům nastavena hodnota typu boolean, díky které víme o prediktoru, že je nejlepším v tomto uzlu.

Druhá metoda pro rozdělení podle vybraného prediktora vezme toho označeného prediktora a začne procházet jeho třídy. Pro každou jeho třídu vytvoří nový uzel a tomu začne nastavovat všechny příslušné proměnné. Tedy musí projít i současnou tabulku dat vybraného prediktora a pro konkrétní třídu nového uzlu uložit ty data se stejnou třídou. Nakonec se pro nový uzel překopírují všichni prediktoři z děleného uzlu a vybraný prediktor se v tomto novém uzlu označí jako již použitý.

4.5 Jak na aplikaci

Úvodní okno aplikace shrnuje v krátkosti teorii o klasifikaci, rozhodovacích a klasifikačních stromech, pojmech entropie a informační zisk. Říká, co jednotlivé pojmy znamenají, kde se využívají a s jakými daty, jak se počítají a podobně. Tato část je ve windows okně, ale uvnitř se jedná o webovou aplikaci o jedné stránce, kde jednotlivé odkazy skrývají a odkrývají jim určené bloky textu pomocí javascriptu. Po kliknutí na odkaz „MyTree“ se spustí druhá aplikace, která vytváří rozhodovací strom.

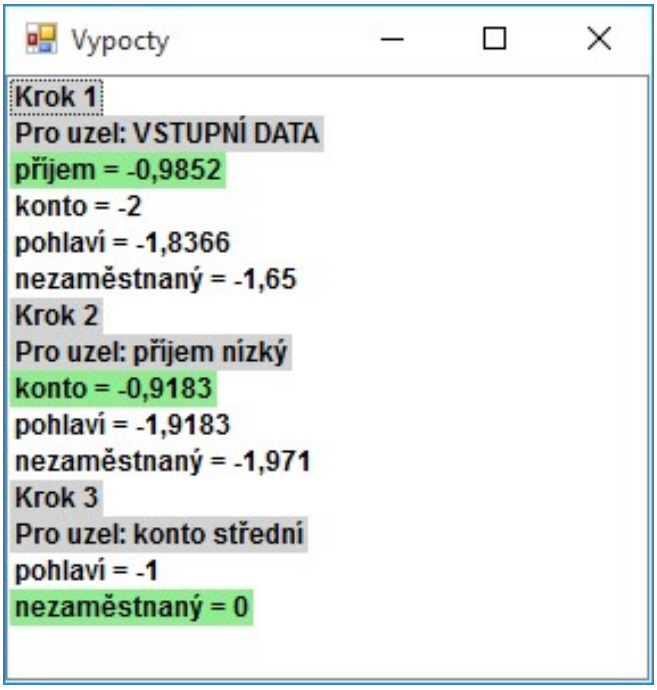


Obrázek 15 Úvodní obrazovka

V části „MyTree“ si uživatel může pomocí menu na liště nahrát vlastní datový soubor. Je zde omezení. Datové soubory mohou být pouze typu csv, kde jsou hodnoty oddělené středníkem a důležitým bodem je, aby takový soubor měl v sobě hlavičku k datům. Aplikace sice při nahrávání dat bez hlavičky nijak nezakolísá, nicméně do zobrazené tabulky nahraje jako hlavičku první řádek datového souboru.

Aplikace po nahrání souboru zobrazí v prvním listboxu vlevo seznam atributů, které nalezl v hlavičce (na prvním řádku datového souboru). Také po úspěšném nahrání souboru zobrazí aplikace vpravo od hlavního okna další úzké okno pro seznam tlačítek (uzlů stromu), které se budou při stavbě stromu tvořit. V aplikaci je nazváno jako „Halda“. Po kliknutí na jedno z těchto tlačítek se vykreslovací plátno přesune na pozici konkrétního uzlu stromu. Zároveň se zobrazením toho okna se do plátna vykreslí první uzel se vstupními daty.

Nahrané atributy lze pomocí tlačítek vedle listboxů prohazovat v libovolném množství. Pod prvním listboxem jsou další dva. První, větší, slouží pro nastavení prediktorů, které jsou nezbytné pro spuštění algoritmů. Stejně ten poslední listbox, jednořádkový, musí mít nastavenou pouze jednu hodnotu, kterou lze nastavit pomocí



Obrázek 16 Výpočty

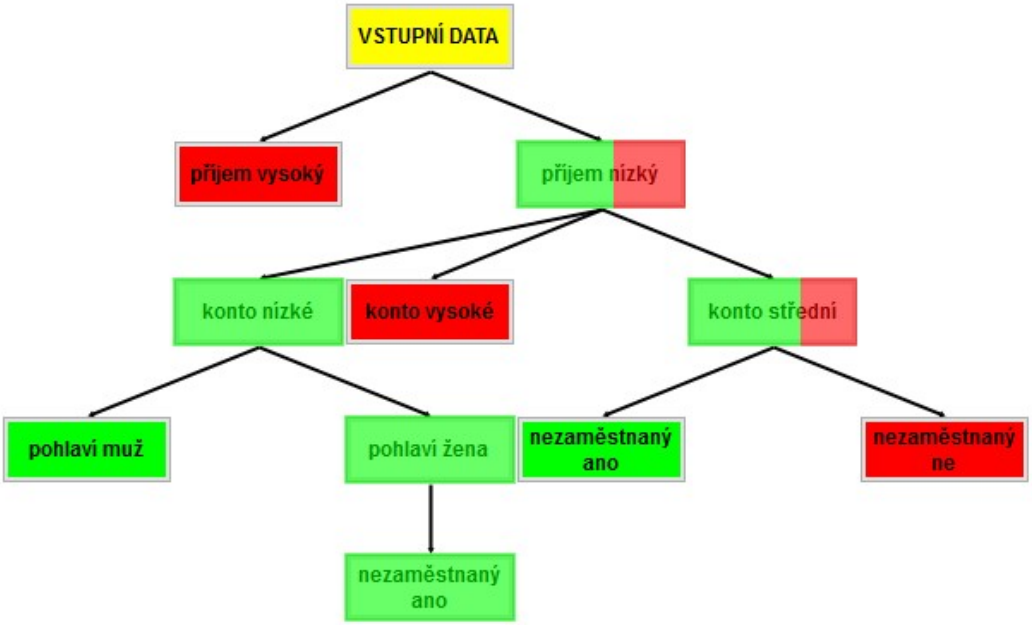
tlačítka „NASTAV PREDIKOVANÝ ATRIBUT“. V menu na liště si také může uživatel v combo boxu zvolit, jakým způsobem chce rozdělovat data do stromu. Zda pomocí entropie, nebo informačního zisku.

Takto nastavená aplikace je již spustitelná. O čemž dávají zprávu i zelené, případně růžové bloky ve stavovém řádku. Po kliknutí na tlačítko „SPUST“ se spustí příslušný algoritmus a aplikace začne vykonávat operace. Po skončení se vykreslí celý strom na plátno.

Jednotlivé uzly stromu jsou obarveny. Hlavní uzel je vždy žlutý. U ostatních uzlů si může v nastavení aplikace uživatel nastavit, jakou ze základních barev se budou uzly obarvovat. V aplikaci se neuvažuje více než 4 třídy pro predikovaný atribut, a tak lze navolit pouze 4 barvy. Každá barva je pak přidělena konkrétní třídě, a tak lze přibližně sledovat v jakém počtu zastoupení jsou třídy v datové množině uzlu.

V menu na liště v záložce Soubor je kromě tlačítka otevřít ještě právě tlačítko nastavení a tlačítko nápovědy a v záložce Zobrazení lze nalézt tlačítka pro přesun na střed plátna a pro otevření oken Úvodní okno, Halda a Výsledky.

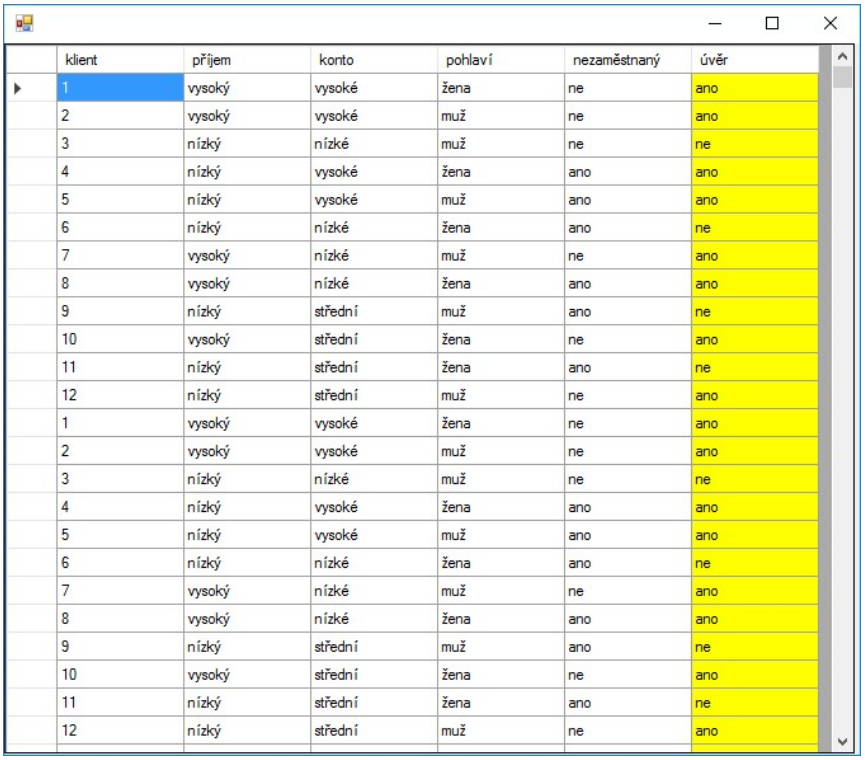
V nastavení aplikace je dále možnost nastavit velikost písma, která se změní až po restartu aplikace, dále se zde dají zapnout či vypnout plovoucí nápovědy o tom, co jaká komponenta dělá, a samozřejmě již zmíněné nastavování barev. Kliknutím na čtvereček barvy se otevře okno pro výběr barev. Všechna nastavení se při zavření okna ukládají do souboru „nastavení.ini“ a v aplikaci, kromě písma, se změna projeví ihned.



Obrázek 17 Vytvořený strom

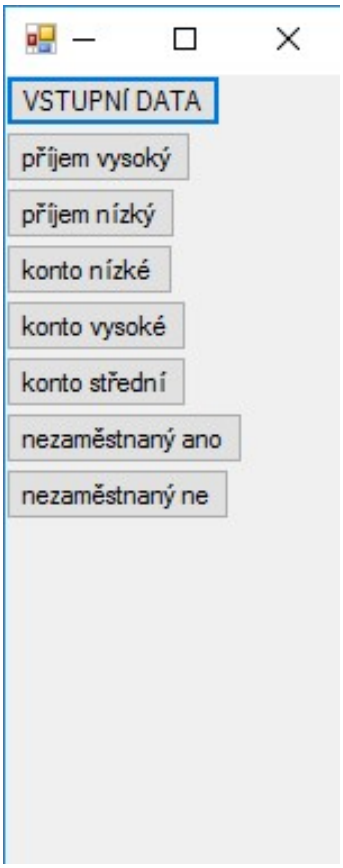
Uzly stromu jsou tedy klasické butony. Po kliknutí na některý z nich se zobrazí nové okno s datovou tabulkou a příslušnými daty k uzlu. Uzel je vždy pojmenován prediktorem a třídou podle které byl rozdělen. Zároveň s tím se ještě vpravo od tohoto okno zobrazí další okno, kde jsou zobrazeny jednotlivě četnosti tříd ještě nepoužitých prediktorů. Po spuštění algoritmu a kliknutí na hlavní uzel se ještě, krom zmíněných oken, zobrazí po levé straně okno s tabulkou, která zobrazuje pro každý atribut všechny jeho třídy a jakou má atribut v algoritmu roli. Tedy zda je stále jen nepoužitým atributem, nebo prediktorem nebo predikovaným atributem.

Po skončení algoritmu si můžeme v menu zobrazení otevřít okno pro výpočty. Zde se po krocích zobrazují všechny konečné výsledky pro každého prediktora a zároveň zobrazuje, jaký prediktor byl vybrán pro další dělení uzlu. Ten je označen vždy zeleně.

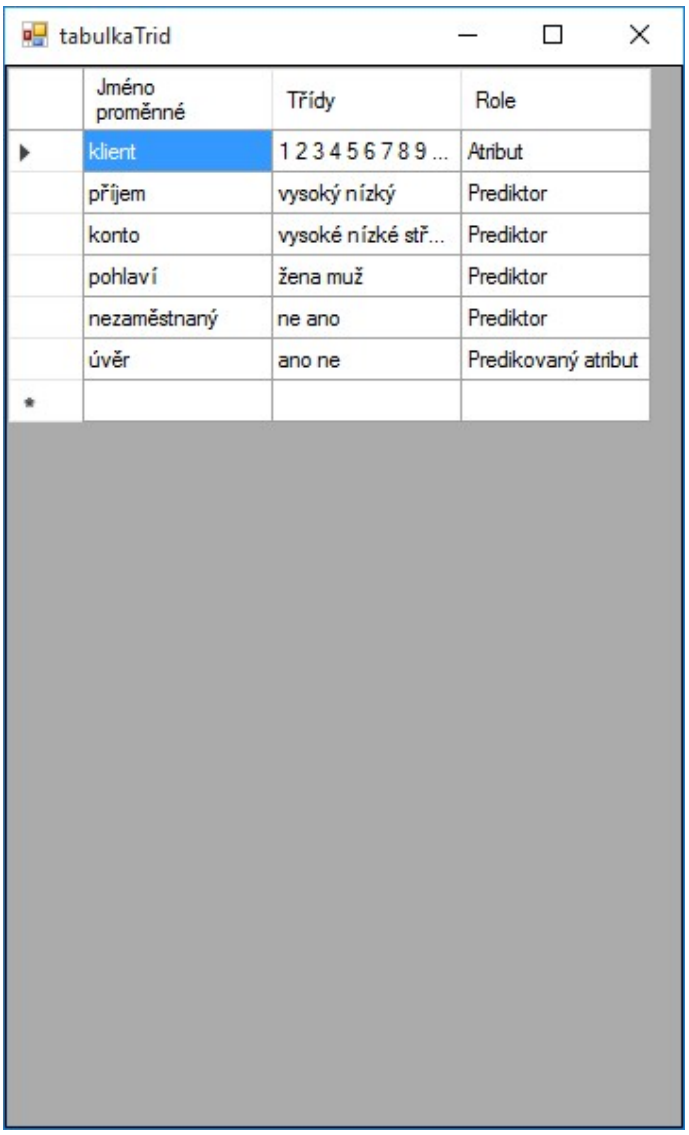


klient	příjem	konto	pohlaví	nezaměstnaný	úvěr
1	vysoký	vysoké	žena	ne	ano
2	vysoký	vysoké	muž	ne	ano
3	nízký	nízké	muž	ne	ne
4	nízký	vysoké	žena	ano	ano
5	nízký	vysoké	muž	ano	ano
6	nízký	nízké	žena	ano	ne
7	vysoký	nízké	muž	ne	ano
8	vysoký	nízké	žena	ano	ano
9	nízký	střední	muž	ano	ne
10	vysoký	střední	žena	ne	ano
11	nízký	střední	žena	ano	ne
12	nízký	střední	muž	ne	ano
1	vysoký	vysoké	žena	ne	ano
2	vysoký	vysoké	muž	ne	ano
3	nízký	nízké	muž	ne	ne
4	nízký	vysoké	žena	ano	ano
5	nízký	vysoké	muž	ano	ano
6	nízký	nízké	žena	ano	ne
7	vysoký	nízké	muž	ne	ano
8	vysoký	nízké	žena	ano	ano
9	nízký	střední	muž	ano	ne
10	vysoký	střední	žena	ne	ano
11	nízký	střední	žena	ano	ne
12	nízký	střední	muž	ne	ano

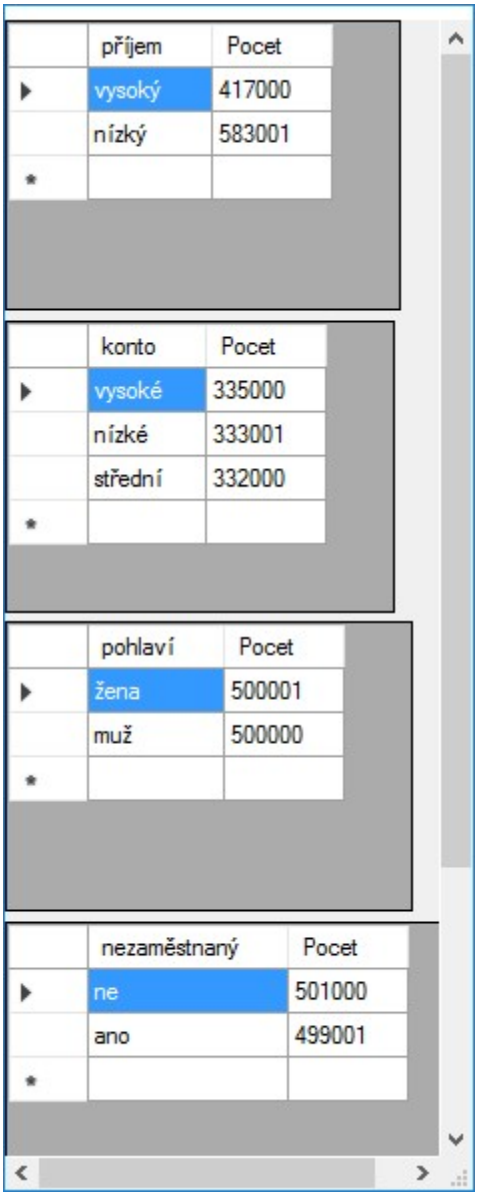
Obrázek 18 Zobrazená vstupní data



Obrázek 19 Halda



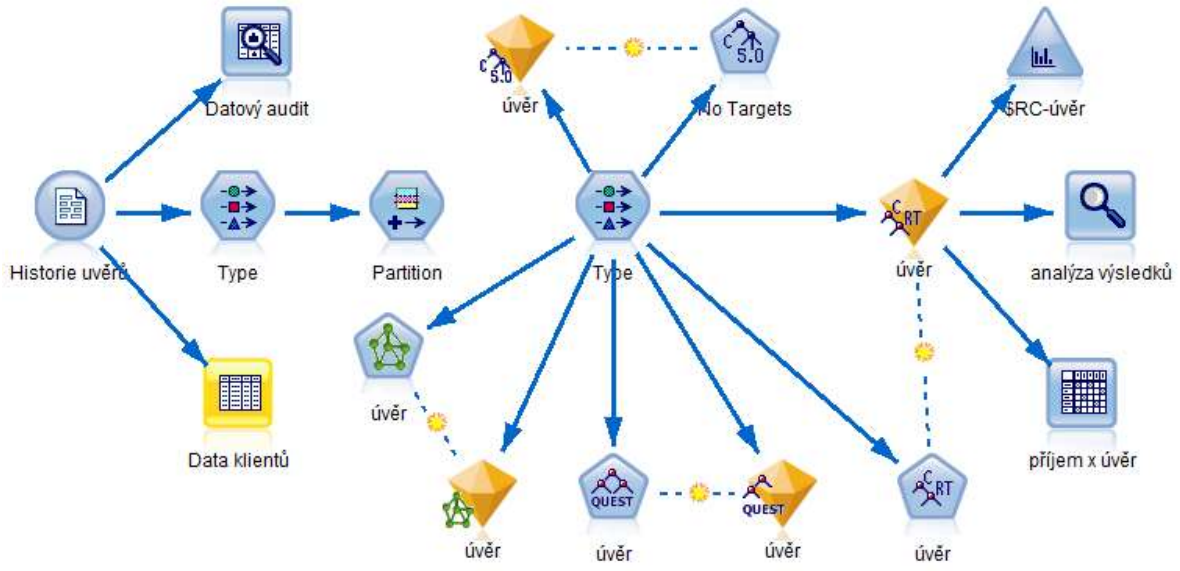
Obrázek 20 Tabulka tříd pro vstupní data



Obrázek 21 Četnosti prediktorů pro vstupní data

5. Ověření aplikace

Pro porovnání budování stromu moji aplikací s klasifikační úlohou v Modeleru, jsme postavili v Modeleru dva proudy nad stejnými daty jako v naší aplikaci. V prvním proudu byly použité modelovací uzly C5.0, CART (někdy také C&RTree) a Chaid a proud byl uveden na Obrázek 2 Příprava dat. Druhý proud Obrázek 22 Proud pro modelování poskytování úvěrů II. sloužil k testování více algoritmů: Quest, který staví binární strom, neuronové sítě Neural net, CART a C5.0. Poslední dva uzly už byly použité v prvním proudu, ale tady jsme navíc testovali budování bagging modelů. Jedná se o postup, který navrhl 1994 Leo Breitman, kterým se zvyšuje úspěšnost klasifikace. Uzlu se nastaví počet modelů, pro které se z datové množiny generuje trénovací množina a modely řeší úlohu kolektivně podle nastavitelných parametrů. Sledovali jsme čas budování klasifikačních stromů, kdy základní data měla milión záznamů, z nichž byla testovací množina náhodně generovaná jako 50%:50% (testovací - trénovací) a počet modelů byl u všech uzlů nastaven na 10. Čas budování modelu se očekávaně zvýšil, ale ono zvýšení u C5.0 bylo nečekaně malé a naopak zvýšení u neuronové sítě velmi velké (Tabulka 6 Výsledky klasifikace v Modeleru). Hledání vysvětlení tohoto výsledku by mohlo být dalším pokračováním této práce.



Obrázek 22 Proud pro modelování poskytování úvěrů II.

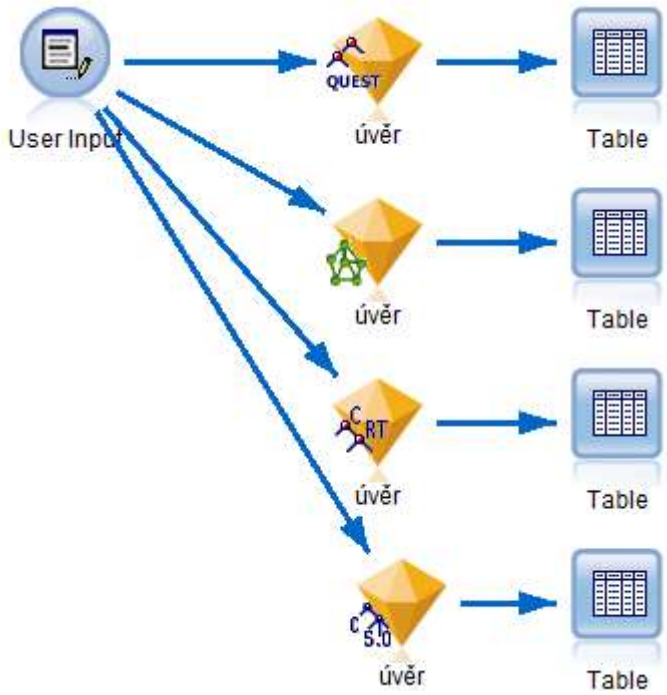
Shoda klasifikace na „novém případu“ se bagging postupem posílila. Jednoduché modely se shodovaly ve 3 případech a jednou Quest rozhodoval opačně, modely s baggingem rozhodovaly shodně.

Použitý klasifikační model	Doba budování základního modelu v m:ss	Shoda v predikci pro základní model	Doba budování Bagging modelu v m:s	Shoda v predikci
C5.0	0:17	ANO	1:32	ANO
CART	0:14	ANO	9:26	ANO
Quest	0:13	NE	9:15	ANO
Neuronová síť	1:27	ANO	12:23	ANO

Tabulka 6 Výsledky klasifikace v Modeleru

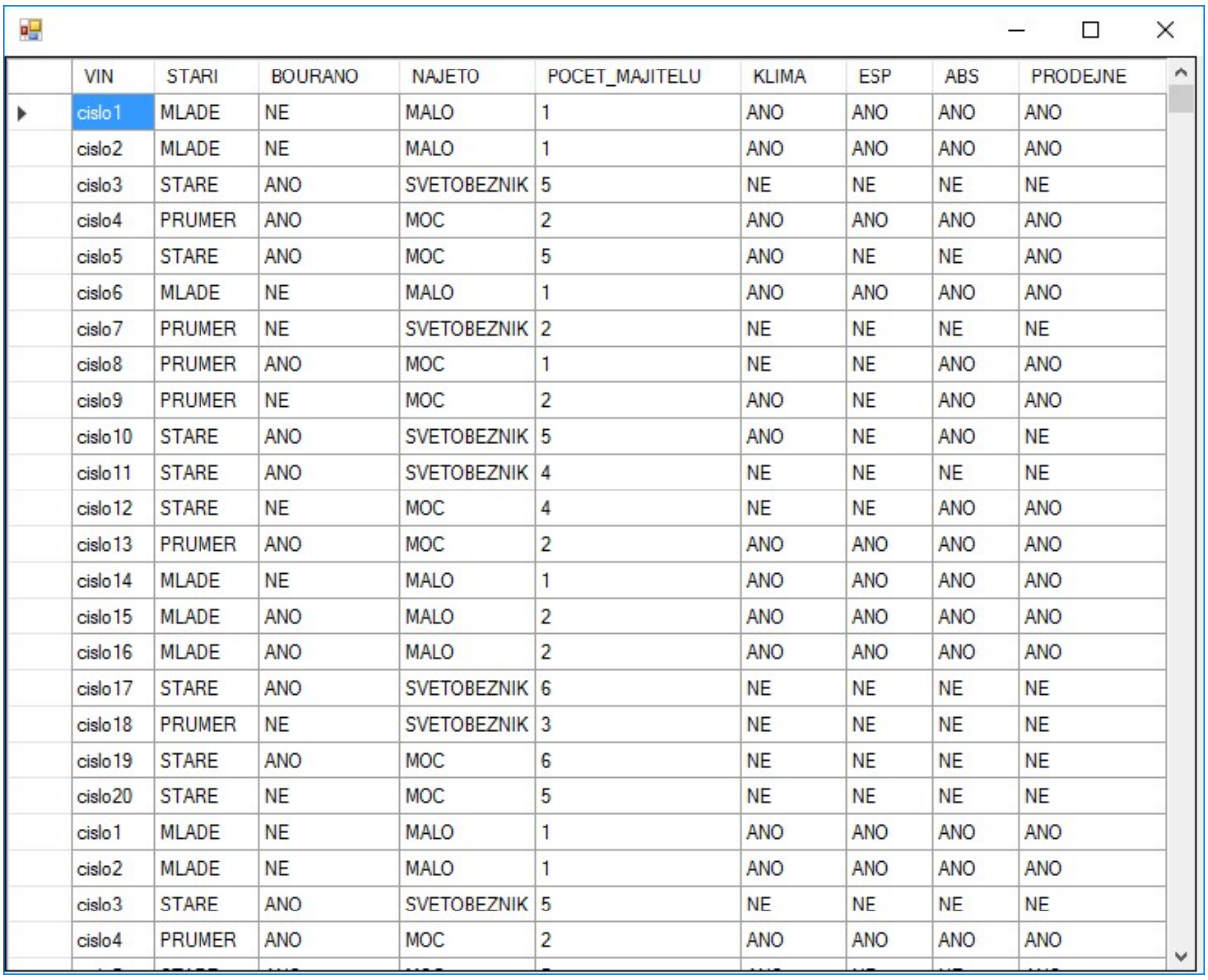
Výběr prediktorů pro jednotlivé modely je v příloze 2

Nasazení modelu pro neznámý případ a porovnání klasifikace je na Obrázek 23 . K vypočítanému modelu se přidá uzel Input, do kterého vložíme případ, který se má klasifikovat a výsledek klasifikace zobrazíme „reportem“ nebo tabulkou. Modely po baggingu a naše aplikace MyTree se v klasifikaci shodly.



Obrázek 23 Stream pro implementaci modelu – nasazení do praxe

Další ověřování aplikace MyTree proběhlo na experimentálních datech pro prodej bazarových aut. Vstupní data byla podle podmínek zvolena jako kategoriální, uložena jako auta.csv souboru, kde hodnoty jsou odděleny středníky a první řádek je hlavička tabulky, která říká, co jaký atribut reprezentuje. Načtení a zobrazení dat je vidět na obrázku.



	VIN	STARI	BOURANO	NAJETO	POCET_MAJITELU	KLIMA	ESP	ABS	PRODEJNE
▶	cislo1	MLADE	NE	MALO	1	ANO	ANO	ANO	ANO
	cislo2	MLADE	NE	MALO	1	ANO	ANO	ANO	ANO
	cislo3	STARE	ANO	SVETOBEZNIK	5	NE	NE	NE	NE
	cislo4	PRUMER	ANO	MOC	2	ANO	ANO	ANO	ANO
	cislo5	STARE	ANO	MOC	5	ANO	NE	NE	ANO
	cislo6	MLADE	NE	MALO	1	ANO	ANO	ANO	ANO
	cislo7	PRUMER	NE	SVETOBEZNIK	2	NE	NE	NE	NE
	cislo8	PRUMER	ANO	MOC	1	NE	NE	ANO	ANO
	cislo9	PRUMER	NE	MOC	2	ANO	NE	ANO	ANO
	cislo10	STARE	ANO	SVETOBEZNIK	5	ANO	NE	ANO	NE
	cislo11	STARE	ANO	SVETOBEZNIK	4	NE	NE	NE	NE
	cislo12	STARE	NE	MOC	4	NE	NE	ANO	ANO
	cislo13	PRUMER	ANO	MOC	2	ANO	ANO	ANO	ANO
	cislo14	MLADE	NE	MALO	1	ANO	ANO	ANO	ANO
	cislo15	MLADE	ANO	MALO	2	ANO	ANO	ANO	ANO
	cislo16	MLADE	ANO	MALO	2	ANO	ANO	ANO	ANO
	cislo17	STARE	ANO	SVETOBEZNIK	6	NE	NE	NE	NE
	cislo18	PRUMER	NE	SVETOBEZNIK	3	NE	NE	NE	NE
	cislo19	STARE	ANO	MOC	6	NE	NE	NE	NE
	cislo20	STARE	NE	MOC	5	NE	NE	NE	NE
	cislo1	MLADE	NE	MALO	1	ANO	ANO	ANO	ANO
	cislo2	MLADE	NE	MALO	1	ANO	ANO	ANO	ANO
	cislo3	STARE	ANO	SVETOBEZNIK	5	NE	NE	NE	NE
	cislo4	PRUMER	ANO	MOC	2	ANO	ANO	ANO	ANO

Obrázek 24 vstupní data - auta

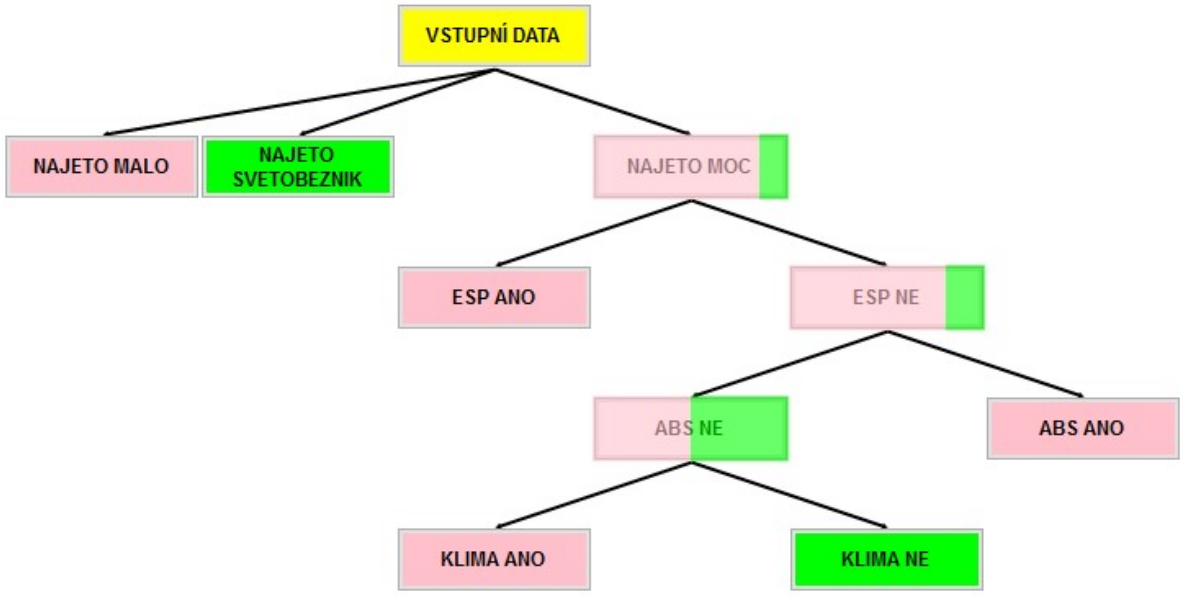
Při výběru prediktorů pro klasifikaci byl zavržen atribut „VIN“. Jedná se o unikátní atribut, který je vždy jiný, a tedy je pro klasifikaci nepoužitelný. Ostatní atributy byly tedy zvoleny jako prediktoři a za predikovaný atribut byl zvolen atribut „Prodejné“. Výsledný strom má tedy klasifikovat auta podle toho, zda jsou pro majitele bazaru ještě prodejné, či nikoli. Po spuštění algoritmů byl vytvořen strom, který vidíte na obrázku.

ATRIBUT	TYP	HODNOTY	ROLE
VIN	číslo	-	NONE
STÁŘÍ	ORDINAL	Mladé, průměr, staré	INPUT
BOURÁNO	FLAG	Ano, ne	INPUT
NAJETO	ORDINAL	Malo, moc, světoběžník	INPUT
POČET_MAJITELŮ	ORDINAL	1, 2, 3, 4, 5, 6	INPUT
KLIMA	FLAG	Ano, ne	INPUT
ESP	FLAG	Ano, ne	INPUT
ABS	FLAG	Ano, ne	INPUT
PRODEJNÉ	FLAG	Ano, ne	TARGET

Tabulka 7 Typy atributů pro data Auta.csv

Algoritmus tedy ani nemusel použít všechny prediktory.

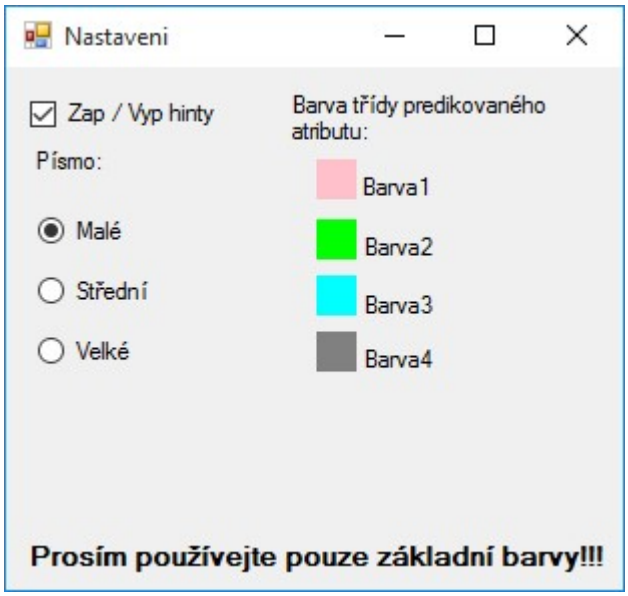
Můžeme tedy říci, že aplikace je schopná načíst jakákoli data, která splňují aplikační podmínky – csv soubor s hodnotami oddělenými středníkem, na prvním řádku je hlavička tabulky, data jsou kategoriální.



Obrázek 25 Strom auta

Pro zobrazený strom vidíme různě obarvené uzly. Již na počátku spuštění algoritmu, aplikace nastaví každé třídě predikovaného atributu konkrétní barvu. Tuto barvu si třída pamatuje až do konce stromu. Velikost plochy obarvení se liší pouze v tom, jak moc je konkrétní třída v uzlu zastoupená. Uživatel si v nastavení aplikace může tyto barvy změnit, ale pro aplikaci jsou zde uvažovány maximálně 4 barvy.

Obarvené uzly tedy neříkají, zda je auto prodejné. To lze vidět po kliknutí na uzel. Při pohledu na strom můžeme tedy jednoznačně říci, že data byla spolehlivě rozdělena, protože list stromu je vždy obarven jednou barvou.



Obrázek 26 Nastavení aplikace MyTree

6. Závěr

Cílem bakalářské práce bylo seznámit se s dataminingem, který stále více vstupuje do povědomí veřejnosti. Seznámit se s SPSS Modelerem od firmy IBM, který je komplexním datamingovým nástrojem, a který jako jeden z mála dokáže díky své robustnosti a rozsáhlasti zahrnout většinu dataminingových úloh a hlavně realizovat celý projekt v metodologii CRISPDM. Problém nastává v komerčnosti Modeleru. V současné době společnost IBM poskytuje trial verzi, kterou si lze po registraci stáhnout z jejich webových stránek. Kromě trialové verze jsem měl možnost pracovat v počítačové učebně, kde je pro výzkum a výuku Modeler nainstalovaný.

Dále práce pokračuje seznámením se s klasifikačním problémem v dataminingových úlohách. Je zde popsána nejen klasifikace, ale také to, jaké úlohy lze pomocí klasifikace řešit a jakými způsoby. Výčet nemůže být úplný, protože téma je velmi rozsáhlé. V práci jsme se zaměřili především na rozhodovací a klasifikační stromy. Pro klasifikaci je možné použít celou řadu dalších postupů, ze kterých jsem se zabýval ještě shlukovou analýzou. Při budování konkrétní dataminingové úlohy v IBM SPSS Modeleru jsem pro klasifikaci použil i další algoritmy, které mohou být pro klasifikaci velmi dobře použité. Jsou k dispozici v hotových modulech, kterým se v Modeleru říká uzly a mohou být použité bez hluboké znalosti „skrytého“ algoritmu. V praxi to znamená, že postavit proud lze i po experimentální zkušenosti. Výklad výsledků a ladění modelů potřebuje ale další znalosti. Takto jsem pro porovnání vložil do proudu neuronovou síť s defaultním nastavením a zabýval jsem se jen výsledkem klasifikace. Různé algoritmy nakonec dospěly ke stejné klasifikaci pro neznámý případ. Použité uzly C5.0, CART, Quest a Neural net pro klasifikaci jsem testoval na různých datech a také testoval čas výpočtu. Výsledek ukazuje Tabulka 6. Ukázalo se, že při nasazení základního modelu se Quest v predikci lišil od ostatních. Teprve nastavením bagging modelů – 10 spolupracujících modelů pro každý uzel, jsem dosáhl shodné klasifikace. Doba budování modelu výpočtem se násobně zvýšila.

Pro budování klasifikačního stromu vlastním programem, který jsem nazval MyTree, byly vybrány algoritmy založené na pojmech z teorie informace, konkrétně na entropii a informačním zisku. Oba algoritmy byly naprogramovány do aplikace, která zobrazuje vznik rozhodovacího stromu a poskytuje výklad potřebných pojmů k porozumění budování stromů pro klasifikaci. Tato aplikace je k dispozici studentům na e-learningovém portále ALS. Vše lze realizovat pohodlně Modelerem, ale

algoritmy „ukryté“ v modelovacích uzlech z Modeleru lze poznat obtížně. MyTree má doplnit studentům potřebné informace k plnému pochopení algoritmů a vizualizací výsledku pochopení podpořit.

Aplikaci MyTree jsem testoval na různých datových sadách. Použitá data podobného charakteru byla vždy v souboru csv a musela v prvním řádku obsahovat hlavičku. Byla použita také data o různé velikosti. Největší data, která jsem testoval, měla jeden milion řádků. MyTree tyto data zpracovala během půl minuty, což je rychlejší než neuronová síť v Modeleru, ale pomalejší než odpovídající C5.0.

Aplikace má sloužit pouze jako učební pomůcka studentům dataminingového kurzu a je zařazená v kurzu DM2015.

7. Zdroje informací

[1] Hendl J.: Přehled statistických metod zpracování dat, Portál, s. r. o. 2006

[2] Olivia Parr Rud: Datamining, Computer Press, a.s. 2006

[3] http://www.msps.cz/data-mining/

[4] Schönberger V. M., Cukier K.: Big Data, Comperter Press. Brno, 2014

[5] Berka P.: Dobývání znalostí z databází, Academia, 2003

[6] http://www.spss.cz/pasw_modeler.htm

[7] http://www.algoritmy.net/article/104/Strom

[8] https://algoritmy.net/article/1399/Prohledavani-do-sirky

[9] Klaschka J., Kotrč E.: Klasifikační a regresní lesy, Robust, 2004

8. Seznam obrázků

9. Seznam tabulek

10. Příloha 1

Pizza - podobnost podle asociačních koeficientů

	smetana	eidam	ementál	uzený sýr	niva	šunka	žampiony
O1	1	1	0	1	1	0	1
O2	1	1	0	0	0	1	1
O3	1	0	1	0	0	1	0
O4	0	1	1	0	0	0	1
O5	1	0	0	1	1	0	1

tuňák	olivy	slanina	Název
0	0	0	Sýrová
1	1	0	Speciál
1	0	1	Bača
1	0	1	Dáša
1	0	0	Uzená

Četnosti shod

	O1		O2		O3	
O1	5	0	3	3	1	4
	0	5	2	2	4	1
O2	3	3	6	0	3	3
	2	2	0	4	2	2
O3	1	4	3	2	5	0
	4	1	3	2	0	5
O4	2	3	3	2	3	2
	3	2	3	2	2	3
O5	4	1	3	2	2	3
	1	4	3	2	3	2

O4		O5	
2	3	4	1
3	2	1	4
3	3	3	3
2	2	2	2
3	2	2	3
2	3	3	2
5	0	2	3
0	5	3	2
2	3	5	0
3	2	0	5

Jaccardův koeficient a/(a+b+c)

	O1	O2	O3	O4	O5
O1	1,000				
O2	0,375	1,000			
O3	0,111	0,375	1,000		
O4	0,250	0,375	0,429	1,000	
O5	0,667	0,375	0,250	0,250	1,000

čím větší číslo, tím podobnější objekty

Sokal-Michenerův koeficient
(a+b)/(a+b+c+d)

	O1	O2	O3	O4	O5
O1	1,000				
O2	0,500	1,000			
O3	0,200	0,500	1,000		
O4	0,400	0,500	0,600	1,000	
O5	0,800	0,500	0,400	0,400	1,000

Russel-Raoův koeficient
a/(a+b+c+d) co je relativní četnost shody 1

	O1	O2	O3	O4	O5
O1	0,500				
O2	0,300	0,600			
O3	0,100	0,300	0,500		
O4	0,200	0,300	0,300	0,500	
O5	0,400	0,300	0,200	0,200	0,500

Diceův koeficient
2*a/(2*a+b+c)

	O1	O2	O3	O4	O5
O1	1,000				
O2	0,750	1,000			
O3	0,200	0,545	1,000		
O4	0,400	0,545	0,600	1,000	
O5	0,800	0,545	0,400	0,400	1,000

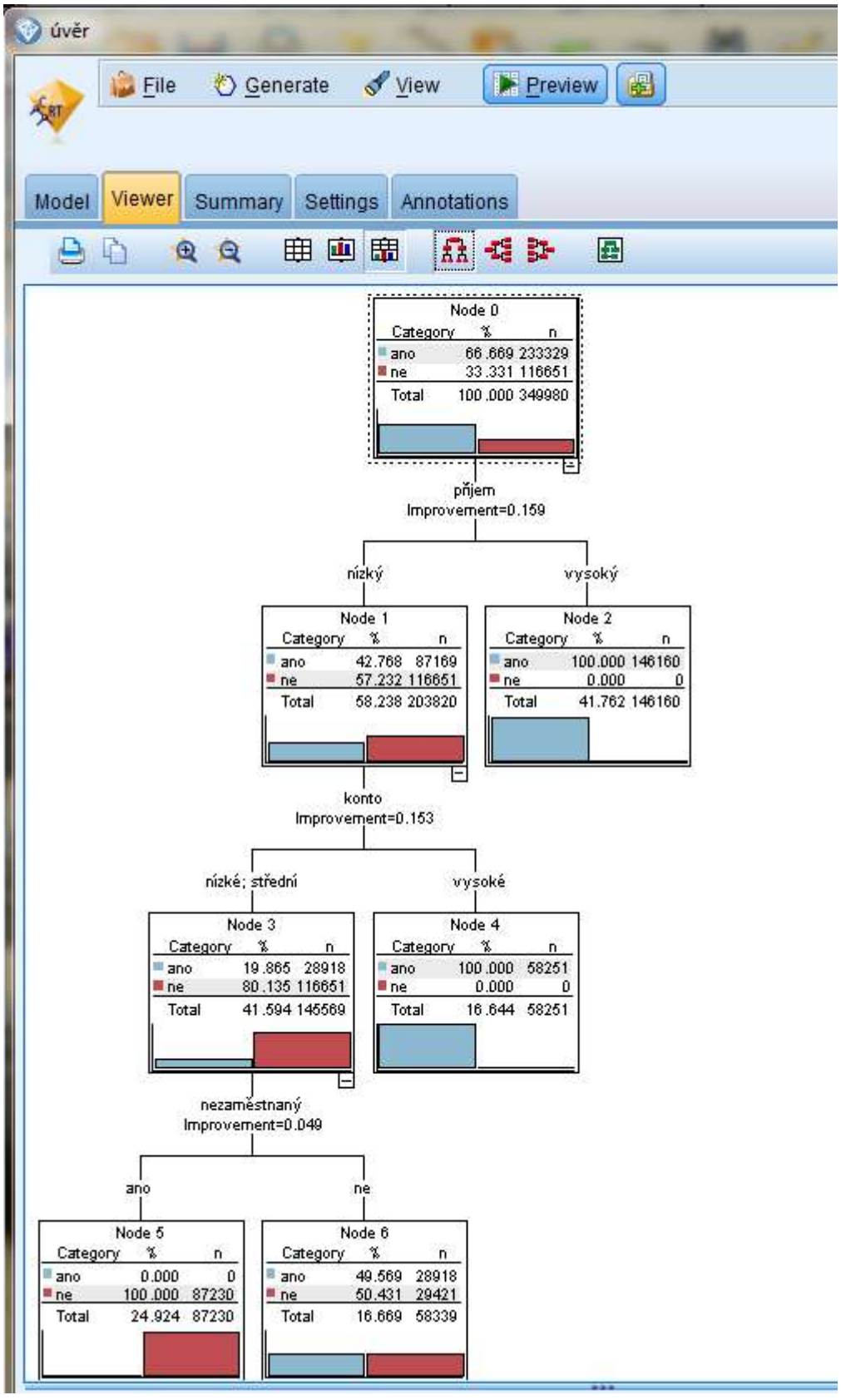
Rogers-Tanimotoův koeficient					
(a+d)/(a+d+2*(b+c)					
	O1	O2	O3	O4	O5
O1	1,000				
O2	0,333	1,000			
O3	0,111	0,333	1,000		
O4	0,250	0,333	0,429	1,000	
O5	0,667	0,333	0,250	0,250	1,000

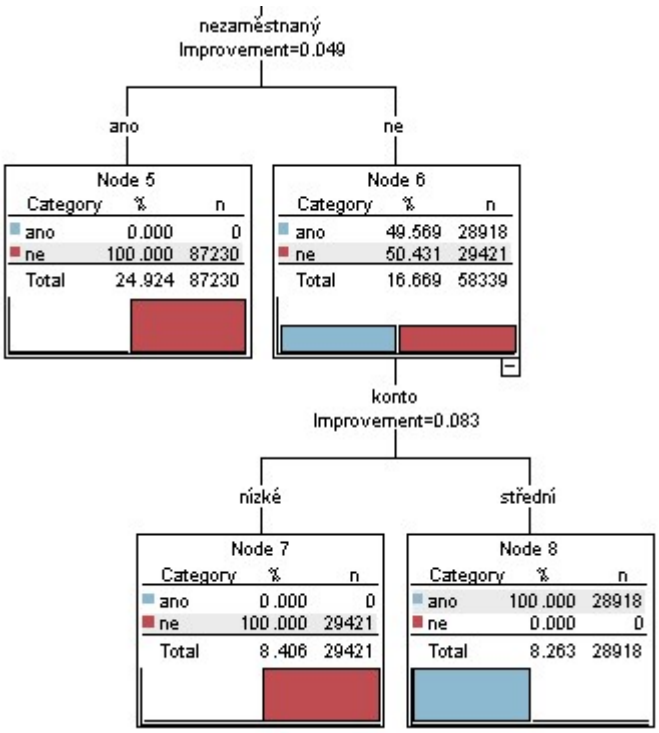
Hamannův koeficient					
(a+d-(b+c))/(a+b+c+d)					
	O1	O2	O3	O4	O5
O1	1,000				
O2	0,000	1,000			
O3	-0,600	0,000	1,000		
O4	-0,200	0,000	0,200	1,000	
O5	0,600	0,000	-0,200	-0,200	1,000

11. Příloha 2

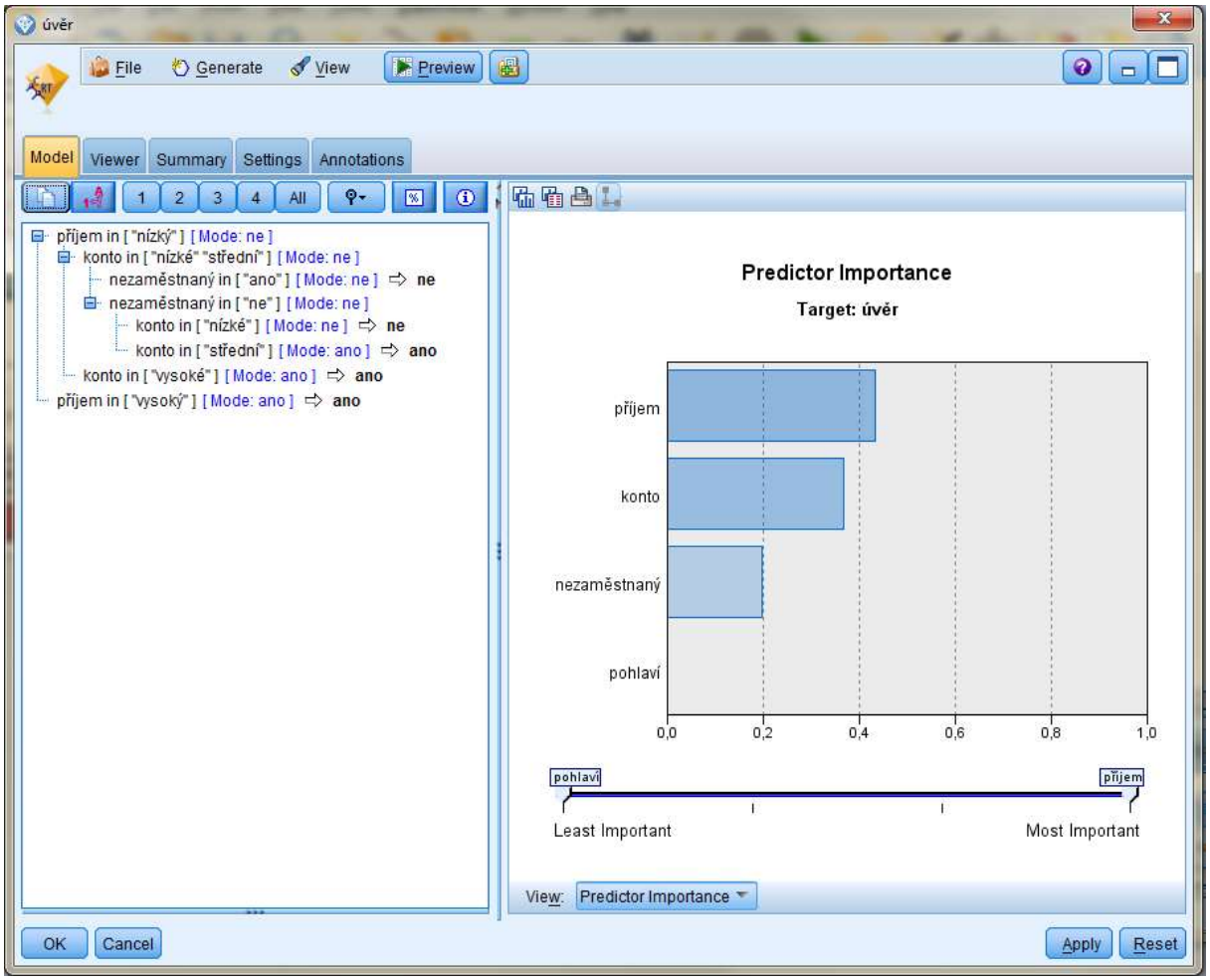
Výsledky modelů v Modeleru

Výsledný strom:




nezaměstnaný
Improvement=0.049
ano
ne
Node 5
Category % n
ano 0.000 0
ne 100.000 87230
Total 24.924 87230
Node 6
Category % n
ano 49.569 28918
ne 50.431 29421
Total 16.669 58339
konto
Improvement=0.083
nízké
střední
Node 7
Category % n
ano 0.000 0
ne 100.000 29421
Total 8.406 29421
Node 8
Category % n
ano 100.000 28918
ne 0.000 0
Total 8.263 28918

Predikce pro modul CART:



Predikce pro neuronovou síť: