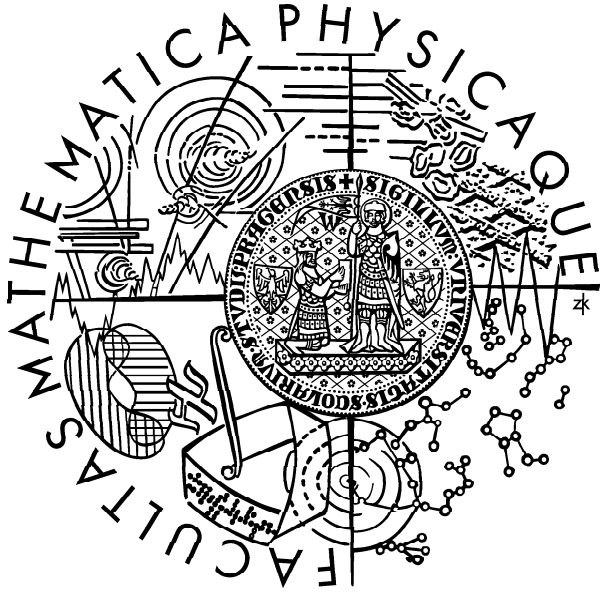
Univerzita Karlova v Praze

Matematicko-fyzikální fakulta

DIPLOMOVÁ PRÁCE



Peter Lacký

Teoretické způsoby modelování uživatelského rozhodování

Katedra softwarového inženýrství

Vedoucí diplomové práce: Prof. RNDr. Peter Vojtáš, DrSc.

Studijní program: Informatika

2010

Na tomto mieste by som rád poďakoval vedúcemu diplomovej práce Prof. RNDr. Petrovi Vojtášovi, DrSc. a konzultantovi Mgr. Alanovi Eckhardtovi za ich rady a pripomienky, ktoré mi pomohli pri vytváraní tejto práce. Ďalej ďakujem rodičom za podporu počas celej doby štúdia, RNDr. Matúšovi Ondrejčkovi, Mgr. Petrovi Bubelínymu a Erikovi Horničákovi za pomoc a konzultáciu tejto práce a Zuzane Grešákovej za celkovú korekciu práce.

Prehlasujem, že som svoju diplomovú prácu napísal samostatne a výhradne s použitím citovaných prameňov. Súhlasím so zapožičiavaním práce.

V Prahe dňa 16. apríla 2010 Peter Lacký

Obsah

Název práce: Teoretické způsoby modelování uživatelského rozhodování

Autor: Peter Lacký

Katedra (ústav): Katedra softwarového inženýrství

Vedoucí diplomové práce: Prof. RNDr. Peter Vojtáš, DrSc.

e-mail vedoucího: Peter.Vojtas@mff.cuni.cz

Abstrakt: Táto práca sa zaoberá problematikou modelovania užívateľských preferencií. Obsahuje rozbor rozdielnych pohľadov na užívateľské preferencie a prehľad súčasných modelov užívateľských preferencií. V práci je navrhutý model vychádzajúci z pravdepodobnostného rozdelenia, ktorý je porovnaný s modelom založeným na teórii Suppor Vector Regression. V závere práce sú uvedené návrhy na rozšírenie jednotlivých modelov.

Klíčová slova: užívateľské preferencie, pravdepodobnostné modely užívateľských preferencií, pravdepodobnostné rozhodovanie

Title: Theoretical aspect of modelling of user decision

Author: Peter Lacký

Department: Department of Software Engineering

Supervisor: Prof. RNDr. Peter Vojtáš, DrSc.

Supervisor's e-mail address: Supervisor's e-mail address: Peter.Vojtas@mff.cuni.cz

Abstract: This work concerns the problem of modelling of users preferencies. It contains different points of view on users preferencies and a summary of present models of users preferencies. There is suggested a model based on a probability distribution, which is compared with the model based on a theory of Suppor Vector Regression. At the end of the work, there are mentioned options, how to extend and develop particular models.

Keywords: user preferences, probabilistic models of users preferences, probabilistic reasoning

Kapitola 1

Úvod

V dnešnej dobe má každý užívateľ internetu prístup k obrovskému množstvu informácií a ich počet rýchlo narastá. Pre užívateľa je stále náročnejšie vyhľadávať informácie na základe toho, čo potrebuje, teda na základe svojích preferencií. Veľké množstvo informácií je uložených v databázach obsahujúcich rôzné objekty s rôznymi vlastnosťami. Pri výbere najvhodnejšieho objektu sa užívateľ rozhoduje práve podľa týchto vlastností.

Úlohou oblasti modelovania užívateľských preferencií je navrhnúť nástroj na reprezentáciu užívateľských preferencií, ktorý bude schopný postupného učenia sa. Ďalej bude tento nástroj schopný na základe neúplných preferencií užívateľa o objektoch, predpovedať jeho preferencie v prípadoch neznámych objektov.

Táto práca je zameraná na modelovanie preferencií nad množinou podobných objektov ako sú napríklad nootebooky, byty alebo zájazdy. Jedným z ďalších cieľov práce je zostaviť prehľad súčasných modelov užívateľských preferencí, ich rozdielne interpretácie a stručný prehľad teorií, na ktorých sú založené. Hlavnou úlohou práce je navrhnúť vlastný model reprezentujúci užívateľské preferencie. Tento model následne porovnať s už existujúcim modelom (Support Vector Regression) a rozhodnúť o jeho efektívnosti.

Kapitola 2 sa zaoberá problematikou užívateľských preferencií a rozdielnymi pohľadmi na preferencie samotné. Taktiež definuje jednotlivé možnosti reprezentácie

užívateľských preferencií a bližšie popisuje ich funkčnosť. V závere sú predstavené možnosti pri práci s viacerými užívateľmi.

V kapitole 3 je zostavený stručný prehľad jednotlivých modelov reprezenujúcich užívateľské preferencie. Kapitola obsahuje tiež prehľad už dokázaných ekvivalencií a prevodov medzi jednotlivými modelmi.

V kapitole 4 je predstavený samotný experiment práce. Postupne sú prezentované jednotlivé problémy exeperimentu a spôsoby ich riešenia. Kapitola popisuje navrhnutý model rozhodovania a podrobne sa rozoberá jeho funkčnosť.

Kapitola 5 obsahuje výsledky experimentu, v ktorých je navrhnutý model rozhodovania porovnaný s modelom, ktorý pracuje na základe teórie "Support Vector Regresion". V závere tejto kapitoly sú postrehy a návrhy na možné rozšírenia navrhnutého modelu, taktiež návrh ako vytvoriť obecnejšie a silnejšie modely. Príloha obsahuje užívateľskú dokumentáciu k programu.

Kapitola 2

Užívateľské preferencie

Užívateľskou preferenciou sa rozumie čokoľvek, čo konkrétny užívateľ uprednostňuje, prípadne neuprednostňuje. Preferencie sú zamerané na skupiny podobných objektov s podobnými vlasnosťami (notebooky, byty). Na základe týchto informácií (preferencií) je následne možné definovať usporiadanie množiny podobných objektov a to podľa preferencií užívateľa od najpreferovanejších po najmenej preferované. Vo všeobecnosti sú však zaujímavé len najlepšie objekty. Teda na základe preferencií je možné pre množinu O objektov $o_1, ..., o_n$ definovať usporiadanie týchto objektov.

Príklad: V prípade kúpy notebooku môže, ale nemusí byť zaujímavá informácia, že užívateľ preferuje byt 2+KK. Táto práca je však zameraná len na preferencie nad objektami s rovnakými atribútmi. Ak užívateľ preferuje notebooky značky Hewlett–Packard, tak najviac preferované budú práve tieto notebooky aj pri iných, menej vyhovujúcich vlastnostiach.

2.1 Vplyvy na preferencie

Samotné preferencie, predstavujúce požiadavky užívateľa bez akýchkoľvek vedľajších vplyvov, sú často menené na základe vedľajších faktorov. Pri následnom definovaní preferencií užívateľom sú tieto vplyvy už zohľadnené.

Vnútorné podmienky sú obmedzenia závislé na osobných pohnútkach užívateľa. Typickým príkladom je finančné obmedzenie. Užívateľ preferuje určitý druh vysoko výkonného notebooku, ktorý si finančne nemôže dovoliť. Na základe toho, je nútený zmeniť svoje preferencie. Ďalším príkladom môže byť predchádzajúca skúsenosť s určitou značkou notebooku a následne jej preferovanie respektíve odmietnutie.

Vonkajšie podmienky predstavujú vplyv na preferencie, ktorý nezávisí na situácii užívateľa, ale na podmienkach poskytovaných okolím. Užívateľ môže preferovať len notebook s takými vlasnosťami aké poskytuje trh.

Časový faktor je zmena preferencií s odstupom času. Napríklad, ak užívateľ preferoval viac palcový notebook, ale v súčasnej dobe veľa cestuje, zmení svoju preferenciu na malý a ľahký notebook.

2.2 Možnosti získavania preferencií

K dosiahnutiu čo najpresnejšieho výberu najlepších objektov je potrebné získavať stále viac informácii o užívateľovi a tým spresňovať jeho preferencie. Získavanie preferencií nemusí byť nutne založené na ich samotnom definovaní užívateľom. Spôsoby získavania preferencií je možné rozdeliť na:

Priamy prístup je najpresnejším získavaním preferencií. V tomto prístupe užívateľ špecifikuje svoje preferencie a to pomocou minimálnych a maximálnych hodnôt, prípadne definovaním usporiadania alebo preferovania, ak sa jedná o nespojité atribúty. Získanie preferencií priamym prístupom je pre užívateľa časovo najnáročnejšie, ale na druhej strane najpresnejšie. Napríklad, pri kúpe notebooku užívateľ definuje maximálnu cenu, minimálne požiadavky na veľkosť HDD, RAM, CPU a iné. V prípade značky môže uprednostňovať $Dell$ pred IBM atď.

Explicitný prístup znamená, že užívateľ nešpecifikuje svoje preferencie, ale hodnotí určité konkrétne objekty (notebooky) ako celky. Explicitne získané preferencie

neposkytujú takú presnosť ako priamo získané preferencie, ale pre užívateľa sú časovo prístupnejšie. V prípade notebookov užívateľ ohodnotí určitý notebook hodnotou od jedna po desať (jedna - najhoršie, desať - najlepšie).

Implicitný prístup predstavuje zber preferencií, ktoré užívateľ poskytuje podvedome, respektíve vyplyvajú z jeho činností. Tento druh preferencií je najmenej presný, ale zároveň pre užívateľa najmenej zaťažujúci. Napríklad, monitorovanie aktivít užívateľa ako sú:

- označovanie textu na stránkach,
- čas strávený na jednotlivých detailoch produktu,
- kúpa určitého notebooku cez internetový obchod

väčšinou iniciujú k zaujímavejším (preferovanejším) notebookom.

2.3 Objekty a preferencie

Bude zavedená notácia pre rozdielnych užívateľov $u_1, ..., u_k \in U$ a pre množinu objektov $o_1, ..., o_n \in O$. Každý objekt má definovaných m rozdielnych hodnôt atribútov patriacich do množín $A^1, ..., A^m$. Jednotlivé atribúty objektu $o_i \in O$ budú značené ako $a_i^1, ..., a_i^m$. Podľa [20] je rozlíšených niekoľo typov domén atribútov:

- *Nominálny* neobsahuje žiadne usporiadanie, väčinou sa jedná o textový atribút. Napríklad, značka notebooku reprezentovaná množinou $\{Dell, HP, IBM\}$.
- *Ordinálny* definuje usporiadanie nie však samotný rozdiel medzi dvoma hodnotami. Napríklad, {Zlý, Priemerný, Dobrý}.
- *Intervalový* – medzi dvoma hodnotami je možné určiť rozdiel, ale nie pomer. Typickým príkladom je Celsiova stupnica teploty, kde nie je možné povedať, že 20°C je dvakrát teplejších ako 10°C.

- *Pomerový* – obsahuje takzvanú absolútnu (pravú) nulu, tým sa dá určiť rozdiel a aj pomer medzi dvoma hodnotami. Napríklad, Kelvinova stupnica teploty, cena, váha a výdrž batérie notebooku.

Intervalová a pomerová doména sa spoločne nazývajú *numerická doména*. Atribúty s nominálnou doménou patria medzi *diskrétne atribúty*, atribúty s numerickou doménou zase medzi *spojité attribúty*. Ordinálna doména môže byť zaradená medzi diskrétne atribúty, tým sa však stráca informácia o usporiadaní. Na duhej strane jej zaradenie medzi spojité atribúty vyžaduje definovať samostatný nástroj na porovnávanie, alebo jeho prekonvertovanie na číselné hodnoty. V tejto práci je ordinálna doména zaradená medzi diskrétne atribúty.

Každý spojitý atribút je možné rozdeliť na intervaly, z ktorých každý bude reprezentovať jednu hodnotu novovzniknutého diskrétneho atribútu. Podľa vzniknutých intervalov sa rozlišujú dva základné typy rozdelenia:

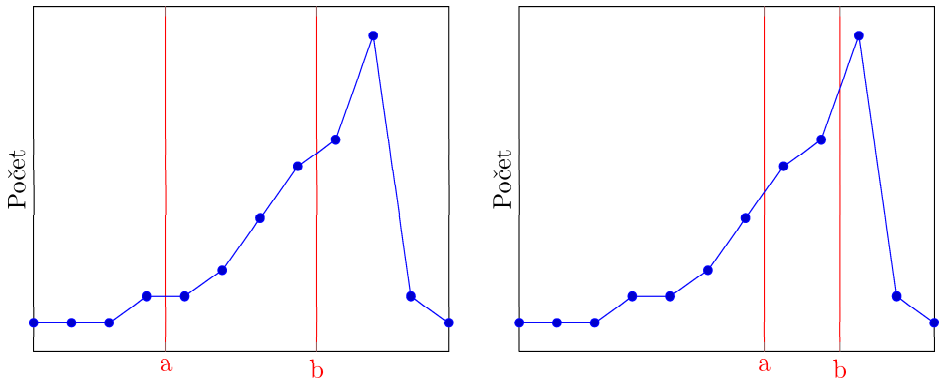
- *Ekvidistantné rozdelenie*, v ktorom je celá doména hodnôt atribútu rozdelená na rovnako veľké intervaly (viď obrázok 2.1a).
- *Ekvifrekvenčné rozdelenie*, v ktorom každý z intervalov obsahuje (približne) rovnaký počet objektov (viď obrázok 2.1b).

Ak sa napríklad, za doménu hodnôt objektov zoberie interval celých čísel od 1 do 12, z ktorých má každý objekt určitý počet výskytov

$$\{1_1, 2_1, 3_1, 4_2, 5_2, 6_3, 7_5, 8_7, 9_8, 10_{12}, 11_2, 12_1\}$$

kde dolný index definuje počet výskytu danej hodnoty objektu. Celkový počet všetkých objektov je 45. Potom ekvidistančné rozdelenie na tri diskrétne atribúty obsahuje intervaly $[1-4,66]$, $[4,66-8,33]$ a $[8,33-12]$, pričom každý interval obsahuje práve 4 rozdielne hodnoty objektov. Počet objektov v prvom intervaje je $1+1+1+2=5$, v druhom $2+3+5+7=17$ a v treťom $8+12+2+1=23$. Na druje strane ekvifrekvenčné rozdelenie na tri diskrétne atribúty obsahuje intervaly $[1-7,5]$, $[7,5-9,5]$ a $[9,5-12]$. Všetky intervaly obsahujú rovnaký počet (15)

objektov (1+1+1+2+2+3+5, 7+8, 12+2+1), ale počet hodnôt objektov v jednotlivých intervaloch je rozdielny (7, 2 a 3). Celý príklad je zobrazený na grafoch obrázku 2.1.



(a) Ekvidistantné rozdelenie (b) Ekvifrekvenčné rozdelenie

Obrázek 2.1: Rozdelenie spojitého atribútu.

Každý objekt $o \in O$ má definovaných m hodnôt atribútov $A^i, i = 1, ..., m$. Jednotlivé atribúty je možné rozdeliť na spojité a diskrétne. Bude zavedená notácia preferencie $Pref(o_i)$ objektu $o_i \in O$ a preferencie $Pref(a^k)$ atribútu $a^k \in A^k$, kde u predstavuje konkrétneho užívateľa. Podľa toho je možné samotné preferencie rozdeliť na:

- *Lokálne preferencie* vyjadrujúce ako užívateľ preferuje jednotlivé atribúty.
- *Globálne preferencie* vyjadrujúce ako užívateľ preferuje jednotlivé objekty.

Lokálna preferencia poukazuje na to, ako sa preferuje objekt len podľa k-tého atribútu $A^k, k = 1, ..., m$. V prípade definovania lokálnych preferencií objektu o_i nad atribútmi $Pref^u(a_i^1), \ldots, Pref^u(a_i^m)$ je potrebné definovať spôsob, ako z týchto lokálnych preferencií vytvoriť preferenciu celého objektu $Pref(o_i)$.

2.4 Modely užívateľských preferencií

Pod pojmom *model užívateľských preferencií* sa rozumie spôsob ako definovať užívateľské preferencie. Model užívateľských preferencií musí obsahovať

- *reláciu usporiadania* nad množinou objektov O, pomocou ktorej je možné pre každé dva objekty $o_i, o_j \in O$ určiť ich vzájomný vzťah na základe preferencií $Pref(o_i)$ a $Pref(o_j)$. Teda objekt o_i je lepší ako o_j ($Pref(o_i) > Pref(o_j)$), horší (Pref(o_i) $< Pref(o_j)$), rovnaký ($Pref(o_i) = Pref(o_j)$), prípadne ich vzájomný vzťah nie je definovaný.

2.4.1 Hodnotový model

Pri tomto modele je každému objektu alebo atribútu priradená hodnota z množiny ohodnotení R, nad ktorou je definovaná relácia usporiadania. Obecne teda existuje zobrazenie $f : V \rightarrow R$, kde V reprezentuje množinu objektov alebo atribútov. Hodnotové modely je možne rozdeliť podľa veľkosti množiny R:

- *Dvojhodnotový model* (*Boolean model*), v ktorom množina R obsahuje hodnoty 1 a 0, prípadne Ano/Nie, Páči/Nepáči, ktoré sú pre užívateľa prijateľnejšie.

- *Viac hodnotový model*, v ktorom R nadobúda väčšiu konečnú množinu hodnôt. Napríklad hodnoty od 1 do 10, kde 1 je najhoršie a 10 najlepšie, prípadne v praxi často používaný počet hviezdičiek.

- *Fuzzy model*, v ktorom je množina ohodnotení R rovná intervalu $[0, 1]$, kde 0 je najhoršie ohodnotenie a 1 nalepšie. Každému objektu alebo atribútu je priradené reálne číslo z intervalu R.

Agregačná funkcia predstavuje jednu z možností, pomocou ktorej sa dá pri hodnotovom modele vypočítať globálna preferencia z lokálnych preferencií. Medzi najzákladnejšie a načastejšie používané funkcie patria:

- *Aritmetický priemer* je najobecnejším typom agregačnej funkcie a pre navrhnutý model bol vybraný ako najvhodnejšie riešenie.

$$Pref(o_i) = \frac{Pref(a_i^1) + \ldots + Pref(a_i^m)}{m}$$

- *Vážený priemer* pridáva k aritmetickému priemeru váhu jednotlivých atribútov.

$$Pref(o_i) = \frac{w^1 Pref(a_i^1) + \ldots + w^m Pref(a_i^m)}{w^1 + \ldots + w^m}$$

kde $w^1, \ldots, w^m$ reprezentuje váhu atribútov 1 až m.

- *Vzdialenosť od bodu* 0 je ďalší typ agregačnej funkcie, ktorá môže byť reprezentovaná s váhou alebo bez váhy atribútov.

$$Pref(o_i) = \sqrt{\frac{Pref(a_i^1)^2 + \ldots + Pref(a_i^m)^2}{m}}$$

$$Pref(o_i) = \sqrt{\frac{w^1 Pref(a_i^1)^2 + \ldots + w^m Pref(a_i^m)^2}{w^1 + \ldots + w^m}}$$

- *Minimum, Maximum* ako posledné z vybraných agregačných funkcií

$$Pref(o_i) = \max(Pref(a_i^1) + \ldots + Pref(a_i^m))$$

$$Pref(o_i) = \min(Pref(a_i^1) + \ldots + Pref(a_i^m))$$

2.4.2 Relačný model

Definuje konkrétne usporiadanie, teda samotné relácie na množine objektov O. Podľa [27] je definovaná množina relácií $P(v_i, v_j)$, ak v_i je preferovanejšie ako v_j. Množina objektov alebo atribútov je definovaná ako V a $v_i, v_j \in V$. Ďalej množina relácií $I(v_i, v_j)$ ak v_i je rovnako preferované ako v_j a množina $J(v_i, v_j)$ ak v_i nieje možné porovnať s v_j. Kvôli spresneniu výsledkov je dodaná množina $Q(v_i, v_j)$ ak v_i je o trochu viac preferované ako v_j. Na príklade značky notebooku je Dell o trochu viac preferovaný ako HP. Pri určovaní globálnych preferencií z lokálnych sa najčastejšie používa tranzitívna vlastnosť. Problémom relačného modelu je určiť rozdiel medzi preferovanými a nepreferovanými objektmi (atribútmi).

K jednotlivým reláciam medzi dvoma atribútmi alebo objektami je ďalej možné pridať ohodnotenie pomocou fuzzy funkcie alebo viac–hodnotového modelu a tým vytvoriť relačno hodnotový model.

2.5 Induktívne metódy

Pri práci obecne sú k dispozícii samotné dáta, model a výsledky. Jednotlivé metódy je možné rozdeliť podľa toho, na zisťovanie čoho sú určené:

- *Deduktívne metódy* - zisťujú výsledky z modelu a dát
- *Abduktívne metódy* - zisťujú dáta z modelu a výsledkov
- *Induktívne metódy* - zisťujú model z dát a výsledkov

Pri pohľade na užívateľské preferencie, reprezentujú výsledky preferencie a model predstavuje model rozhodovania. Samotnú teóriu reprezentácie užívateľských preferencií je teda nutné rozšíriť o spôsob, ako z neúplných preferencií a dát modelovať užívateľove rozhodovanie a podľa neho následne riešiť predikciu užívateľských preferencií. K tomu slúžia práve induktívne metódy, ktoré z úplných dát a z informácií o (neúplných) preferenciách vytvárajú model rozhodovania.

2.5.1 Logické programovanie

Logické programovanie je teória, ktorá podrobne skúma logické odvodzovanie (napríklad v Prologu, ale obecnejšie). V logickom programovaní sú používané čiarky (,) namiesto symbolu (&) pre konjunkciu a obrátená šípka (←) namiesto implikácie.

Logický program: Logický program je ľubovoľná konečná množina Hornových klauzúl. (Prologovsky) program P je zoznam programových klauzúl, faktov alebo pravidiel kde:

- *Fakt:* deklaruje vždy pravdivé veci
- *Pravidlo:* deklaruje veci, ktorých pravdivosť závisí na daných podmienkach
- *Dotaz (cieľ)*: užívateľ sa pýta programu, či sú veci pravdivé

Odpoveď na dotaz môže byť:

- *pozitívna* – dotaz je splniteľný a uspel

- *negatívna* – dotaz je nesplniteľný a neuspel

Reprezentácia modelu rozhodovania jednoduchým logickým programom nie je úplne dostatočná.

Induktívne logické programovanie [22] predstavuje rozšírenie logického programovania, ktorého myšlienkou je vytvárať novú množinu pravidiel H (hypotéz). IPL obsahuje množiny:

- B obsahujúcu známe pravidla a fakty
- E^- negatívne príklady
- E^+ pozitívne príklady

Na základe množín B a E^+ vytvorí novú množinu hypotéz H, ktorá súčasne nesmie kolidovať s množinou E^-.

Fuzzy logické programovanie je založené na teórii *fuzzy* (viac–hodnotovej) *logiky* [46][47], ktorá narozdiel od dvoj–hodnotovej logiky nepracuje len s množinou pravdivostných hodnôt $\{true, false\}$, ale z fuzzy množinou definovanou intervalom $[0, 1]$, reprezentujúcou stupeň pravdivosti. Vo fuzzy logike sú definované *t–norm* a *t–conorm*, ktoré istou formou nahrádzajú operátor konjukcie (t–norm) a (t–conorm) disjunkcie z dvoj–hodnotovej logiky. Funkcia $T : [0, 1]^2 \rightarrow [0, 1]$ je *t-norm* (*trianguláma norma*), ak platia nasledujúce štyri podmienky:

- Ekvivalentná podmienka $T(1, x) = x$ pre $\forall x \in [0, 1]$
- T je komutatívne $T(x, y) = T(y, x)$ pre $\forall x, y \in [0, 1]$
- T je neklesajúce v oboch prvkoch $T(x, y) \leq T(u, v)$ pre všetky $0 \leq x \leq u \leq 1$ a $0 \leq y \leq v \leq 1$
- T je asociatívne $T(x, T(y, z)) = T(T(x, y), z)$ pre $\forall x, y, z \in [0, 1]$

Obdobne potom funkcia $S : [0, 1]^2 \rightarrow [0, 1]$ je *t-conorm* (tiež *s-norm*), ak platia nasledujúce štyri podmienky:

- Ekvivalentná podmienka $S(0, x) = x$ pre $\forall x \in [0, 1]$
- S je komutatívne $S(x, y) = S(y, x)$ pre $\forall x, y \in [0, 1]$
- S je neklesajúce v oboch argumentoch $S(x, y) \leq S(u, v)$ pre všetky $0 \leq x \leq u \leq 1$ a $0 \leq y \leq v \leq 1$
- S je asociatívne $S(x, S(y, z)) = S(S(x, y), z)$ pre $\forall x, y, z \in [0, 1]$

Tabuľka 2.1 ukazuje príklady t–norm a t–conorm, ktoré spĺňajú ich vlastnosti.

Názov	t-norm	t-conorm
Zadeh	$\min(x, y)$	$\max(x, y)$
pravdepodobnostný	$x * y$	$x + y - xy$
Lukasiewicz	$\max(x + y - 1, 0)$	$\min(x + y, 1)$
drastický	x ak $y = 1$	x ak $y = 0$
	y ak $x = 1$	y ak $x = 0$
	inak 0	inak 1

Tabulka 2.1: Príklady t norm a t conorm.

Vo FLP [43][34] je jednotlivým faktom a pravidlám logického programu priradená fuzzy hodnota z intervalu $[0, 1]$ reprezentujúca stupeň pravdivosti logických formúl, ktorý sa dopočítava práve na základe t–norm a t–conorm. Notácia fuzzy logiky bola prevzatá z [27].

2.5.2 Pravdepodobnostné modely

Nech $(\Omega, \mathcal{A})$ je merateľný priestor. Prvky množiny Ω sú nazývané *elementárne javy* a sú značené ako ω, prvky σ–algebry A budú nazývané *javy* a značené veľkým písmenom zo začiatku abecedy. *Pravdepodobnosť* P je definovaná ako miera na $\mathcal{A}$ s vlastnosťou $P(\Omega) = 1$, to znamená P je množinová funkcia na $\mathcal{A}$ s vlastnosťami:

- $P(A) \geq 0$, $A \in \mathcal{A}$
- $P(\Omega) = 1$, $P(\emptyset) = 0$

- $P(\bigcup_{n=1}^{\infty} A_n) = \sum_{n=1}^{\infty} P(A_n)$, ak je $\{A_n\}$ postupnosť po dvoch disjunktných javoch

Nech $A, B \in \mathcal{A}$ a $P(B) > 0$ potom *podmienná pravdepodobnosť* javu A za podmienky B je definovaná vzťahom

$$P(A|B) = \frac{P(A \cap B)}{P(B)}$$

Bayesova veta teórie pravdepodobnosti udává, ako podmienená pravdepodobnosť nejakého javu súvisí s opačne podmienenou pravdepodobnosťou. Pre dva náhodné javy A a B s pravdepodobnosťami $P(A)$ a $P(B)$ a predpokladu, že $P(B) > 0$ platí

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

kde $P(A|B)$ je podmienená pravdepodobnosť javu A za predpokladu, že nastal jav B, a naopak $P(B|A)$ je pravdepodobnosť javu B podmienená výskytom javu A. Javy A, B sa nazývaju *nezávislé javy* ak $P(A \cap B) = P(A)P(B)$, v opačnom prípade sú to *závislé javy.*

Bayesové siete Nech $V = \{v_1, \ldots, v_n\}$, $n > 1$ je množina premenných. *Bayesová sieť* B nad množinou premenných V je dvojica (B_G, B_P), kde B_G je neorientovaný acyklický graf (DAG)[1] nad množinou premenných V a B_P je množina pravdepodobnostných tabuliek $B_P = \{P(v|Parent(v)|v \in V)\}$, kde $Parent(v)$ je množina rodičov vrcholu v v štruktúre B_G. Bayesová sieť reprezentuje pravdepodobnostné rozdelenie $P(V) = \prod_{v \in V} P(v|Parent(v)$.

Markovové siete (tiež Markovové nahodné polia)[2] je teória pre združené rozdelenie množiny premenných $X = (X_1, X_2, ..., X) \in \mathbf{X}$. Je zložená z neorientovaného grafu G a množiny potencionálnych funkcií φ_k. Graf obsahujc vrcholy prc každú premennú a celý model grafu má potencionálnu funkciu prc každú kliku v grafc. Potencionálna funkcia je nezáporná reálna funkcia stavov odpovedajúcej kliky. Združené

[1] Directed Acyclic Graph

[2] Markov Random Fields

pravdepodobnostné rozdelenie reprezentované Markovovou sieťou je dané

$$P(X = x) = \frac{1}{Z} \prod_k \varphi_k(x_{\{k\}}) \tag{2.1}$$

kde $x_{\{k\}}$ je stav k–tej kliky (stav premenných obsiahnutých v klike) a Z (tiež rozdeľovacia funkcia alebo partition funcion) je daná ako $Z = \sum_{x \in X} \prod_k \varphi_k(x_{\{k\}})$. Markovove siete sú často bežne reprezentované ako log-lineárne modely, s každou potenciálnou funkciou kliky, nahradenou exponentom vážených súm vlastností stavov.

$$P(X = x) = \frac{1}{Z}(\sum_j w_j f_j(x)) \tag{2.2}$$

Vlastnosťou môže byť každá reálna funkcia stavov. Pre jednoduchosť je možné predstaviť si túto vlastnosť ako binárnu $f \in \{0, 1\}$. Vo väčšine priamych prekladov z potenciálnej funkcie jedna vlastnosť korešponduje pre každý možný stav $x_{\{k\}}$ každej kliky, s váhou $log(\varphi_k x_{\{k\}})$. Reprezentácia Markovovej siete je exponenciálna v závislosti na veľkosti kliky. V Markovovej sieti je Markov blanket nejakého vrcholu A množina všetkých jeho susedov.

2.5.3 Rozhodovacie stromy

Rozhodovacie stromy patria medzi veľmi dobre prepracovanú teóriu. Sú reprezentované orientovaným stromom s ohodnotením v listoch a s rozhodovacími pravidlami v uzloch. Indukcia nad rozhodovacím stromom je ukázaná v [33]. Rozhodovacie stromy rozšírené na fuzzy rozhodovacie stromy je možné nájsť v [2].

2.5.4 Support Vector Regresion

Bude predpokladaná existencia trénovacích dát $\{(x_1, y_1), ..., (x_n, y_n)\} \subset X \mathrm{x} R$, kde X je priestor vstupných vzorov. Ďalej bude predpokladané, že je zadané $\epsilon > 0$ (chyba, ktorá je v tolerancii). Cieľom SV-algoritmu je nájsť funkciu $f(x)$, ktorá má nanajvýš ϵ-ovú odchylku od skutočných hodnôt y_i pre všetky trénovacie dáta a zároveň je čo najviac "plochá".

Najjednoduchším prípadom je považovať funkciu f za lineárnu funkciu. To znamená, že môže byť zapísaná ako

$$f(x) =< w, x > +b \qquad x \in X,\, b \in R,$$

kde $< ., . >$ označuje skalárny súčin. Plochosť v tomto prípade znamená nájsť w tak, aby norma $||w||^2 =< w, w >$ bola čo najmenšia. Tento problém sa dá prepísať ako konvexný optimalizačný problém v tvare:

$$\text{minimalizuj} \qquad \frac{1}{2}||w||^2$$

$$\text{za podmienky} \begin{cases} y_i - < w, x_i > -b \leq \epsilon \\ < w, x_i > +b - y_i \leq \epsilon \end{cases}$$

Problém však je, že nie vždy takáto funkcia existuje. Preto sa podmienky na chyby zoslabujú a zavádzajú sa voľné premenné ξ_i a ξ_i^*. Úloha potom je:

$$\text{minimalizuj} \qquad \frac{1}{2}||w||^2 + C\sum_{i=1}^{n}(\xi_i + \xi_i^*)$$

$$\text{za podmienky} \begin{cases} y_i - < w, x_i > -b \leq \epsilon + \xi_i \\ < w, x_i > +b - y_i \leq \epsilon + \xi_i^* \\ \xi_i, \xi_i^* \geq 0, \end{cases}$$

kde $C \geq 0$. Tento minimalizačný problém sa rieši cez duálnu úlohu s pomocou Lagrangeových multiplikátorov. Táto duálna úloha sa dá zapísať v tvare:

$$\text{maximalizuj} \quad -\frac{1}{2}\sum_{i,j=1}^{n}(\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*) < x_i, x_j > -\epsilon\sum_{i=1}^{n}(\alpha_i + \alpha_i^*) + \sum_{i=1}^{n} y_i(\alpha_i - \alpha_i^*)$$

$$\text{za podmienky} \quad \sum_{i=1}^{n}(\alpha_i - \alpha_i^*) = 0 \;\; a \;\; \alpha_i, \alpha_i^* \in [0, C],$$

kde α_i, α_i^* sú Lagrangeové multiplikátory. Navyše platí: $w = \sum_{i=1}^{n}(\alpha_i - \alpha_i^*)x_i$ a $f(x) = \sum_{i=1}^{n}(\alpha_i - \alpha_i^*) < x_i, x > +b$. Z rovnice pre f je zrejmé, že pre jej výpočet nie je potrebné počítať w. Parameter b môže byť spočítaný, napríklad, s využitím Karush-Kuhn-Tuckerových podmienok.

Zatiaľ sa uvažovalo iba nad lineárnym SV-algoritmom. Rozšírenie na nelineárny prípad sa dá urobiť, napríklad, predspracovaním trénovacích vzorov pomocou zobrazenia $\Phi : X -> F$, kde F je priestor nových vzorov a na tieto predspracované vzory

použiť lineárny SV–algoritmus. SV–algoritmus závisí iba na skalárnom súčine medzi vzormi x_i. Preto stačí vedieť iba $k(x, x') =< \Phi(x), \Phi(x') >$ namiesto explicitného vyjadrenia Φ, kde k je vhodné jadro. To umožnuje prepísať SV optimalizačný problém následujúco:

$$\text{maximalizuj} \quad -\frac{1}{2}\sum_{i,j=1}^{n}(\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*)k(x_i, x_j) - \epsilon\sum_{i=1}^{n}(\alpha_i + \alpha_i^*) + \sum_{i=1}^{n} y_i(\alpha_i - \alpha_i^*)$$

$$\text{za podmienky} \quad \sum_{i=1}^{n}(\alpha_i - \alpha_i^*) = 0 \ a \ \alpha_i, \alpha_i^* \in [0, C],$$

Funkcia f a w sa dajú zapísať ako: $w = \sum_{i=1}^{n}(\alpha_i - \alpha_i^*)\Phi(x_i)$ a $f(x) = \sum_{i=1}^{n}(\alpha_i - \alpha_i^*)k(x_i, x) + b$. Viac o teórii Support Vector Regression je možné nájsť v [39] a [38]

2.6 Preferencie viacerých užívateľov

Samotné preferencie sú odlišné pre rozdielnych užívateľov, preto je možné jednotlivé modely rozhodovania rozdeliť podľa toho, koľko rozdielnych užívateľov sú schopné spracovať.

- **Jeden užívateľ:** tieto modely pracujú s jedným konkrétnym užívateľom. Teda pre každého užívateľa vytvoria samostatnú inštanciu modelu rozhodovania, ktorá nevyužíva informácie o preferenciách iných užívateľov, respektíve nevyužíva žiadne už známe informácie.

- **Viac užívateľov:** tieto modely rozlišujú rozdielnych užívateľov a pri rozhodovaní o preferenciách využívajú informácie o preferenciách iných (podobných alebo absolútne rozdielnych) užívateľov.

Rozhodovacie modely môžu k spresneniu výsledkov využívať aj informácie (ak sú poskytnuté) o samotnom užívateľovi (pohlavie, vek, zamestnanie, záľuby, finančná situácia a iné), podľa ktorých je možné predpokladať, alebo postupom času vypozorovať podobnosti medzi jednotlivými vlasnosťami užívateľov a tým vytvoriť *užívateľský profil*. K spresňovaniu profilu budú prispievať všetci užívatelia patriaci do daného

profilu. Podobnosť medzi viacerými užívateľmi je sledovaná na preferenciách jednotlivých užívateľov.

Jednou z možností nachádzania podobnosti medzi preferenciami dvoch užívateľov, respektíve podobnosti preferencií užívateľa s profilom užívateľov, predstavuje kolaboratívne filtrovanie. Základným algoritmom kolaboratívneho filtrovania je K–NN algoritmus, ktorý hľadá K najbližších susedov. Teda pre užívateľa v nachádza užívateľov $u_1, \ldots, u_K$ s najmenšou vzdialenosťou. Vzdialenosť medzi dvoma užívateľmi u a v je možné definovať ako:

$$s(u,v) = \sqrt{\sum_{i=1}^{n} (Pref^u(o_i) - Pref^v(o_i))^2}$$

kde n je počet objektov. Preferencie objektu o užívateľom v je možné vypočítať z K najbližších susedov, napr. ako priemer ich preferencií $Pref^v(o) = \sum_{i=1}^{K} Pref^{u_i}(o)$.

2.7 Zameranie práce

Táto práca je zameraná na explicitne získavané preferencie o užívateľovi, pri ktorých sú použité globálne preferencie. Ako model preferencií bol zvolený hodnotový model s hodnotami od 1 do 10, kde 1 je najhoršie ohodnotenie a 10 najlepšie. Navrhnutý model rozhodovania je založený na teórii pravdepodobnosti. Tento model je porovnaný s modelom vychádzajúcim z teórie Support Vector Regression viď. 2.5.4.

Kapitola 3

Existujúce modely rozhodovania

3.1 Rozdelenie modelov

Vo všeobecnosti existujú dva prístupy (pohľady) na logické programy *modelový* (*grafický*) a *dôkazový* (*výpočtový*).

Modelový pohľad obsahuje takzvané *Herbrandovo univerzum* (respektíve *báza*), zložené zo všetkých skutočností, ktoré je možné konštruovať z predikátových, konštantných a funkčných symbolov programu. Herbrandovo univerzum (v istom zmysle) určuje množinu všetkých možných situacií vo svete, popisovaných programom. Priradenie pravdivostnej hodnoty k prvkom Hebrandovej báze sa niekedy tiež nazýva *Herbrandova interpretácia*. Logický program, založený na modelovom prístupe, obmedzuje množinu všetkých situacií len na tie, ktoré sú možné. Najdôležitejšou vecou v modelovom prístupe je teda to, že špecifikuje, ktorá časť vo svete je možná vzhľadom k použitej logickej teórii.

Výpočtový pohľad predstavuje iný spôsob pohľadu na logické programy a pochádza z dôkazovej teórie. Z tejto perspektívy môže byť logický program použitý k poskytnutiu informácie o tom, že niektoré fakty, pravidlá a dotazy sú logicky spojené s programom. Ako príklad môže byť zobraná funkcia kompilátora k určeniu syntaktickej správnosti jednotlivých častí programu, prípadne podobná funkcia bez-

kontextových gramatík. V týchto situáciách je dôležitá pravdivosť zadaného vstupu, takže odpoveďou je pravdivostná hodnota. Dôkaz je typicky konštruovaný pomocou SLD rezolúcie. Dôležité je tiež uvedomiť si súvislosť medzi SLD deriváciou a tradičnou deriváciou, používanou v bez-kontextových gramatikách. V oboch prípadoch však existuje mapovanie jedna k jednej medzi týmito dvoma pohľadmi na gramatiky.

3.2 Prehľad modelov

Existujú dva spôsoby, ako sa pozerať na logický program ako taký. Prvý pohľad, je modelový (grafický) a zavisí na tom, čo z reálneho sveta je možné interpretovať tak, aby to uspokojilo potreby programu. Patria sem modely založené na teórii Bayesových a Markovových sietí. Druhý pohľad je dôkazový. Ten určuje, čo je možné dokázať vzhľadom na daný cieľ. Samozrejme, oba z týchto pohľadov sú úzko spojené, pretože logický program má rezolučné vyvrátenie, len v prípade, že patrí do Herbrandovho modelu programu.

Jednotlivé modely je možné rozdeliť podľa teórií, z ktorých vznikli a z ktorých vychádzajú. Medzi modely neobsahujúce logické programovanie je možné zaradiť dvojhodnotový model. K ostatným druhom modelov je nutné povedať, že vznikli kombináciou (minimálne) dvoch teórií, z ktorých jednu poväčšine predstavuje logické programovanie, respektíve predikátovú logiku prvého rádu. K modelom založeným na grafickom pohľade patria modely vychádzajúce z kombinácie Bayesových sietí a logického programovania, obecne nazývané Knowledge-Based Model Construction ($KBMC$) [45]. K najrozšírenejším predstaviteľom tejto skupiny patrí Bayesovo Logické Programovanie (BLP) [16][17][18], ale tiež Probabilistic Logic Programs (PLP) [25][26].

Z trochu iného uhlu pohľadu vychádza Probabilistic Relational Model (PRM) [7][8][9][10], ktorý je kombináciou frame-based systémov a Bayesových sietí. Tento model bol rozšírený o Entity–Relationship model (ER–model), čím vznikol istý jazyk pre pravdepodobnostný ER–model [13]. Je nutné poznamenať, že ER–model je

vlastne špeciálnym prípadom logiky, teda pravdepodobnostný ER–model pripúšťa logické výrazy ako obmedzenie pre konštrukciu základnej siete, ale pravdivostná hodnota týchto výrazov musí byť známa vopred.

Ďalším príkladom sú modely vychádzajúce z kombinácie Markovových sietí a predikátovej logiky prvého rádu, kde patria Markové Logické Siete (MLN) [6] [5]. Obdobne z Markovových sietí vychádza aj model Relational Markov Networks (RMN) [41][42], ktorý používa databázové dotazy ako šablóny pre kliku a má funkciu pre každý stav kliky. MLN zovšeobecňujú RMN tým, že poskytujú silnejší jazyk pre výstavbu funkcií (logika prvého rádu namiesto konjunktívnych dotazov), a umožňujú neistotu vo vzťahoch medzi atribútmi (nie len medzi atribútmi jednotlivých objektov). RMN sú exponenciálne vo veľkosti kliky, zatiaľ čo MLN umožňuje užívateľovi zistiť počet prvkov, aby bola možná mierka do omnoho väčšej veľkosti kliky. Ďalej sem patrí Relational Dependency Network (RDN) [24], v ktorých každý uzol pravdepodobnostne podmienený Markovovým blanketom, je daný rozhodovacím stromom. Každá RDN korešponduje s MLN a naopak, danú pevnou distribúciou Gibbs sampler operating [12].

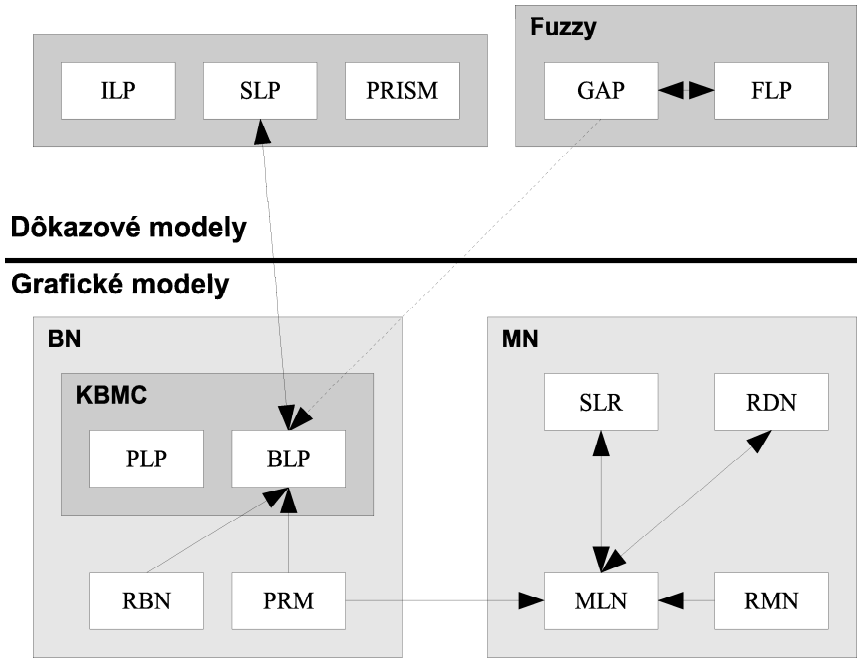
K modelom založeným na dôkazovej teórii patria modely vychádzajúce z kombinácie logického programovania a log–lineárneho modelu. Stochastické Logické Programy (SLP) [23] [4] predstavujú rozšírenie stochastických bez-kontextových gramatík. V zásade podobné modely sú Independent Choice Logic (ICL) [28][29] a taktiež PRogramming In Statistical Modeling ($PRISM$) [36][37]. Do tejto skupiny je ďalej možné zaradiť Probabilistic Constraint Logic Programming ($P-CLP$) [35].

Samostatnú skupinu predstavuje Fuzzy Logické Programovanie (FLP) [43][34], vychádzajúce z rozšírenia logického programovania o fuzzy (viac-hodnotovú) logiku. Podobne ako SLP a $PRISM$, patrí do skupiny modelov založených na dôkazovej teórii. Medzi modely vychádzajúce z fuzzy logiky patria aj zovšeobecnené anotované programy ako Generalized Annotated Programs (GAP) [14][19]. GAP pozostáva z Hornových klauzúl logického programovania zložených len z anotovaných atómov (atómov, ktoré majú naviac priradenú hodnotu z intervale $[0, 1]$).

Iný pohľad na preferencie je ukázaný v [40], ktorý vychádza zo štatistiky. Na základe užívateľom volených objektov, vytvára vektor vlastnosti, ktoré majú zvolené objekty podobné. Následne ponúka ďalšie objekty obsahujúce len vektor týchto vlastností. Ako príklad môže byť uvedené vyhľadávanie v dokumentoch, kde vlastnosti sú reprezentované slovami (alebo kombináciou slov) a objekty sú jednotlivé dokumenty.

3.3 Ekvivalencie a prevody

Medzi jednotlivými modelmi už existujú dokázané ekvivalencie (prevody), či už na základe vhodnej substitúcie alebo zmenou interpretácie. Ekvivalencia medzi SLP a BLP bola ukázaná v [30]. V článku [21] bola predstavená ekvivalencia medzi modelom FLP a GAP. Ďalej v práci [44] bol ukázaný spôsob transformácie z GAP do BLP. Hlavnou myšlienkou je, že každý anotovaný atóm (atóm, ktorý má naviac



Obrázek 3.1: Prehľad porovnaní modelov.

priradenú hodnotu z intervale $[0, 1]$) bude braný ako jeden predikát. Teda z pohľadu BLP, každý anotovaný atóm predstavuje jeden vrchol Bayesovej siete popisujúcej BLP. Problémom však ostáva, že interval $[0, 1]$ je nekonečný, takže z jedného atómu

z rozdielnou anotáciou môže vzniknúť nekonečne veľa rozdielnych vrcholov. Riešením je rozdelenie intervalu na niekoľko menších časti (napríklad interval $[0.25, 0.75]$) s tým, že všetky rovnaké atómy s anotáciou patriacou do toho isté intervalu by reprezentovali jeden vrchol. Aj keď existuje formálny prevod z GAP (a teda aj FLP) do BLP pri prevode sa môže stratiť veľká časť informácií, ktorú GAP obsahuje. Podrobný prehľad porovnaní modelov ukazuje je ukázaný na obrazku 3.1.

Kapitola 4

Experimentálne porovnanie modelov

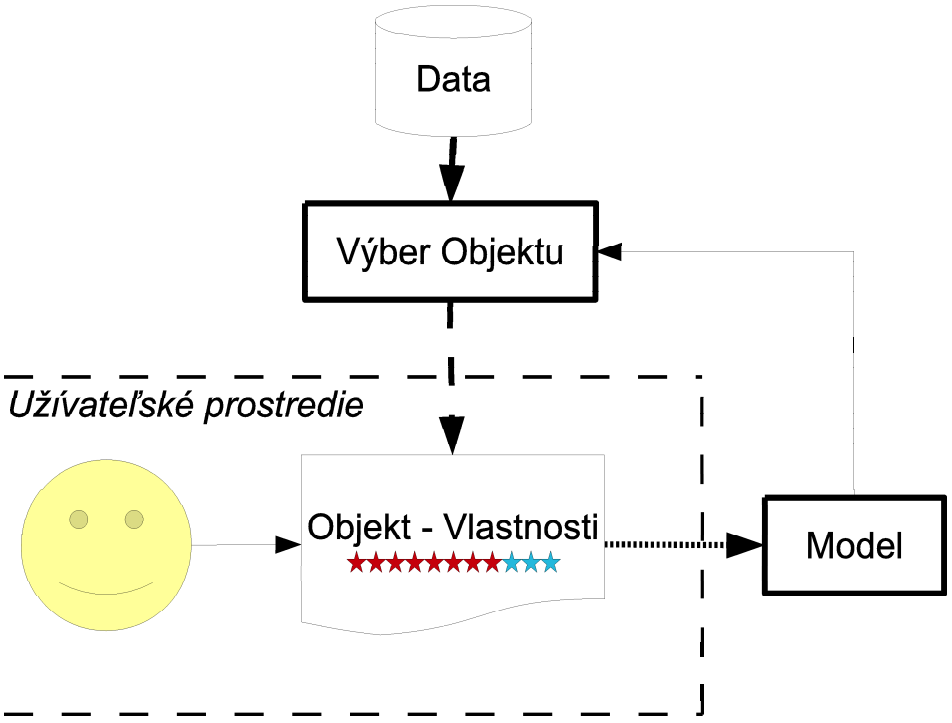
Hlavnou myšlienkou experimentu je určiť presnosť navrhnutého modelu rozhodovania so skrytým užívateľom a zároveň porovnať jeho presnosť s iným existujúcim modelom rozhodovania. Data o objektoch sú uložené v relačnej databáze *MySQL* ako jedna tabuľka, ktorá má m atribútov. Celý program je naprogramovaný v jazyku *Java*.

- **Skrytý užívateľ** reprezentuje konkrétneho užívateľa a obsahuje ohodnotenie všetkých objektov. K porovnaniu presnosti modelu rozhodovania boli použití štyria rozdielni užívatelia (viď. 4.3). Kazdý z nich je reprezentovaný jednou tabuľkou v databáze s atribútmi ID a $RATING$.
- **Použité dáta**, na ktorých boli prevádzané testy sú zobrané z [49]. Veľkosť testovacej sady je 800 rozdielnych a reálnych notebookov.

4.1 Popis experimentu

Experiment predpokladá na jednej strane existenciu skrytého užívateľa a na druhej strane program predkladajúci užívateľovi objekty k zadaniu preferencií, ktoré

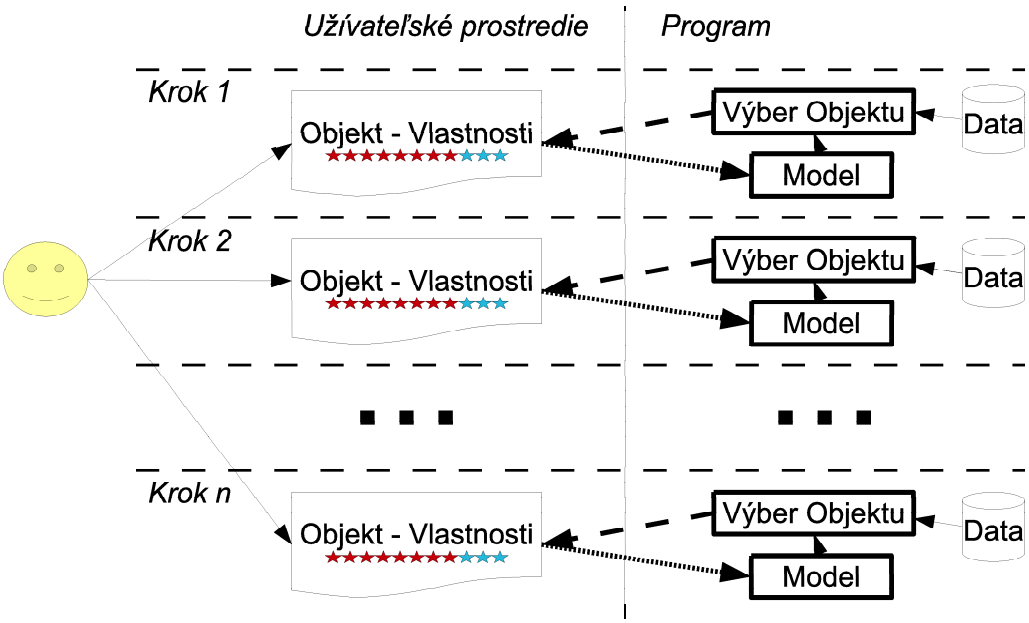
následne použije k vytvoreniu modelu rozhodovania. Skrytý užívateľ obsahuje preferencie o všetkých objektov, ale program nevyužíva informácie o týchto preferenciách. V experimente je skrytému užívateľovi predložený objekt s určitými vlasnosťami. Tento objekt má ohodnotiť na základe svojich preferencií hodnotou od 1 do 10, kde 1 je najhoršie a 10 najlepšie hodnotenie. Množina ohodnotení bude značená ako R a jednotlivé hodnoty $r_1, \ldots r_{10}$. Aj keď sa užívateľ rozhoduje na základe vlasností objektu, hodnotenie je len jedno pre celý objekt. Táto infomácia je predaná programu, ktorý podľa nej vytvorí model rozhodovania daného užívateľa (obrázok 4.1).



Obrázek 4.1: Jeden krok experimentu

Krok ohodnocovania objektov sa opakuje, čiže užívateľovi je ponúknutý iný objekt, ktorý má opäť, na základe svojich preferencií, ohodnotiť hodnotou z množiny R od 1 do 10. Ohodnotenie skrytým užívateľom je predané programu, ktorý postupne s každým ďalším ohodnoteným objektom upravuje a spresňuje model rozhodovania daného skytého užívateľa (obrázok 4.2).

Po každom ohodnotenom objekte sú modelom rozhodovania vygenerované ohodno-



Obrázek 4.2: Viac krokov experimentu

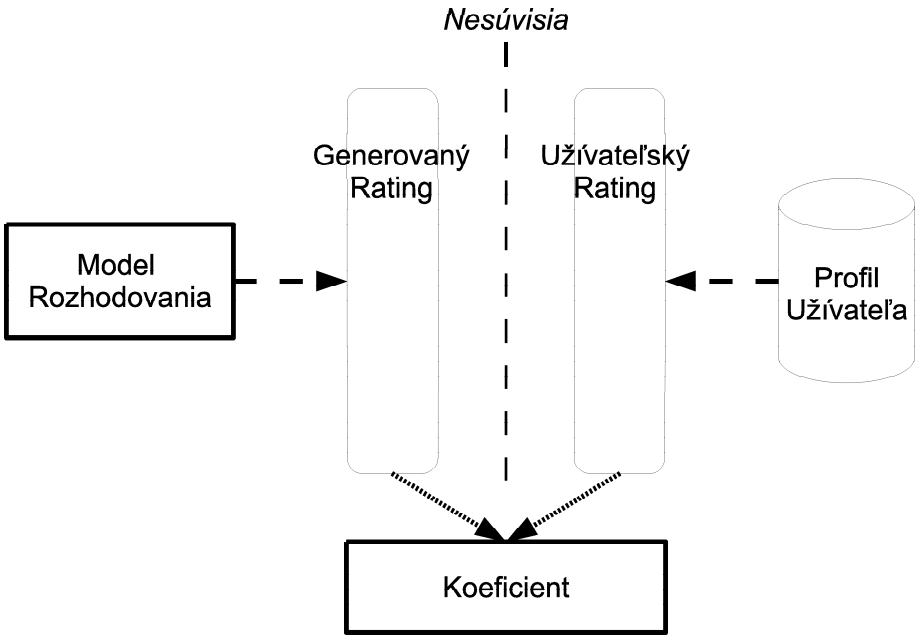
tenia všetkých existujúcich objektov, ktoré sú porovnané so skutočným ohodnotením všetkých objektov, obsiahnutých v skrytom užívateľovi (viď. obrázok 4.3). Generované ohodnotenie reprezentuje preferencie užívateľa z pohľadu modelu rozhodovania, ktorý vychádza z doposiaľ známych informacií o preferenciách.

4.2 Cieľ, problémy a výsledky

Cieľom je aby užívateľ ohodnotil, čo najmenej objektov (veľa hodnotení je pre užívateľa zatažujúce) a vzniknutý model rozhodovania bol čo najpresnejší. Respektíve, aby modelom vygenerované ohodnotenie bolo čo najvernejšou kópiou skutočného ohodnotenia všetkých objektov.

Problémy plynúce z navrhnutého experimentu na hľadanie čo najefektívnejšieho modelu rozhodovania je možné rozdeliť do niekoľkých skupín.

- **Výber nasledujúceho objektu**, ktorý má byť ponúknutý užívateľovi k ohodnoteniu. Výber sa aplikuje v každom kroku (viď. 4.5).



Obrázek 4.3: Analýza výsledku

- **Model rozhodovania** je napodstatnejšou časťou. Za model rozhodovania je možné použiť akýkoľvek nástroj schopný postupného učenia z ohodnotených objektov a predpovedania užívateľského rozhodovania (viď. 4.6).
- **Presnosť** postupne vznikajúceho modelu rozhodovania sa porovnáva s ohodnotením všetkých objektov podľa daného skrytého užívateľa. V skutočnosti takéto ohodnotenie modelu nie je známe a tieto informácie pri práci model nevyužíva (viď. 4.4).

Výsledky sa pozorujú na zmene presnosti modelu pri postupnom ohodnotení objektov. Pri predstave na grafe os x reprezentuje počet ohodnotených objektov, z ktorých bol vytvorený model rozhodovania a os y reprezentuje presnosť modelu. Príklady grafov je možné nájsť v kapitole 5 obsahujúcej výsledky.

4.3 Použité typy užívateľov

Požiadavky jednotlivých užívateľov su rozdielne a vzájomne sa líšia, pre potreby experimentu boli definovaní štyria modeloví užívatelia. Dá sa povedať, že každý z nich reprezentuje určitú skupinu užívateľov, v exerimente sú však považovaní za jednotlivcov.

Kancelária: Tento typ užívateľov uprednostňuje na prvom mieste cenu, potom kvalitu a výdrž batérie. Nekladie sa dôraz na odolnosť, prenositelnosť, nie je potrebný 3D výkon ani veľká RAM pamäť. Užívateľ tejto kategórie väčšinou používa Windows XP (Vista, 7) plus balíček MS Office prípadne iný, špecializovaný program potrebný pre prácu (napríklad účtovníctvo), internet a občasné prehrávanie multimédií (hudba alebo filmy).

- Cena: 15 20 tisíc Kč. Cena je hlavným limitom, preto jej rozpätie sa drží v dolných cenových hladinách ponuky.
- Procesor: AMD 1.8 – 2.2 GHz
- Pamäť: 1 - 3 GB RAM DDR2
- Display: 15.4”
- Grafická karta: integrovaná (zdieľaná pamäť)
- Značka: Acer, Asus

Manažér: Predstavuje užívateľa, ktorý je neustále v pohybe a potrebuje mať svoj počítač pri sebe. Dôraz je kladený na hmotnosť, rozmery a výdrž baterie. Samotný výkon nie je úplne zanedbateľný a zvládá už náročnejšie aplikácie. Kvôli nadštandardným požiadavkám (výkon a mobilita) je cena vyššia ako v prípade kancelárskych notebookov. Umožňuje používať akékoľvek potrebné aplikácie a zároveň dlhodobo cestovať s počítačom (dlhá služobná cesta, prípadne zamestnanie vyžadujúce mobilitu). Hlavnou výhodou týchto notebookov sú ich malé rozmery a z toho vyplývajúca

malá hmotnosť (2 Kg) a odolnosť. Hardwareová výbava je pritom pomerne výkonná a udržuje si nízku spotrebu a tým aj vysokú výdrž batérie (3 hodiny).

- Cena: 40 – 50 tisíc Kč
- Procesor: Intel, AMD 1.6 – 2.5 GHz
- Pamäť: 3 - 5 GB RAM DDR2
- Display: 12" - 14.1"
- Grafická karta: integrovaná (zdieľaná pamäť)
- Značka: Hewlet-Packard, Fujitsu-Siemens ale aj ostatní výrobcovia

Hráč: Užívateľ vyžaduje hlavne výkon pre najnovšie hry, ktoré sú veľmi náročné na celkovú výkonnosť počítača. Cena a výdrž batérie v tomto prípade nehrá dôležitú úlohu. Podstatný je výkonný procesor, veľká RAM pamäť a hlavne grafická karta. Užívateľ sa zameriava na najnovšie technológie.

- Cena: 30 – 60 tisíc Kč, alebo bez limitu
- Procesor: Intel 2.3 – 3.06 GHz a viac
- Pamäť: 4 - 8 GB RAM DDR3
- Display: 15.4" - 17" a viac
- Grafická karta: samostatné grafické karty (nezdieľaná pamäť)
- Značka: Dell, Hewlet-Packard, Ferrari, Acer

Desktop: Ide o užívateľa, ktorý sa rozhodol pre notebook plne nahrádzajúci domáci desktopový počítač. Pomerne dôležitý je výkon, ktorý by mal byť porovnateľný s desktopovou variantou. Kvôli univerzálnosti sú dôležité porty. Naopak sa nekladie dôraz na výdrž batérie a celkovú odolnosť počítača. Doplnkom vybavenia býva dokovacia stanica s dobíjaním batérie a pripojením ďalších komponentov, taktiež

z dôvodu nízkej (až žiadnej) mobility sa notebook napája na väčší monitor. Hardware týchto notebookov je pomerne výkonný a spadá medzi manažérske a hráčske notebooky. Cenová hladina je vyššia ako u kancelárskych notebookov, ale pre nízku prenositelnoť je možné zvýšiť vykonnosť.

- Cena: 20 – 30 tisíc Kč
- Procesor: Intel, AMD 2.0 – 2.5 GHz
- Pamäť: 3 - 5 GB RAM DDR2
- Display: 15.4”
- Grafická karta: samostatné grafické karty (nezdieľaná pamäť)
- Značka: všetci výrobcovia

4.4 Porovnávanie presnosti výsledku

K porovnávaniu presnosti a zmeny jednotlivých výsledkov bol zvolený korelačný koeficient a výpočet priemernej absolútnej chyby[1]. Použité metódy sú názorne ukázané na príklade uvedenom v nasledujúcej tabuľke 4.1, ktorá ukazuje dve postupnosti ohodnotení G a U, kde G reprezentuje generované ohodotenie objektov modelom rozhodovania a U skutočné ohodnotenie objektov užívateľom. Podľa generovaného ohodnotenia je nalepším objektom objekt s $ID = 4$ a podľa užívateľa je najleším objektom objekt s $ID = 2$

4.4.1 Korelačný koeficient

Korelačný koeficient reprezentuje lineárnu závislosť medzi dvoma premennými, inak tiež meria silu závislosti medzi dvoma kvantitatívnymi premennými. Premenná G nezávisí na premennej U, ale dve náhodné premenné G a U sa spoločne menia. Korelačný koeficient môže byť vyjadrený nad poradovými alebo neporadovými premennými. Taktiež množina porovnávaných hodnôt môže byť rozdielna. Pre potreby tejto

[1]Mean Absolute Error

Id	G	U
1.	6	6
2.	9	1
3.	2	6
4.	5	9
5.	1	4

Tabulka 4.1: Príklad dvoch ohodnotení

práce boli vybrané len koeficienty porovnávajúce poradové premenné nad rovnakou množinou hodnôt. Korelačný koeficient nadobúda hodnoty na intervale $[-1, 1]$, kde -1 predstavuje nezhodu pre každé $i \neq j$ a 1 predstavuje zhodu pre každé $i \neq j$. Podľa J.Cohena [3] je korelácia pod $0,1$ triviálna, $0,1 - 0,3$ malá, $0,3 - 0,5$ stredná a nad $0,5$ veľká. Korelácia $0,7 - 0,9$ sa často uvádza ako veľmi veľká a $0,9 - 1$ ako takmer dokonalá. Vzťahy medzi dvoma premennými z množiny G a U s indexom i a j, kde $i \neq j$ teda g_i, g_j, u_i a u_j sú

Súhlasné(C - Concordant)	$(g_i < g_j)\&(u_i < u_j)$	alebo $(g_i > g_j)\&(u_i > u_j)$
Nesúhlasné(D - Discordant)	$(g_i > g_j)\&(u_i < u_j)$	alebo $(g_i < g_j)\&(u_i > u_j)$
Zhoda v X a $Y(XY0)$	$(g_i = g_j)(u_i = u_j)$	
Zhoda v $X(X0)$	$(g_i = g_j)(u_i \neq u_j)$	
Zhoda v $Y(Y0)$	$(g_i \neq g_j)(u_i = u_j)$	

Aplikovaním na príklad uvedený v tabuľke 4.1 vychádza

i	j	$g_i <=> g_j$	$u_i <=> u_j$	typ
$i=1$	$j=2$	$6<9$	$6>1$	Nesúhlasné
$i=1$	$j=3$	$6>2$	$6=6$	Zhoda v Y
$i=1$	$j=4$	$6>5$	$6<9$	Nesúhlasné
$i=1$	$j=5$	$6>1$	$6>4$	Súhlasné
$i=2$	$j=3$	$9>2$	$1<6$	Nesúhlasné
$i=2$	$j=4$	$9>5$	$1<9$	Nesúhlasné
$i=2$	$j=5$	$9>1$	$1<4$	Nesúhlasné
$i=3$	$j=4$	$2<5$	$6<9$	Súhlasné
$i=3$	$j=5$	$2>1$	$6>4$	Súhlasné
$i=4$	$j=5$	$5>1$	$9>4$	Súhlasné

Celkový počet súhlasných párov je 4, nesúhlasných 5 a nastala jedna zhoda v Y.

Kendall Tau-A Tento koeficient [15] vyjadruje rozdiel medzi pravdepodobnosťou dvoch hodnôt premenných v rovnakom poradí oproti tomu, že tieto dve premenné nie sú v rovnakom poradí.

$$\tau_A = \frac{C-D}{(N*(N-1))/2} \tag{4.1}$$

kde N je počet prvkov množiny G (U). Na vyššie uvedenom príklade je Kendall Tau-A koeficient rovný $-0,1$ $((4-5)/10)$.

Kendall Tau-B Nevýhodou Kednall Tau-A koeficienu je, že neberie do úvahy zhodné páry objektov. Preto je vhodnejšie použiť Kendall Tau-B koeficient.

$$\tau_B = \frac{C-D}{\sqrt{(C+D+X0)*(C+D+Y0)}} \tag{4.2}$$

Z príkladu je hodnota koeficientu rovná $\frac{-1}{\sqrt{90}} = -0,105$. Pri absencii zhodných párov je Kendall Tau-A koeficient rovný Kendall Tau-B koeficientu.

4.4.2 Priemerná absolútna chyba

Jedná sa o spôsob definovania podobnosti na základe výpočtu priemernej odchylky medzi jednotlivými pármi g_i a u_i pre každé i z n.

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|x_i - y_i| \tag{4.3}$$

V modelovom príklade má priemerná absolútna chyba hodnotu $3,8$ ($\frac{|6-6|}{5} + \frac{|9-1|}{5} + \frac{|2-6|}{5} + \frac{|5-9|}{5} + \frac{|1-4|}{5} = \frac{19}{5}$). Pri ohodnocovaní objektov hodnotami od 1 do 10 a pri braní v úvahu zaokrúhľovanie hodnôt sa dá povedať, že veľkosť MAE od 0 do 0.5 je hodnotenie velmi presné. Pri porovnávaní presnosti na základe korelačného koeficientu a priemernej absolútnej chyby je možné sledovať súvislosť a to, že pri veľkej hodnote korelačného koeficientu (približujúceho sa k 1) je samotná hodnota priemernej absolútnej chyby zanedbateľná. Z tohoto prípadu vyplýva, že usporiadanie objektov modelom rozhodovania je takmer rovnaké ako skutočné usporiadanie, aj keď ohodnotenie bolo posunuté. Na druhej strane malá hodnota priemernej absolútnej chyby naznačuje pomerne presné ohodnotenie objektov a to aj pri korelačnom koeficiente približujúcom sa k 0. Sledovaním oboch nástrojov (korelačného koeficientu a najmenšej absolútnej chyby) je možné pomerne presne hodnotiť efektívnosť modelu rozhodovania.

4.4.3 Porovnanie top–k hodnotení

V praxi je často hlavnou myšlienkou nájsť len najpreferovanejšie objekty a nie určiť poradie aj tých najmenej preferovaných, a teda nezaujímavých objektov. Preto je efektívnosť jednotlivých modelov sledovaná aj na základe podobnosti top k množín najlepších objektov. Teda pri porovnaní top k množín sa zoberie množina U_k obsahujúca k najlepších užívateľom ohodnotených objektov a množina G_k predstavujúca k najlepších modelom ohodnotených objektov a vypočítava sa percentuálna podobnosť množiny, teda koľko percent z množiny G_k

$$top - k = \frac{G_k}{U_k * 100}$$

je obsiahnutých v množine U_k. Pri ohodnocovaní objektov z malej množiny hodnôt R a pri dostatočne malom k nie sú množiny U_k a G_k rovnako veľké. V tomto prípade musia tieto množiny obsahovať aspoň k hodnotení. Viac o hľadaní top k najlepších objektov je možné nájsť v [32] a [31].

4.5 Výber objektu

Pri výbere objektu je možné použiť viacero metód. Hlavnou myšlienkou je, aby informácia o ohodnotenom objekte bola pre model rozhodovania čo najprínosnejšia. Teda jej ohodnotenie čo najviac spresnilo samotný model. Pri výbere platí podmienka, že každý objekt môže byť užívateľom ohodnotený len raz. Jednotlivé možnosti výberu objektov určených k ohodnocovaniu budú ukázané na príklade piatich objektov s tromi atribútmi, z ktorých prvý nadobúda hodnoty $\{A, B, C\}$, druhý $\{K, L, M\}$ a tretí $\{X, Y, Z\}$. Z tabuľky 4.2 je možné vypočítať výskyt jednotlivých hodnôt atri-

ID	Atribút 1	Atribút 2	Atribút 3	Výskyt
1	A	K	X	5
3	B	L	Y	4
4	A	L	Z	8
5	A	M	Z	8
6	C	M	Z	6

Tabulka 4.2: Príklad dvoch ohodnotení

bútov vo všetkých objektoch, takže $A = 3$, $B = 1$, $C = 1$, $K = 1$, $L = 2$, $M = 2$, $X = 1$, $Y = 1$ a $Z = 3$. Následne je možné spočítať celkový výskyt jednotlivých objektov (viď. tabuľka 4.2).

Lineárny výber: pri tomto výbere sa berie v úvahu len ID objektu. Výber podľa ID objektu neobsahuje žiadnu predikciu, keďže pod ID sa môže skrývať objekt s ľubovoľnými vlasnosťami. Na príklade z tabuľky 4.2 sa ako prvý zoberie objekt s $ID = 1$, ako druhý objekt s $ID = 3$, keďže objekt s $ID = 2$ nie je v príklade

obsiahnutý. Tento výber neberie v úvahu prínos vybraného objektu.

Maximálny výskyt: je založený na súčte výskytov jednotlivých atribútov obsiahnutých v objekte. Pri maximálnom výskyte sa berie ako prvý objekt s najväčším výskytom jednotlivých atribútov. Na príklade z tabuľky 4.2, by boli ako prvé dva vybrané objekty s $ID = 4$ a $ID = 5$, na poradí nezáleží, keďže ich výskyt je rovnaký.

Minimálny výskyt: výber je založený na podobnom princípe ako u maximálneho výskytu, ale pri minimálnom výskyte sa berie ako prvý objekt s najmenším výskytom jednotlivých atribúov. Z príkladu v tabuľke 4.2 je prvým vybraným objekt s $ID = 3$ a druhý s $ID = 1$.

Najlepšie/Najhoršie ohodnotený: pri tomto výbere sú všetky objekty, ktoré ešte užívateľ nehodnotil, ohodnotené modelom rozhodovania. Na základe tohto ohodnotenia je vybraný objekt s najlepším/najhorším ohodnotením a ten je predaný užívateľovi k hodnoteniu.

Najväčšia informácia: tento výber berie do úvahy už použité objekty a vyberá objekt, ktorý obsahuje čo najviac nepoužitých vlasností jednotlivých atribútov. Prvý objekt je vybraný náhodne, keďže ešte nebola použitá žiadna z vlasností. Napríklad, ako prvý sa vyberie objekt s $ID = 4$, ktorý obsajuje informáciu o vlastnostiach A, L a Z. Na základe toho objekt s $ID = 1$ obsahuje nové informácie o dvoch vlastnostiach K a X, objekt s $ID = 3$ o B a Y, objekt s $ID = 5$ jedine o M a objekt s $ID = 6$ o vlastnostiach M a C. Takže ako druhý bude vybraný objekt 1,3 alebo 5.

Najlepšie/najhoršie ohonotený výber spolu s výberom podľa najväčšej informácie nedefinujú výber prvého objektu a taktiež pri každom ďalšom výbere objektu môžu poskytnúť väčšie množstvo objektov. K získaniu väčšej kontroly nad výberom objektu je možné na túto skupinu objektov následne použiť lineárny výber alebo výber podľa najväčšieho/najmenšieho výskytu.

V práci sú rozlíšené primárne a sekundárne výbery objektov. Primárne sa uplatňujú ako prvé na výber zo všetkých objektov, následne je na možnosti, vyhovujúce primárnemu výberu, aplikovaný sekundárny výber. V práci boli medzi primárne výbery zaradené najlepšie/najhoršie hodnotený objekt a výber podľa najväčšej informácie. Medzi sekundárne výbery potom patrí minimálny/maximálny výskyt a linárny výber.

4.6 Model rozhodovania

Navrhnutý model rozhodovania je založený na výpočte preferencií objektov na základe pravdepodobností. Model postupne vytvára (upravuje), maticu obsahujúcu rozloženie pravdepodobností jednotlivých hodnotení pre každú použitú vlastnosť jednotlivých atribútov. Takže ak $|R|$ je počet ohodnotení (v experimente $|R| = 10$) a počet hodnôt všetkých atribútov je $|A'| = \sum_{k=1}^{m} |A^k|$, kde $|A^k|$ je počet atribútov k–teho atribútu, potom maximálna veľkosť matice je $|R| \times |A'|$. Model taktiež postupne vytvára maticu rozloženia pravdepodobností pre každú použitú dvojkombináciu atribútov A^k a A^l, kde $k \neq l$. Maximálny celkový počet riadkov matice je $|A''| = \sum_{k=1}^{m} \sum_{l=k+1}^{m} |A^k| * |A^l|$.

Ohodnotenie objektu K výpočtu ohodnotenia neznámeho objektu sú použité všetky vyhovujúce riadky. To znamená všetky riadky, ktorých vlastnosť alebo dojkombinácia vlastností je obsiahnutá v hodnotiacom objekte. Na výpočet ohodnotenia neznámeho objektu môže byť použitá:

- len tabuľka obsahujúca samostatné vlastnosti atribútov
- len tabuľka obsahujúca dvojkombináciu vlastnosti atribútov
- obe tabuľky pravdepodobnostného rozdelenia

Z príslušných vyhovujúcich riadkov pravdepodobnostného rozdelenia sú vypočítané (vybrané) ohodnotenia objektu v závislosti na danej vlastnosti alebo dvojkombinácii

vlastností. Možnosti hodnotenia objektu v závislosti na vlastnosti alebo dvojkombinácií vlastností objektu sú buď:

- vybrať najlepšie hodnotenie z hodnotení s najväčšou pravdepodobnosťou

$$\max(r_i), \text{ kde } r_i \in \max(P(r_i)) \tag{4.4}$$

- vybrať najhoršie hodnotenie z hodnotení s najväčšou pravdepodobnosťou

$$\min(r_i), \text{ kde } r_i \in \max(P(r_i)) \tag{4.5}$$

- vypočítať priemerné hodnotenie

$$\sum_{i=1}^{n}(r_i * P(r_i)) \tag{4.6}$$

kde r_i je hodnotenie $(1, \ldots, 10)$ a $P(r_i)$ je pravdepodobnosť hodnotenia r_i v danom riadku pravdepodobnostného rozdelenia. Vysledné ohodnotenie objektu je vypočítané ako aritmetický priemer všetkých hodnotení použitých riadkov $\frac{1}{l}\sum_{i=1}^{n}(r_i * P(r_i))$. Modelový príklad v tabuľke 4.3 ukazuje ohodnotenie po prvých piatich krokoch.

Krok	RAM	Výrobca	Cena	Rating
1.	1024	*Dell*	10k–20k	2
2.	1024	*HP*	10k–20k	1
3.	2048	*HP*	20k–30k	2
4.	2048	*Dell*	20k–30k	3
5.	4096	*HP*	20k–30k	3

Tabulka 4.3: Príklad hodnotenia prvých piatich notebookov.

Teda ohodnotenie prvých piatich notebookov s tromi vlastnosťami *RAM*, *Cena* a *Výrobca*. Tabuľka 4.4a ukazuje pravdepodobnostné rozdelenie jednotlivých vlastností atribútov a tabuľka 4.4b pravdepodobnostné rozdelenie použitých dvojkombinácií atribútov. Vypočítané ohodnotenia notebooku od výrobcu *HP*, veľkosťou RAM 2048MB a cenou od 10 000Kč do 20 000Kč závisia na použitom výbere pravdepodobnosti z možných riadkov a taktiež na použitých tabuľkách. Jednotlivé ohodnotenia

Rating			Level Jedna
1	2	3	
0.5	0.5	0	1024
0	0.5	0.5	2048
0	0	1	4096
0.5	0.5	0	10k–20k
0	0.33	0.66	20k–30k
0	0.5	0.5	*Dell*
0.33	0.33	0.33	*HP*

(a) Použité atribúty

Rating			Level Dva
1	2	3	
1	0	0	1024–*HP*
0	1	0	2048–*HP*
0	0	1	4096–*HP*
0	1	0	1024–*Dell*
0	0	1	2048–*Dell*
0.5	0.5	0	10k–20k–1024
0	0.5	0.5	20k–30k–2048
0	0	1	20k–30k–4096
1	0	0	*HP*–10k–20k
0	0.5	0.5	*HP*–30k–30k
0	1	0	*Dell*–10k–20k
0	0	1	*Dell*–20k–30k

(b) Dvojkombinácia atribútov

Tabulka 4.4: Príklad pravdepodobnostného rodelenia pre tabuľku 4.3

sú ukázané v tabuľke 4.5. V tabuľke riadky reprezentujú spôsob výpočtu preferencie z jedného riadku pravdepodobnostného rozdelenia a stĺpce reprezentujú aké tabuľky boli k výpočtu použité.

	Úroveň jedna	Úroveň dva	Obe úrovne
Nalepší rating	2.66	2.5	2.2
Najhorši rating	1.33	2	1.4
Priemerný rating	2	2.25	1.8

Tabulka 4.5: Výsledné hodnotenie pre notebook *HP*, RAM 2048MB a cena 10 000Kč–20 000Kč

Kapitola 5

Analýza výsledkov experimentu

Všetky testy sú prevádzané na sade reálnych notebookov prevzaných z [49]. Veľkosť testovacej sady je 800 rozdielnych a reálnych notebookov. Každý notebook obsahuje jedinečné atribúty ID a Url z ktorých ID slúži ako primáry kľúč a Url nie je zahrnuté do testov, kedže nadobúda vždy rozdielnu hodnotu. Ďalej obsahuje 13 rozdielnych atribútov cena ($Price$), veľkosť LCD v palcoch (LCD), výrobca procesoru (CPU_maker), výkon procesoru v GHz (CPU_GHz), typ RAM pamäte (RAM_type), veľkosť RAM pamäte (RAM_MB), prevácková frekvencia RAM pamäte (RAM_MHz), typ a otáčky HDD (HDD_type), veľkosť HDD (HDD_MB), informácia o tom či obsahuje alebo neobsahuje DVD–ROM (DVD_ROM), výrobca notebooku ($Brand$), váha notebooku ($Weight\_Kg$) a výdrž batérie ($Battery\_H$), z ktorých všetky sú požité pri výpočte modelu.

Jednotlivé testy boli prevadzané na štyroch rozdilenych skrytých uživateloch popisaných v kapitole 4.3. Výsledky testov sú rozdelené do štyroch skupín podľa spôsobu hodnotenia presnosti modelu a to na Kendall Tau-A koerelačný koeficient, Kendall Tau-B korelačný koeficient, výpočet najmenšej absolutnej chyby a porovnanie percentuálnej podobnosti top–k objektov. Jednotlivé možnosti porovnania presností sú blyžšie rozobrané v kapitole 4.3.

Vo výsledných grafoch je navrhnutý model porovnaný s modelom založeným na teorií Support Vector Regression (viď. kapitola 2.5.4), ktorý bol spustený za tých istých

podmienok ako navrhnutý model UserDecision. Použitá implementácia modelu Support Vector Regression bola prevzaná z [48], ktorá je založená na voľne šíritelnej licencii GPL. Viac je možné nájsť na [11].

5.1 Nastavenie aplikácie UserDesicion

Celková variabilita nastavení aplikácie je pomorne široká a preto bolo urobeních veľké množstvo testov, ktoré nie sú zahrnuté vo výsledkoch. Niektoré z testov dosahovali veľmi kolísajúce výsledky pri špecifických nastaveniach. K finálnemu hodnoteniu bola zo všetkých možných globálnych nastavení výbraná kombinácia, pri ktorej model UserDecision dosahoval najlepšie a najvyrovnanejšie výsledky:

Výber objektu pri výbere objektu (viď kapitola 4.5) sa ukázalo, že najväčším prínosom pre model bola kombinácia:

- **primárny výber** podľa najmenšieho použitia
- **sekundárny výber** podľa minimálneho výskytu

Typ rozdelenia efektívnejším rozdelením spojitých atribútov je podľa testov ekvifrekvenčné rozdelenie oproti ekvidistančnému rozdeleniu (viď kapitola 2.3)

Počet skupín rozdelenia spojitých atribútov (viď kapitola 2.3) sa na testovaných dátach ukázal ako 5 pričom jeho rozdiel oproti 4 a 6 bol pomerne malý

Pre nastavenie modelu UserDecision (viď. kapitola 4.6) sa najlepšie a najvyrovnanejšie výsledky dosahli pri nastaveniach:

Úroveň tabuliek je najlepšie nastaviť len na samotnú dvojkombináciu vlastností atribútov teda na úroveň dva. Požitie tabuľky len úrovne jedna sa ukázalo ako slabé pre špeciálne prípady užívateľov a použitie oboch úrovni dosahovalo v priemere o niečo slabšie výsledky ako pri použití len druhej úrovne.

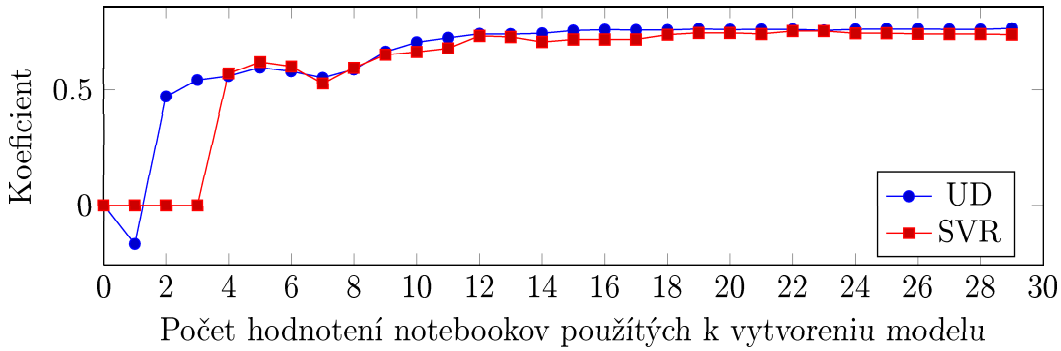
Výber pravdepodobnosti dosahoval vo šetkých nastaveniach porovnatelné výsledky ale o málo lepšie sa ukázal výbočet aritmetického priemeru oproti výberu najlepšieho a najhoršieho hodnotenia s najväčšou pravdepodobnosťou.

Nastavenie top–k Porovnanie top–k skupín sa ukázal ako pomerne špecifický problém. Pre každého užívateľa bolo vidieť najleší rast percentuálnej podobností pri rozdielnych hodnotách. Preto sa za hodnotu zvolila $k = 30$ ktorá korešponduje s reálnym svetom, kde viac ako 30 najlepších objektov už nie je tak zaujímavých. V každom prípade je zaujímavé sa zaoberať týmto smerom aj ďalej.

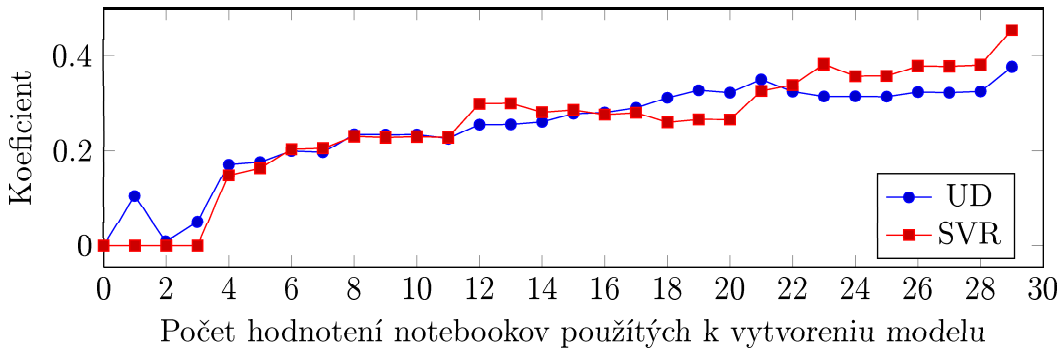
5.2 Porovnanie

Ako už bolo povedané navrhnutý model UserDecision bol testovaný na štyroch rozdielnych užívateľoch a výsledky boli hodnotené štyrmi rozdielnými spôsobmi. Vo všetkých prípadoch je výsledok modelu porovnaný s modelom vychádzajucim zo SVR. Prvá skupina štyroch grafov obsahuje porovnanie štyroch užívateľov podľa Kendall Tau-B kocficientu. Druhá skupina štyroch grafov obsahuje porovnanic štyroch užívateľov podľa Kendall Tau-A koeficientu. Ďalšia skupina štyroch grafov ukazuje porovnanie štyroch užívateľov podľa najmenšej aboslutnej chyby. Nakoniec posledná skupina štyroch grafov obsahuje porovnanie štyroch užívateľov podľa percentuálnej podobnosti top k objektov.

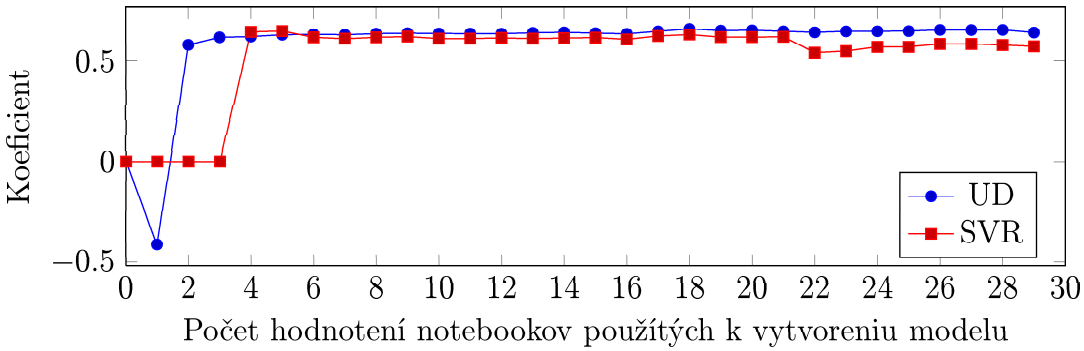
Z grafov (obrázoky, 5.1, 5.2, 5.3 a 5.4) porovnávajúcich užívateľov podľa Kendall Tau–B koeficientu vidieť, že model UserDecision nadobúda stabilnejších hodnot skôr ako model SVR. Taktiež je možné vypozorovať, že dosahuje o málo lepšiu hodnotu a kolísanie krivky je menšie.



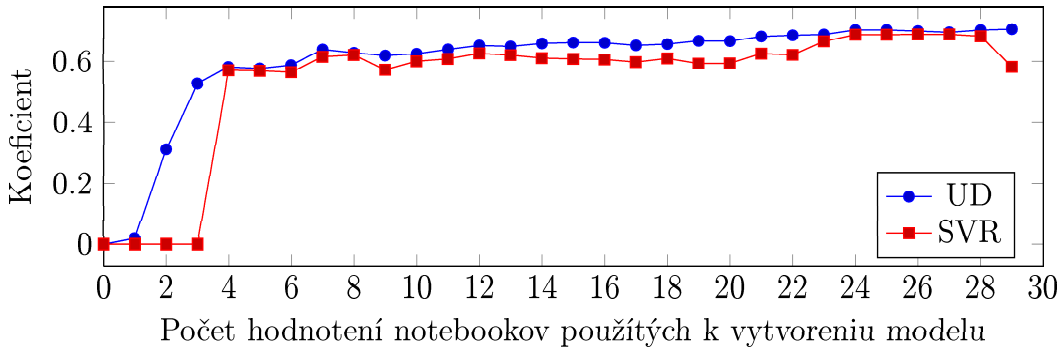
Obrázek 5.1: Desktop užívateľ hodnotený podľa Kendall Tau–B koeficientu.



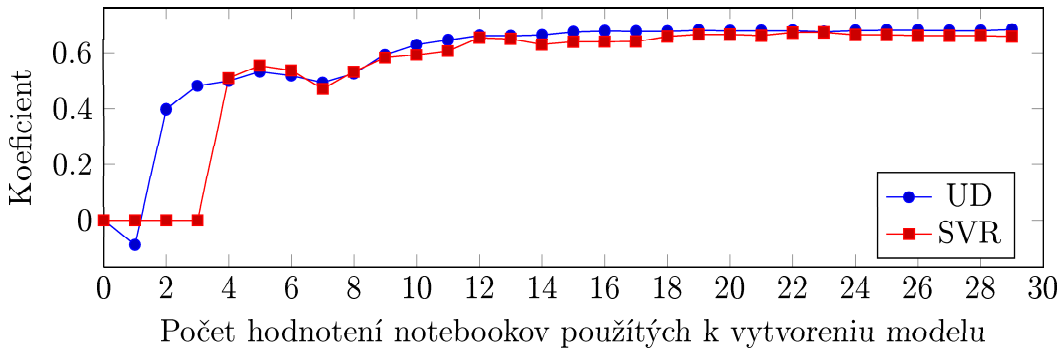
Obrázek 5.2: Užívateľ manažér hodnotený podľa Kendall Tau B koeficientu.



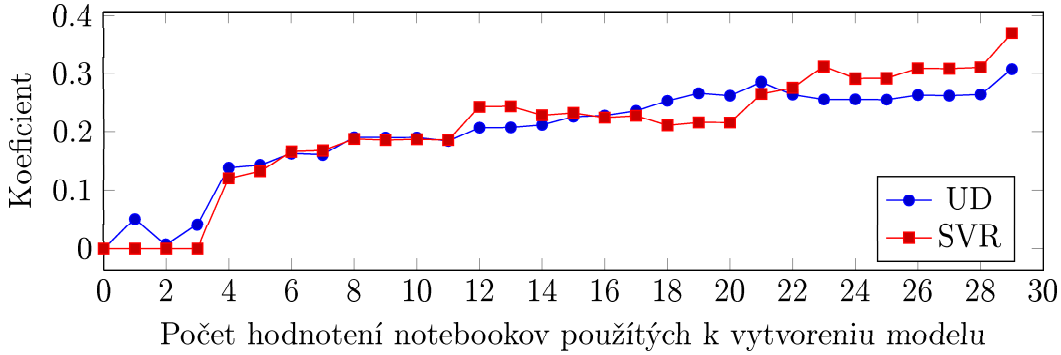
Obrázek 5.3: Užívateľ hráč hodnotený podľa Kendall Tau–B koeficientu.



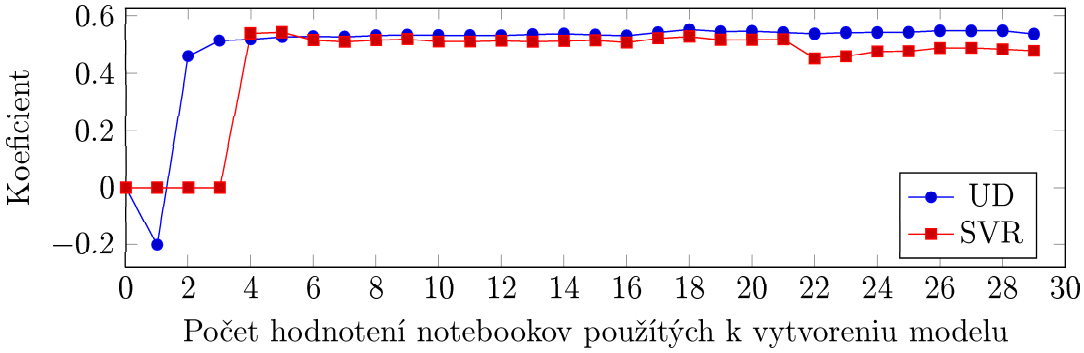
Obrázek 5.4: Kanceársky užívateľ hodnotený podľa Kendall Tau–B koeficientu.



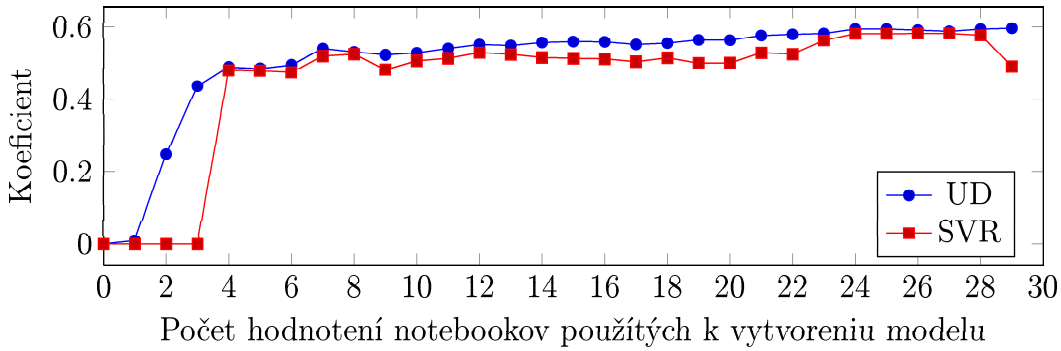
Obrázek 5.5: Desktop užívateľ hodnotený podľa Kendall Tau A koeficientu.



Obrázek 5.6: Užívateľ manažér hodnotený podľa Kendall Tau–A koeficientu.



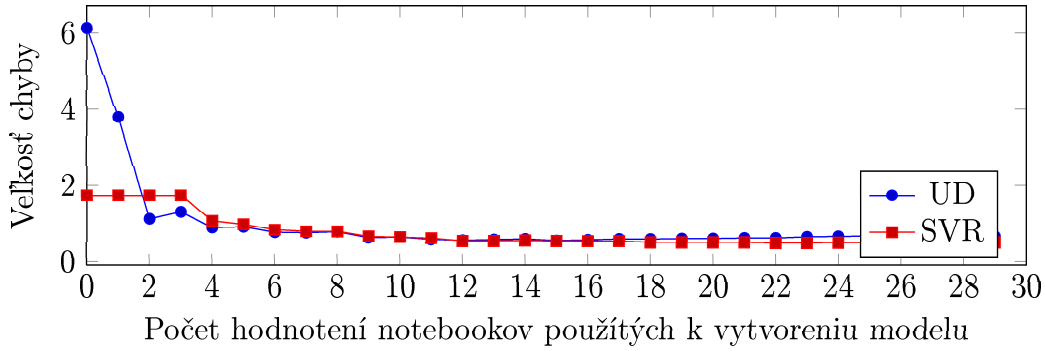
Obrázek 5.7: Hráčský užívateľ hodnotený podľa Kendall Tau–A koeficientu.



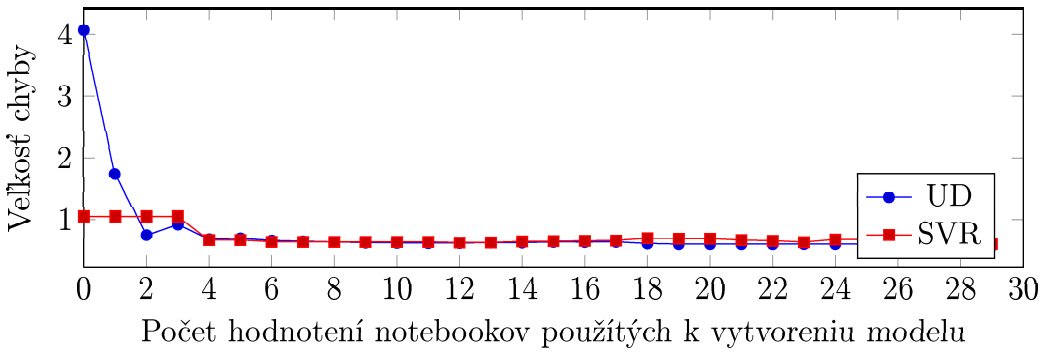
Obrázek 5.8: Kancelársky užívateľ hodnotený podľa Kendall Tau A koeficientu.

Z grafov (obrázoky, 5.5, 5.6, 5.7 a 5.8) porovnávajúcich užívateľov podľa Kendall Tau A koeficientu vidieť, že model UserDecision v porovnaní so SVR sa správa podobne ako pri hodnotení podľa Kendall Tau B koeficientu. To výplýva z podobnosti výpočtu oboch koeficientov. Takže UserDecision model nadobúda stabilnejších hodnôt skôr ako model SRV, dosahuje o niečo lepšiu hodnotu a kolísanie krivky je menšie.

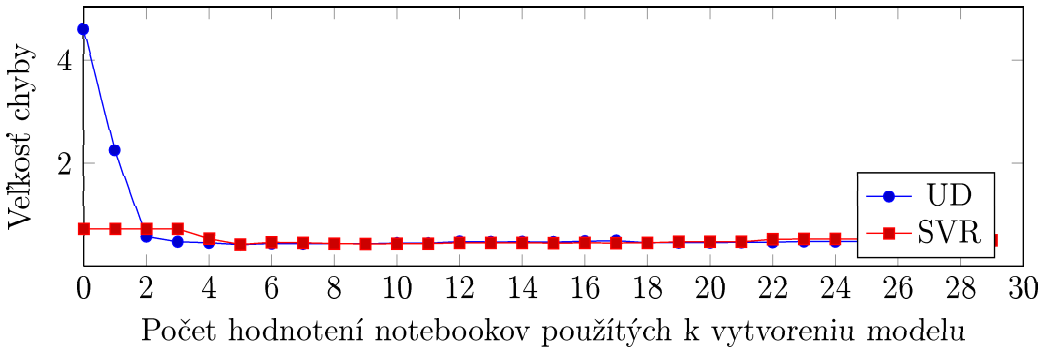
Grafy (obrázoky, 5.9, 5.10, 5.11 a 5.12) obsahujúce porovnanie užívateľov podľa najmenšej absolutnej chyby ukazújú veľkú hodnotu chyby modelu UserDecision ale len v začiatku pri hodnotách do 4-och ohodnotených objektov narozdiel od modelu SRV. Obidva modely sa pri pomerne malom množstve ohodnotených objektov ustália na hodnote $0,5$ až $0,6$.



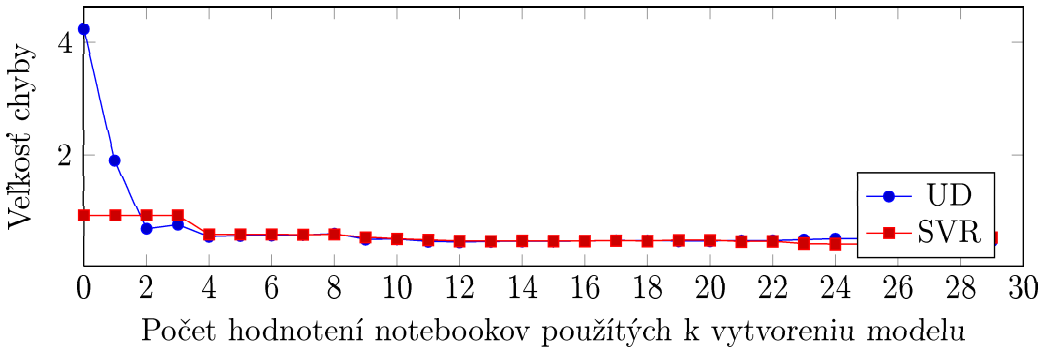
Obrázek 5.9: Desktop užívateľ hodnotený podľa najmenšej absolutnej chyby.



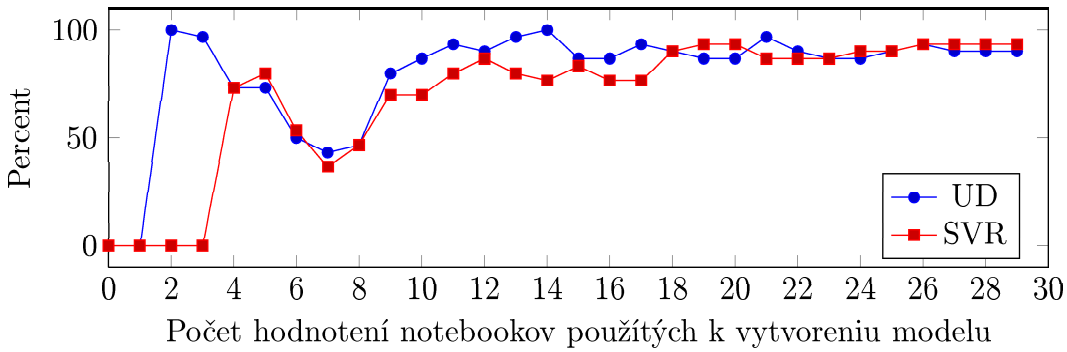
Obrázek 5.10: Užívateľ manažér hornotený podľa najmenšej absolutnej chyby.



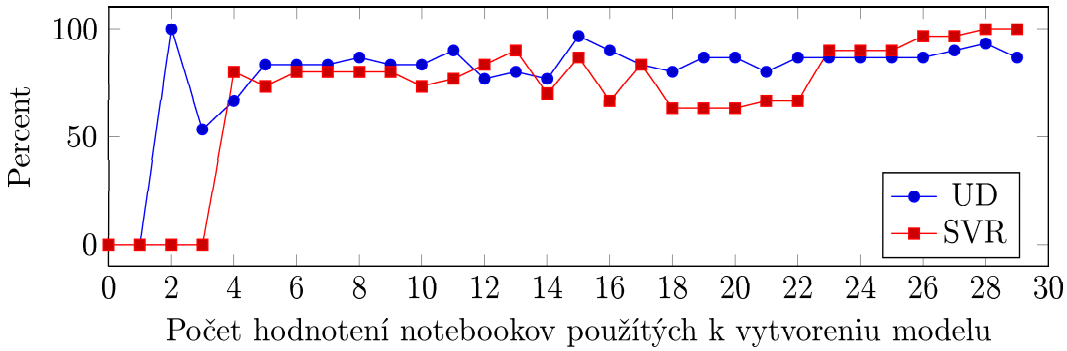
Obrázek 5.11: Užívateľ hráč hodnotený podľa najmenšej absolutnej chyby.



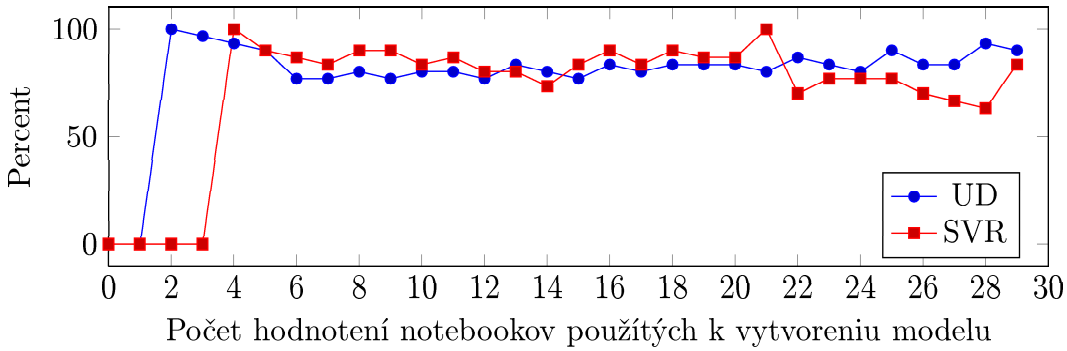
Obrázek 5.12: Kancelársky užívateľ hodnotený podľa najmenšej absolutnej chyby.



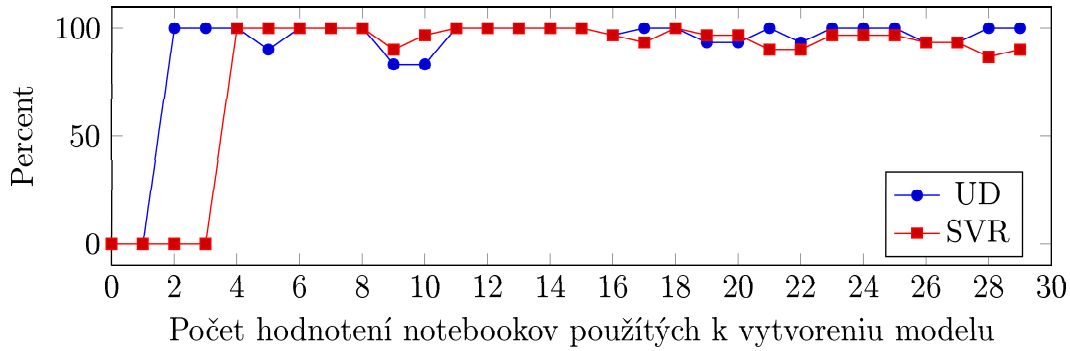
Obrázek 5.13: Desktop užívateľ hodnotenie podľa top–k.



Obrázek 5.14: Užívateľ manažér hodnotený podľa top–k.



Obrázek 5.15: Užívateľ hráč hodnotený podľa top–k.



Obrázek 5.16: Kancelársky užívateľ hodnotený podľa top k.

Grafy obsahujúce porovnanie užívateľov podľa percentuálnej podobnosti top k objektov sú ukázané v obrázokoch, 5.13, 5.14, 5.15 a 5.16. Ako bolo spomináne porovnávanie modelov na základe tejto podobnosti nieje úplne intuitívne a rozdielnosť sa liši od hodnotiaceho uživateľa uživateľa. Pri hodnotení podľa top–k bolo v oboch modeloch UserDecision a SRV pozorované pomerne veľke kolísanie krivky, ktoré sa však po desiatich ohodnotených notebookoch ustálilo na rozumnej hodnote nad 80%.

Navrhnutý model rozhodovania UserDecision simulujúci rozhovanie užívateľa, sa ukázal ako dostatočne funkčný a silný nástroj pre modelovanie užívateľského rozhodovania. Výsledky experimentov ukázali, že model UserDecision dosahuje porovnateľné výsledky s modelom SRV a v niektorých prípadoch boli dosiahnuté dokonca lepšie výsledky.

5.3 Návrhy na rozširenie

Navrhnutý model sa zaoberá modelovaním uživateľského rozhodovania len nad jedným uživateľom. Preto ďalším rozšírením práce by mohlo byť jeho nachádzanie podobností medzi viacerými užívateľmi a následne aplikovanie vypozorovaných súvislostí k určovaniu preferencií.

Pri vytváraví podobných skupín užívteľov je možne sledovať podobnosť tok k objektov. Kedže sa ukázalo, že po ohodnotení 10 až 20 objektov sa percentuálna podobnosť

top–k 30 objektov pohybuje na hranici 90%.

Ďalšou z možností je testovanie modelu UserDecision na jednej strane nad neúplnými dátami, kde niektoré z informácií o notebookoch nie sú známe. Na druhej strane je možné testovať výsledky pri predefinovaní preferencií určitého atribútu. Inak model vo svojej začiatočnej fáze dostane celkové preferencie o určitom atribúte a v ďalších výpočtoch bude tieto preferencie brať v úvahu.

Kapitola 6

Záver

Jedným z cieľov práce bol teoretický rozbor tematiky užívateľského rozhodovania a modelov užívateľských preferencií. Ťažiskom práce bolo navrhnúť vlastný funkčný model rozhodovania a následne vykonať jeho experimentálne overenie. Práca je rozdelená na viaceré časti. V teoretickej časti práce boli rozobrané a predstavené rôzne pohľady na užívateľské preferencie obecne. Ďalej nasleduje časť obsahujúca prehľad jednotlivých modelov užívateľských preferencií a prehľad ich porovnaní.

Navrhnutý model rozhodovania simulujúci rozhovanie užívateľa je založený na pravdepodobnosti nad preferenciami. Ako model užívateľských preferencií bol zobraný viac–hodnotový model užívateľských preferencií. Tento model sa ukázal ako dostatočne funkčný nástroj pre reprezentáciu užívateľského rozhodovania.

V experimentálnej časti boli popísané vykonané testy spolu s popisom navrhovaného modelu rozhodovania. Efektívnosť a presnosť tohto modelu bola porovnaná s modelom vychádzajúcim z teórie Support Vector Regression (SVR). Výsledky experimentov ukázali, že tieto dva modely dosahujú navzájom porovnateľné výsledky. V niektorých experimentoch boli podľa navrhovaného modelu rozhodovania dosiahnuté dokonca lepšie výsledky. Je teda možné konštatovať, že hlavný prínos tejto práce spočíva v tom, že navrhnutý model je plnohodnotným naśtrojom na modelovanie užívateľského rozhodovania.

Ďalším prínosom tejto práce je zostavenie širokého prehľadu v pomerne mladej oblasti

informatiky, ktorý doteraz ešte nebol komplexne zostavený. Práca obsahuje taktiež prehľad širokého okruhu nástrojov, používaných v oblasti modelovania užívateľských preferencií a užívateľského rozhodovania. Práca preto môže poslúžiť ako študijný materiál pre problematiku modelovania užívateľských preferencií. Aplikácia ktorá vnikla v priebehu tejto diplomovej práce je plne funkčná a môže byť použitá k ďalšiemu testovaniu.

Motiváciou pre ďalšiu prácu môže byť návrh univerzálnejšieho modelu analyzujúceho viacerých užívateľov súčasne. Ďalšou motiváciou pre budúci výskum môže byť otestovanie navrhnutého modelu nad rôznymi ďalšími sadami dát a taktiež nad inými skytými užívateľmi.

Literatura

[1] Apolloni B., Zamponi G., and Zanaboni A. M. (1998): *Learning fuzzy decision trees.* Neural Networks, 11(5):885–895.

[2] Cao-Van K. (2003): *Supervised Ranking, from semantics to algorithms.* Ph.D. dissertation, Ghent University.

[3] Cohen, J. (1969). *Statistical Power Analysis for the Behavioral Sciences.*, 1st Edition, Lawrence Erlbaum Associates, Hillsdale (2nd Edition, 1988).

[4] Cussens J. (1999): *Loglinear models for rst-order probabilistic reasoning.* Proceedings of the Fifteenth Conference on on Uncertainty in Articial

[5] Domingos P. and Richardson M. (2006): Markov logic networks. Machine Learning 62, 107-136.

[6] Domingos P. and Richardson M. (2007): *Markov Logic: A Unifying Framework for Statistical Relational Learning.* Introduction to Statistical Relational Learning 339-371.

[7] Friedman N., Getoor L., Koller D. and Pfeffer A. (2001): *Learning probabilistic relational models.* Rclational Data Mining

[8] Friedman N., Getoor L., Koller D. and Pfeffer A. (1999): *Learning probabilistic relational models.* Proccedings of thc Sixtccnth Intcrnational Joint Conferencc on Articial Intelligence, 1300-1307.

[9] Getoor L., Koller D., Taskar B. and Friedman N. (2001): *Learning Probabilistic Models of Relational Structure.* Proceedings of the Eighteenth International Conference on Machine Learning, 170-177.

[10] Getoor L., Koller D., Taskar B. and Friedman N. (2000): *Learning probabilistic relational models with structural uncertainty.* Proceedings of the AAAI-2000 Workshop on Learning Statistical Models from Relational Data, 13-20.

[11] Hall M., Frank E., Holmes G., Pfahringer B., Reutemann P., Witten I. H. (2009): *The WEKA Data Mining Software: An Update;* SIGKDD Explorations, Volume 11, Issue 1.

[12] Heckerman D., Chickering D. M., Meek C., Rounthwaite R. and Kadie C. (2000): *Dependency networks for inference, collaborative filtering.* Journal of Machine Learning Research, and datavisualization, 49-75.

[13] Heckerman D., Meek C. and Koller D. (2004): *Probabilistic entity-relationship models, PRMs, and plate models.* Proceedings of the ICML-2004 Workshopon Statistical Relational Learning and its Connections to Other Fields, 55-60.

[14] Horváth T. and Vojtáš P. (2004): *GAP - Rule Discovery for Graded Classifciation.* Workshop of Advances in Inductive Rule Learning, 46-63.

[15] Kendall M. G. (1948) *Rank Correlation Methods.* Charles Griffin & Company Limited.

[16] Kersting K. and De Raedt L. (2001): *Bayesian Logic Programs.* Technical Report 52.

[17] Kersting K. and De Raedt L. (2001): *Adaptive Bayesian Logic Programs.* Proceedings of the Eleventh Conference on Inductive Logic Programming 2157, 104 – 117.

[18] Kersting K. and De Raedt L. (2001): *Towards combining inductive logic programming with Bayesian networks.* Proceedings of the Eleventh International Conference on Inductive Logic Programming, 118-131.

[19] Kifer M. and Subrahmanian V. S. (1992): *Theory of generalized annotated logic programming and its applications.* Logic Programing, 12, 335-367.

[20] Kim Cao-Van. *Supervised Ranking, from semantics to algorithms.* Ph.D. dissertation, Ghent University, 2003

[21] Krajčí S., Lencses R. and Vojtáš P. (2004): *A comparison of fuzzy and annotated logic programming.* Fuzzy sets and systems 144, 173 192.

[22] Muggleton S. and Raedt L. D. (1994): *Inductive logic programming: Theory and methods.* Journal of Logic Programming, 19/20:629–679.

[23] Muggleton S. (1996): *Stochastic logic programs.* Advances in inductive logic programming, 254-264. Intelligence, 126-133.

[24] Neville J. and Jensen D. (2003): *Collective classication with relational dependency networks.* Proceedings of the Second International Workshop on Multi-Relational Data Mining, 77-91.

[25] Ng R. and and Subrahmanian V.S. (1992): *Probabilistic Logic Programming.* Information and Computation 101, 2, 150-201.

[26] Ngo L. and Haddawy P. (1997): *Answering queries from context-sensitive probabilistic knowledge bases.* Theoretical Computer Science 171, 147-177.

[27] Ozturké M., Tsoukias A., and Vincke P. (2006): *Preference modelling. Multiple Criteria Decision Analysis: State of the Art Surveys.* Springer New York.

[28] Poole D. (2003): *First-order probabilistic inference.* Proceedings of the Eighteenth International Joint Conference on Articial Intelligence, 985-991.

[29] Poole D. (1993): *Probabilistic Horn abduction and Bayesian networks.* Articial Intelligence, 81-129.

[30] Puech A. and Muggleton S. H. (2003): *A comparison of stochastic logic programs and Bayesian logic programs.* Workshop on Learning Statistical Models from Relational Data

[31] Ondreicka, M., Pokorný J. (2008): *Extending Fagin's algorithm for more users based on multidimensional B-tree.* In: Proc. of ADBIS 2008, P. Atzeni, A. Caplinskas, and H. Jaakkola (Eds.), LNCS 5207, Springer-Verlag Berlin Heidelberg, pp. 199–214.

[32] Ondreicka, M., Pokorný J. (2010): *Efficient Top-K Problem Solvings for More Users in Tree-Oriented Data Structures.* SITIS 2009, Proc. of the 5th International Conference on Signal-Image Technology and Internet-Based Systems, Marrakech, Morocco, November 29 – December 3, 2009. IEEE Computer Society Press.

[33] Quinlan J. R. (1986): *Induction of decision trees.* Mach. Learn., 1(1):81–106.

[34] Rafee Ebrahim (2001): *Fuzzy logic programming.* Fuzzy Sets and Systems 117, 215-230.

[35] Riezler S. (1998): *Probabilistic constraint logic programming.* Doctoral dissertation

[36] Sato T. and Kameya Y. (1997): *PRISM: A symbolic-statistical modeling language.* Proceedings of the Fifteenth International Joint Conference on Articial Intelligence, 1330-1335.

[37] Sato T. and Kameya Y. (2008): *New advances in logic-based probabilistic modeling by PRISM.* Probabilistic Inductive Logic Programming, 118-155.

[38] Shevade S.K., Keerthi S.S., Bhattacharyya C., Murthy K.R.K. *Improvements to SMO Algorithm for SVM Regression.* Technical Report CD-99-16, Control Division Dept of Mechanical and Production Engineering, National University of Singapore.

[39] Smola A. J., Scholkopf B. (1998): *A Tutorial on Support Vector Regression.* NeuroCOLT2 Technical Report Series - NC2-TR-1998-030

[40] Sung Young Jung, Jeong-Hee Hong and Taek-Soo Kim (2005): *A statistical model for user preference.* Knowledge and Data Engineering, IEEE Transactions on 17, 6, 834-843.

[41] Taskar B., Abbeel P. and Koller, D. (2002): *Discriminative probabilistic models for relational data.* Proceedings of the Eighteenth Conference on Uncertainty in Articial Intelligence, 485-492.

[42] Taskar B., Abbeel P., Wong M.-F. and Koller D. (2007): *Relational Markov Networks.* Introduction to Statistical Relational Learning, 176-199.

[43] Vojtáš P. (2001): *Fuzzy logic programming.* Fuzzy Sets and Systems 124, 3, 361-370.

[44] Vojtáš P. and Vomlelová M. (2006): *On models of comparison of multiple monotone classifications.* Proc. IPMU 2006, 1236-1243.

[45] Wellman M., Breese J. S. and Goldman R. P. (1992): *From knowledge bases to decision models.* Knowledge Engineering Review, 7.

[46] Zadeh L.A. (1975): *The concept of a linguistic variable and its application to approximate reasoning.* Information Sciences 1, 119 249.

[47] Zadeh L.A. (1965): *Fuzzy sets.* Information Control 8, 338-353.

[48] http://www.cs.waikato.ac.nz/ ml/weka/

[49] http://www.mironet.cz/

Příloha A

Obsah CD

Priložený CD disk obsahuje všetky súbory potrebné k spusteniu programu UserDecision. CD ďalej obsahuje všetko potrebné na zopakovanie testov, všetky zdrojové kódy, testovacie dáta a text diplomovej práce vo formáte pdf.

- `\text` – text diplomovej práce vo formáte pdf
- `\run` – spustiteľná verzia programu
  - `userdecision.bat` spustiteľný súbor k spusteniu aplikácie UserDecision
  - `settings.properties` konfiguračný súbor
- `\userdecision\bin` – preložené zdojové kódy
- `\userdecision\lib` – použité externé knižnice
- `\userdecision\doc` – vygenerovaná dokumentácia Javadoc
- `\userdecision\src` – zdrojové kódy aplikácie
- `\data` – testovacie dáta vo formáte sql určené ako inmport do databázy
- `\instal\JRE` – inštalačné súbory pre JRE - Java Runtime Environment
- `\instal\MySQL` – inštalačné súbory pre MySQL server.
- `\instal\PostgreSQL` – inštalačné súbory pre PostgreSQL server.

Příloha B

Užívateľská dokumentácia k programu

Jedným z cieľov práce bolo experimentálne porovnať navrhnutý model rozhodovania s modelom založeným na teórii Suppor Vector Reggresion. K tomu bol vytvorený program s grafickým užívateľským rozhraním. Po inštalácií, popísanej v kapitole B.1, je možné program spustiť súborom

`\run\userdecision.bat`.

Celá funkčnosť programu je predstavená v kapitole B.2.

B.1 Inštalácia

K správnej funkčnosti programu UserDecision je potrebné nainštalovať JAVA a databázový server MySQL. Kroky išntalácie:

- 1. inštalácia JAVA
- 2. inštalácia MySQL alebo PostgreSQL
- 3. importovanie testovacích dát
- 4. nastavenie konfiguračného súboru

B.1.1 Java

Prostredie JRE - Java Runtime Environment je určené k spúšťaniu JAVA programov. CD obdsahuje súbor, pomocou ktorého je možné v prostredí operačného systému MS Windows XP nainštalovať JRE (prípadne ďalší priložený súbor pre PC s x64 architektúrou).

`\instal\JRE`

Pre iný operačný systém je potrebné použiť príslušný inštalačný postup na adrese http://java.sun.com/javase/.

B.1.2 MySQL a PostgreSQL

K testovaniu programu je potrebný prístup k databáze MySQL alebo PostgreSQL. V prípade, že používateľ nemá prístup k databáze, je potrebné nainštalovať databázový server na použitom PC. Priložené CD obsahuje inštalačný súbor, pomocou ktorého je možné v prostredí operačného systému MS Windows nainštalovať databázový server MySQL (CD obsahuje aj súbor pre PC s x64 architektúrou).

`\instal\MySQL`

Počas inštalácie je vhodné zadať "Standard Configuration" a neupravovať bezpečnostné nastavenia "Modify Security Settings" (štandardne je prístupové meno "root" a heslo prázdne slovo). V prípade iného operačného systému je potrebné použiť príslušný inštalačný postup, ktorý je možné nájsť na adrese http://www.mysql.com/. CD obsahuje inštalačný súbor aj pre databázový server PostgreSQL.

`\instal\PostgreSQL`

B.1.3 Testovacie dáta

Po nainštalovaní serveru MySQL je nutné vytvoriť testovacie dáta. Testovacie dáta tvoria jednu prednastavenú databázu "dipl". Priložené CD obsahuje súbory s testovacími dátami, ktoré je potrebné importovať do vytvorenej databázy. Nasledujúce poradie súborov (a umiestnenie na CD) je pri importovaní do databázy nutné zachovať.

`\data\create-table-notebook.sql`

`\data\insert-notebook-mironet.sql`

`\data\create-view-rating-desktop.sql`

`\data\create-view-rating-gamer.sql`

`\data\create-view-rating-manager.sql`

`\data\create-view-rating-office.sql`

B.1.4 Konfiguračný súbor

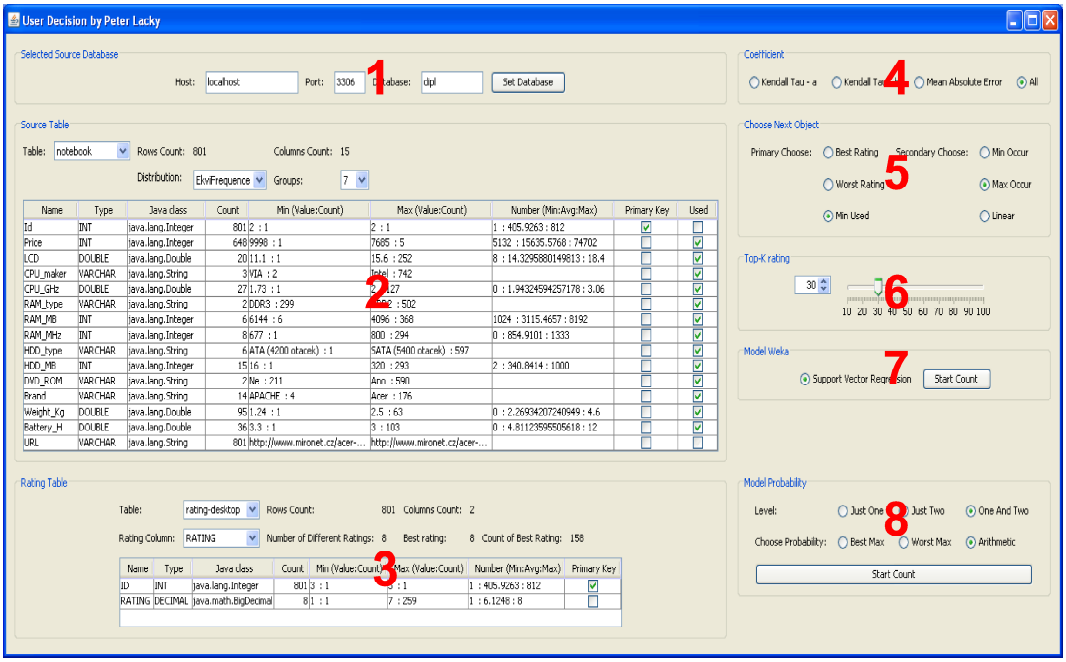
Súbor `\run\settings.properties` obsahuje defaultné nastavenia programu a taktiež rozširuje možnosti v samotnom programe. Obsahom súboru sú premenné

- `DBType` určuje typ databázy. Povolené sú nastavenia `MySQL` (prednastavené) a `PostgreSQL`
- `DBHost` (text) adresa databázového serveru (prednastavené: localhost)
- `DBPort` (číslo) číslo portu k prístupu k databáze (prednastavené: 3306)
- `DBName` (text) názov databázy (prednastavené: dipl)
- `DBUser` (text) login do databáze (prednastavené: root)
- `DBPass` (text) heslo do databáze (prednastavené: prázdna hodnota)
- `DistributionGroupMax` (číslo) maximálny počet povolených skupín pri diskretizácii spojitých atribútov
- `DistributionGroupSet` (číslo) počet diskrétnych skupín atribútov nastavený pri spúšťaní programu (prednastavené: 7)
- `TopKMin` (číslo) minimálny počet nastaviteľných top–k porovnávaných hodnôt (prednastavené: 10)
- `TopKMax` (číslo) maximálny počet nastaviteľných top–k porovnávaných hodnôt (prednastavené: 100)

- `TopKSet` (číslo) počet top–k porovnávaných hodnôt nastavený pri spúšťaní programu (prednastavené: 30)
- `DifferentRating` (číslo) maximálny počet rozdielnych hodnotení skrytým užívateľom počet (prednastavené: 10)
- `IterationStep` (číslo) počet krokov výpočtu, inak počet ohodnotených objektov k ukončeniu výpočtu (prednastavené: 100)

B.2 Grafické užívateľské rozhranie

Po spustení programu sa zobrazí užívateľské rozhranie (obrázok B.1). Toho rozhranie obsahuje všetky potrebné nastavenia k spusteniu výpočtu. Jednotlivé časti a ich funkčnosť je rozobraná v nasledujúcich podkapitolách.



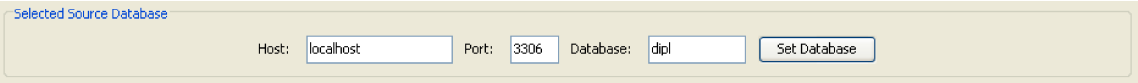
Obrázek B.1: Hlavný program.

Časť 1 obsahuje nastavenie zdroja dát (viď. kapitola B.2.1). Časti 2 a 3 obsahujú informácie o zdrojových dátach vo zvolenej databáze (viď. kapitola B.2.2). V častiach 4, 5 a 6 sú globálne nastavenia pre všetky modely (viď. kapitola B.2.3) a v častiach

7 a 8 sú nastavenia samotných modelov spolu s možnosťou spustenia výpočtu (viď. kapitola B.2.4).

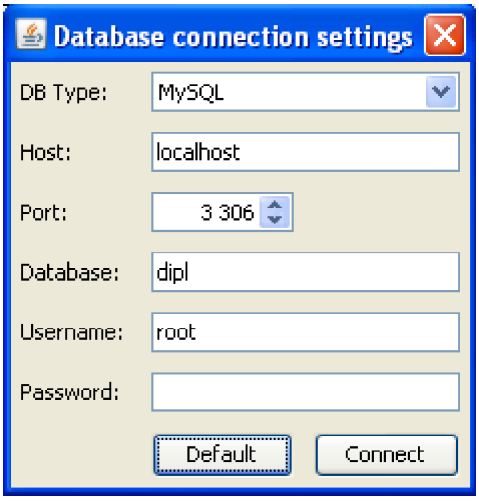
B.2.1 Nastavenie zdroja dát

V hlavnom okne programu časti 1 (obrázok B.2) je zobrazené aktuálne nastavenie databázového pripojenia a to Host, Port a názov databáze. K zmene nastavenia slúži tlačítko **Set Database**. Po jeho stlačení sa zobrazí okno (obrázok B.3), ktoré zmenu nastavenia umožňuje.



Obrázek B.2: Hlavný program - Nastavcná databáza.

Tlačítko **Connect** v okne pripojenia (obrázok B.3) slúži k pripojeniu podľa nastavených položiek. Tačítko **Default** nastaví jednotlivé položky podľa konfiguračného súboru `settings.properties` (viď. kapitola B.1.4).

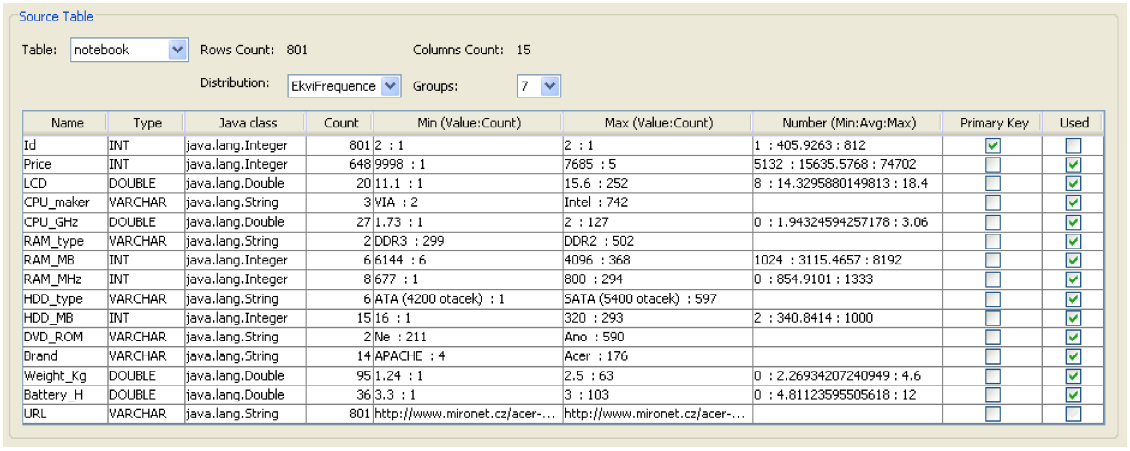


Obrázek B.3: Nastavenie zdroja dát.

B.2.2 Zdrojové dáta

Časť 2 hlavného programu (obrázok B.4) obahuje údaje o testovacích dátach. Voľba *Table* nastavuje tabuľku obsahujúcu testovacie dáta. Ďalším nastavením, obsiahnu-

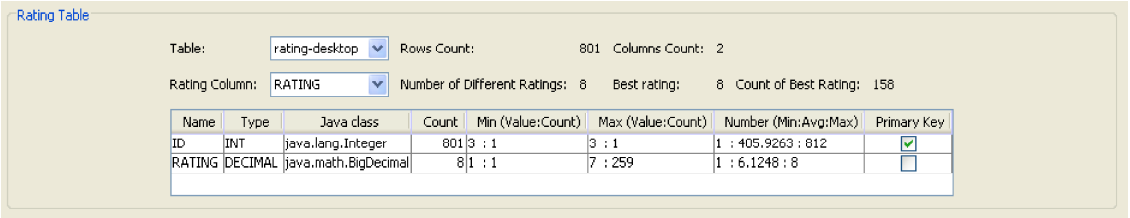
tým v tejto časti, je *Distribution*, ktoré udáva typ rozdelenia spojitých atribútov na diskrétne (ekvifrekvenčný alebo ekvidistančný typ rozdelenia). Počet novovzniknutých atribútov po rozdeleni spojitého atribútu sa nastavuje pomocou *Groups*. Rozdelenie spojitých atribútov je podrobne ukázané v kapitole 2.3.



Name	Type	Java class	Count	Min (Value:Count)	Max (Value:Count)	Number (Min:Avg:Max)	Primary Key	Used
Id	INT	java.lang.Integer	801	2 : 1	2 : 1	1 : 405.9263 : 812	☑	☐
Price	INT	java.lang.Integer	648	9998 : 1	7685 : 5	5132 : 15635.5768 : 74702	☐	☑
LCD	DOUBLE	java.lang.Double	20	11.1 : 1	15.6 : 252	8 : 14.3295880149813 : 18.4	☐	☑
CPU_maker	VARCHAR	java.lang.String	3	VIA : 2	Intel : 742		☐	☑
CPU_GHz	DOUBLE	java.lang.Double	27	1.73 : 1	2 : 127	0 : 1.94324594257178 : 3.06	☐	☑
RAM_type	VARCHAR	java.lang.String	2	DDR3 : 299	DDR2 : 502		☐	☑
RAM_MB	INT	java.lang.Integer	6	6144 : 6	4096 : 368	1024 : 3115.4657 : 8192	☐	☑
RAM_MHz	INT	java.lang.Integer	8	677 : 1	800 : 294	0 : 854.9101 : 1333	☐	☑
HDD_type	VARCHAR	java.lang.String	6	ATA (4200 otacek) : 1	SATA (5400 otacek) : 597		☐	☑
HDD_MB	INT	java.lang.Integer	15	16 : 1	320 : 293	2 : 340.8414 : 1000	☐	☑
DVD_ROM	VARCHAR	java.lang.String	2	Ne : 211	Ano : 590		☐	☑
Drand	VARCHAR	java.lang.String	14	APACHE : 4	Acer : 176		☐	☑
Weight_Kg	DOUBLE	java.lang.Double	95	1.24 : 1	2.5 : 63	0 : 2.26934207240949 : 4.6	☐	☑
Battery_H	DOUBLE	java.lang.Double	36	3.3 : 1	3 : 103	0 : 4.81123595505618 : 12	☐	☑
URL	VARCHAR	java.lang.String	801	http://www.mironet.cz/acer-...	http://www.mironet.cz/acer-...		☐	☐

Obrázek B.4: Hlavný program - Testovacie dáta.

Časť 3 hlavného programu (obrázok B.5) nastavuje skrytého užívateľa, ktorý má byť použitý k testovaniu. Výber tabuľky obsahujúcej hodnotenia všetkých objektov sa prevádza pomocou voľby *Table*. *Rating Column* určuje atribút, v ktorom sú obsiahnuté jednotlivé hodnotenia.



Name	Type	Java class	Count	Min (Value:Count)	Max (Value:Count)	Number (Min:Avg:Max)	Primary Key
ID	INT	java.lang.Integer	801	3 : 1	3 : 1	1 : 405.9263 : 812	☑
RATING	DECIMAL	java.math.BigDecimal	8	1 : 1	7 : 259	1 : 6.1248 : 8	☐

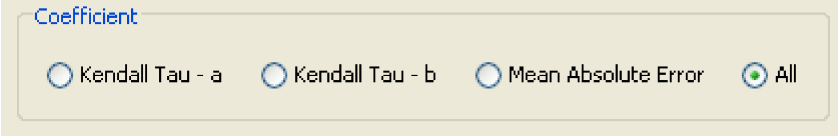
Obrázek B.5: Hlavný program - Skrytý užívateľ.

Zobrazované tabuľky v oboch častiach (2 aj 3) reprezentujú jednotlivé atribúty a ich vlasnosti. V rade za sebou sú to *Name* (názov atribútu), *Type* (typ v databázy), *Java class* (trieda v JAVA), *Count* (počet rozdielnych hodnôt atribútu), *Min (Value:Count)* (hodnota a počet najmenej sa vyskytujúceho atribútu), *Max (Value:Count)* (hodnota a počet najviac sa vyskytujúceho atribútu) a posledný informatívny *Number (Min:Avg:Max)* (najmenšia, priemerná a maximálna hodnota

číselného atribútu). Hodnota *Primary Key* je zaškrtnutá ak atribút je primárnym kľúčom. Tabuľka časti 2 naviac zahŕňa nastaviteľný stĺpec (posledný) *Used*. Pri jeho zaškrtnutí sa daný atribút bude brať do úvahy pri výpočte modela.

B.2.3 Globálne nastavenia

Časť 4 hlavného okna (obrázok B.6) obsahuje možnosti nastavenia podľa akého koeficientu majú byť hodnotené výsledky. Podrobne sa jednotlivým možnostiam venuje kapitola 4.4.

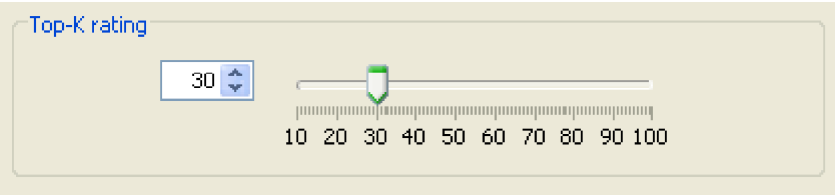


Obrázok B.6: Hlavný program - Kocficient.

Časť 5 hlavného okna (obrázok B.7) nastavuje možnosti výberu objektu určeného k ohodnoteniu. Tento objekt sa vyberá v každom kroku výpočtu. Jednotlivé možnosti su rozobrané v kapitole 4.5.

Obrázok B.7: Hlavný program - Výber objektu.

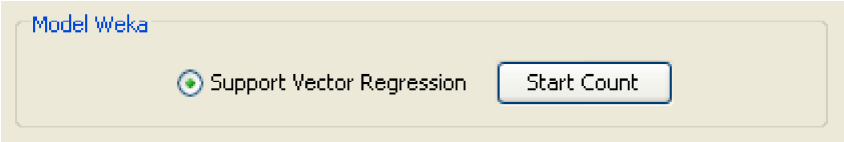
Nastavenie počtu top–k porovnávaných objektov je obsiahnuté v časti 6 (obrázok B.8). Zmenu rozpätia top–k možností (prednastavené od 10 do 100) je možné urobiť pomocou konfiguračného súboru (viď. kapitola B.1.4). Použitiu a významu hodnoty top k sa venuje kapitola 4.4.



Obrázek B.8: Hlavný program - Top-k.

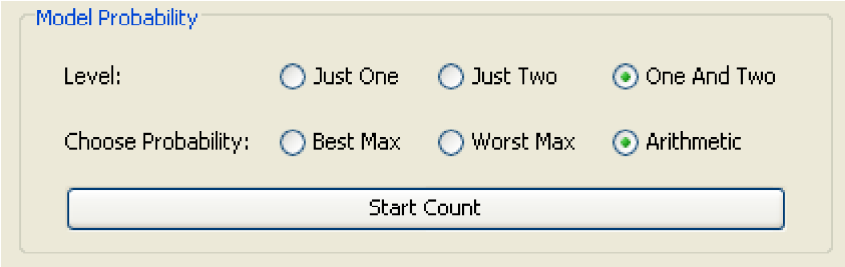
B.2.4 Modely rozhodovania

K spusteniu výpočtu slúži tlačítko **Start Count**, ktoré je obsiahnuté pri oboch testovaných modeloch. Možnosť výpočtu podľa modelu založeného na Support Vector Regression (viď. kapitola 2.5.4) je umožnená v časti 7 hlavného okna (obrázok B.9).



Obrázek B.9: Hlavný program - Weka.

Časť 8 hlavného okna (obrázok B.10) obsahuje možnosť výpočtu navrhnutého modelu. *Level* reprezentuje použitú úroveň tabuliek a nastavenie *Choose Probrability* určuje spôsob výpočtu, respektíve výberu pravdepodobnosti. Jednotlivé funkčnosti sú bližšie rozobrané v kapitole 4.6.

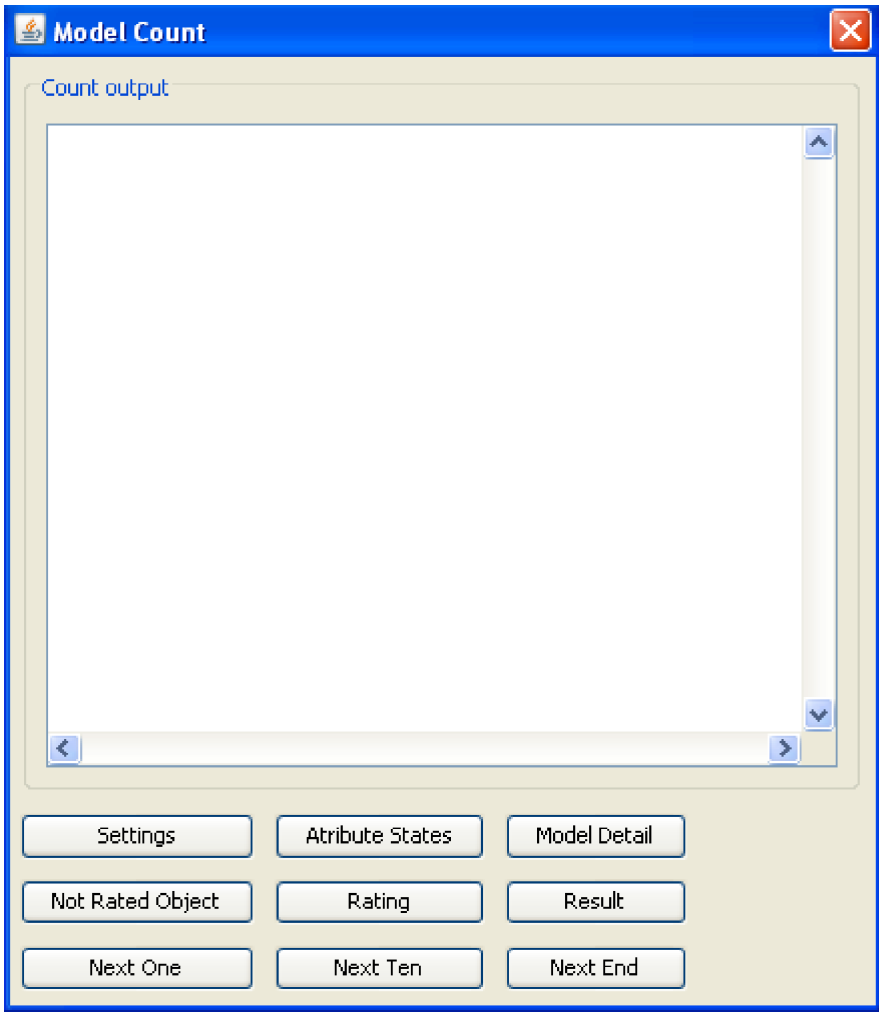


Obrázek B.10: Hlavný program - Model.

B.2.5 Zobrazovanie výsledkov

Tlačítkom **Start Count** sa zobrazí okno (obrázok B.11) určené k práci výpočtu modelu. Jednotlivé tlačítka v dolné časti okna (obrázok B.11) umožňujú kontrolu

nad celým výpočtom.



Obrázek B.11: Zobrazovanie výsledkov.

Settings tlačidlo zobrazuje nastavenia z hlavného okna.

Attribute States tlačidlo pre každý použitý atribút zobrazí jeho vlastnosti. Pre spojité atribúty zobrazuje intervaly vlastností po rozdelení.

Model Detail tlačidlo vypíše deail modelu v akuálnom kroku výpočtu.

Not Rated Object tlačidlo zobrazí všetky ešte neohodnotené objekty (objekty, ktorých hodnotenie ešte nebolo použité k tvorbe modelu).

Rating tlačidlo vypíše ID objektu a spolu s ohodnotením skrytého užívateľa a generovaným ohodnotením. Generované ohodnotenie vychadza z aktuálneho kroku výpočtu.

Result tlačidlo zobrazí zvolené hodnotenie výsledkov v danom kroku výpočtu.

Next One tlačidlo prevedie jeden krok výpočtu a objekt, spolu s ohodnotením a krokom výpočtu, pridá do výstupu.

Next Ten tlačidlo prevedie desať krokov výpočtu a objekty, spolu s ohodnotením a krokom výpočtu, pridá do výstupu.

Next End tlačidlo prevedie kroky výpočtu až do konečného počtu krokov a objekty, spolu s ohodnotením a krokmi, pridá do výstupu.

Pri zobrazovaní krokov výpočtu sa zobrazujú všetky doposiaľ ohodnotené objekty. Konečný počet krokov je daný konfiguračným súborom `settings.properties` (viď. kapitola B.1.4).