Určovanie koreferencie

Michal Novák

24. 4. 2009

Obsah

1 Zadanie

Mnohé slová (hlavne podstatné mená a zámená), ktoré sa nachádzajú v texte, sú pomenovania entít v reálnom svete.

> Pr. *Cesto, ktoré budeme miesiť je o čosi vláčnejšie a jemnejšie ako pri rožkoch. Na jeho prípravu treba...* [1]

V uvedenom príklade slovo „rožkoch“ ukazuje na entitu nejakých abstraktných rožkov, podobne slová „Cesto“ a „jeho“ ukazujú na nejaké cesto. Vzťah medzi týmito pomenovaniami a skutočným cestom sa nazýva **referencia**. Automaticky určovať reálne entity, na ktoré sa pomenovania odkazujú, je ale zatiaľ prakticky nerealizovateľné. Zmysel už má riešiť podúlohu, a to vyhľadávanie dvojíc slov, ktoré ukazujú na rovnakú entitu. V predchádzajúcom príklade sa nemusíme zaoberať tým, že slová „Cesto“ a „jeho“ ukazujú na nejaké cesto, dôležitý je fakt, že tieto pomenovania ukazujú na rovnakú entitu. Sú vo vzťahu **koreferencie**. Výraz uvedený v texte neskôr sa väčšinou označuje pojmom **anafora** (slovo „jeho“), výraz uvedený skôr pojmom **antecedent** (slovo „Cesto“) [2].

Vyriešenie problému automatického určovania koreferencií z textu je jednou z hlavných tém počítačovej lingvistiky a je dôležité pre iné aplikácie z tejto oblasti, napr. vyhľadávanie informácií, sledovanie obsahu dokumentu, strojový preklad a i. [2] Pre češtinu už bolo uskutočnených niekoľko experimentov, ktoré sú zdokumentované v prácach [3] a [4]. Tieto experimenty vychádzajú z metodiky a dát Pražského závislostného korpusu (PDT 2.0)[1] a boli uskutočnené pre rôzne druhy koreferencií (v závislosti na charaktere anafory). Práca [4] bola zameraná na určovanie koreferencií so zámennou anaforou, pričom bol dosiahnutý výsledok F-measure[2] 74,2%. V práci [3] bola úloha rozdelená na viac menších podproblémov, ale do experimentu boli zahrnuté aj nulové anafory a anafory vyjadrené doplnkom. Výsledkov je viacero, napr. pre osobnú zámennú anaforu bola dosiahnutá F-measure 75,8%, pre reflexívnu zámennú anaforu 97,1%, pre posesívnu zámennú anaforu 64,1% atď.[3]

Táto práca má za cieľ čo najlepšie vyriešiť úlohu určovania koreferencií (nezávisle na charaktere anafory), pričom pri experimentoch bude použitá podmnožina dát, ktoré boli vytvorené za účelom experimentov v [3] extrahovaním z PDT 2.0.

[1] http://ufal.mff.cuni.cz/pdt2.0/
[2] Spomenuté evaluačné metriky sú bližšie popísané v kapitole 4.2
[3] Hodnoty F-measure boli vypočítané z hodnôt Precision a Recall uvedených v [3]

2 Dáta

2.1 Veľkosť a typy dát

Dáta pre túto úlohu boli prevzaté z práce [3] a vznikli extrahovaním z PDT 2.0 a transformáciou pre účely experimentov v spomínanej práci. Sú vo formáte CSV[4] a pôvodne pozostávali z dvoch súborov: trénovacích dát (`train.csv`) a testovacích dát (`test.csv`). Za účelom vylepšovania naučených modelov som potreboval vývojové testovacie dáta. Tie som získal rozdelením súboru `test.csv` náhodne približne na dve polovice pomocou skriptu v jazyku Perl `divide-in-two-rand.pl`[5]. Tak vznikol súbor s vývojovými dátami (`dtest.csv`) a súbor s evaluačnými testovacími dátami (`etest.csv`). Evaluačné testovacie dáta slúžili na definitívne otestovanie úspešností modelov pre jednotlivé metódy, jeho výsledky som nepoužíval na ďalšie vylepšovanie modelov. Počty inštancií pre jednotlivé súbory sú v tabuľke 2.1.

Súbor dát		**Počet inštancií**		**Počet kladných klasifikácií**	**Podiel kladných inštancií**
train.csv		10001		1342	13,42%
test.scv	dtest.csv	3009	1494	189	12,65%
	etest.csv		1515	207	13,66%

Tabuľka 2.1 Počty inštancií a kladných klasifikácií (koreferujúcich dvojíc) pre jednotlivé typy dát

2.2 Atribúty

Každý riadok v dátovej tabuľke (inštancia) zodpovedá nejakej dvojici anafora a kandidát - antecedent. Tieto dvojice však nie sú tvorené všetkými dvojicami slov v zdrojových vetách. Princíp výberu spočíva v tom, že najprv sa vyberú anafory a potom sa ku každej anafore vyberajú kandidáti, ktorých býva väčšinou viac, čím vzniká v dátach vzťah M:N (anafora:kandidáti). Spôsoby výberu anafory a kandidátov sa líšia v závislosti na druhu hľadanej koreferencie, napr. pre koreferencie určené anaforami z osobných zámen sú kandidátmi sémantické substantíva z tej istej alebo predchádzajúcej vety, ako je anafora; pre koreferencie určené reflexívnymi zámennými anaforami sú kandidátmi efektívni potomkovia k anafore najbližšieho určitého slovesa alebo uzlu s funktorom DENOM. Viac informácií o výbere potenciálnych koreferenčných dvojíc je v práci [3]. Vzťah referencie medzi odkazovanou entitou a jej pomenovaním je z princípu 1:N. Po premietnutí tohto vzťahu do roviny koreferencie tak, že jedno slovo z koreferujúcej skupiny (antecedent) bude označené za hlavné pomenovanie a ostatné (anafory) budú naň odkazovať (inými slovami, viacero anafor odkazuje na jeden antecedent, ale nie naopak), je vidieť, že už len malá časť z vytvorenej dátovej tabuľky dvojíc anafora – kandidát na antecedent zodpovedá skutočným koreferenčným dvojiciam.

[4] http://cs.wikipedia.org/wiki/CSV
[5] Random seed inicializovaný číslom 1986

Z toho dôvodu je v dátach niekoľkonásobne väčšie zastúpenie nekoreferujúcich dvojíc, čo ukazuje aj tabuľka 2.1.

Inštancie v dátach sú charakterizované 54 atribútmi a klasifikačným ohodnotením. Atribúty sú kategoriálneho (50 atribútov) ako aj kontinuálneho charakteru (3 atribúty). Zostávajúci atribút `anaph_id` plní úlohu identifikátoru anafory, čiže ho nie je možné zaradiť do žiadnej z oboch skupín. Z významového hľadiska je možné atribúty rozdeliť podľa toho, z ktorej roviny anotácie bol daný atribút extrahovaný:

- slovná, textová rovina, napr. `cand_ord`, `sent_dist`, `sibl`
- morfologická rovina, napr. `cand_asubpos`, `cand_acase`, `anaph_apos`
- analytická rovina, napr. `cand_afun`, `subj_agree`
- tektogramatická rovina, napr. `gen_agree`, `cand_fun`, `anaph_tfa`

Keďže určovanie koreferencií je klasifikačná úloha, klasifikačné ohodnotenie je kategoriálneho charakteru, konkrétne sa jedná o binárne ohodnotenie (0 – nekoreferujúca dvojica / 1 – koreferujúca dvojica).

Charakter atribútov a možné hodnoty kategoriálnych typov sú popísané v súbore `all.names`. Tento súbor je využívaný pri načítavaní dát vo funkcii `load`.

3 Teoretická stránka riešenia

3.1 Rozhodovacie stromy

Táto metóda klasifikuje inštancie na základe rozhodovacieho stromu. Začína sa od koreňa stromu, v každom uzle sa otestuje podmienka a podľa jej výsledku sa pokračuje do jedného zo synovských uzlov. Podmienka v konkrétnom uzle je viazaná vždy na jeden klasifikačný atribút a rozhoduje sa podľa hodnoty, ktorú daný atribút testovanej inštancie nadobúda. V listoch stromu sú cieľové klasifikačné ohodnotenia. Rozhodovací strom je možno popísať aj ako disjunkciu konjunkcií hodnôt atribútov (kde konjunkciu reprezentuje cesta z koreňa do listu) [5]. Modifikáciou je pridanie číselných váh pre každý atribút a pri konštrukcii stromu počítanie s týmito váhami. Zlepšeniu úspešnosti rozhodovacieho stromu môže dopomôcť aj orezávanie jeho vetví.

3.2 Naivný Bayesov klasifikátor (Naive Bayes)

Bayesov klasifikátor priraďuje testovanej inštancii najpravdepodobnejšie cieľové ohodnotenie $y \in Y$ na základe hodnôt jej atribútov $a_1, \dots, a_n$. Z Bayesovho vzorca plynie:

$$y = argmax_{y_j \in Y} P(y_j | a_1, \dots, a_n) = argmax_{y_j \in Y} P(a_1, \dots, a_n | y_j) P(y_j)$$

Naivný Bayesov klasifikátor pracuje na základe zjednodušeného predpokladu, že hodnoty atribútov sú podmienene nezávislé za predpokladu daného cieľového ohodnotenia [6]. Preto platí:

$$y = argmax_{y_j \in Y} \prod_{i=1}^{n} P(a_i | y_j) \, P(y_j)$$

Tento klasifikátor teda funguje na princípe výpočtu cieľového ohodnotenia z aposteriórnych pravdepodobností hodnôt atribútov a apriórnych pravdepodobností cieľových klasifikačných ohodnotení, ktoré boli odhadnuté z testovacích dát. Aposteriórne pravdepodobnosti môžu byť navyše modifikované tzv. Laplaceovým vyhladzovaním[6], kde sa pri výpočte pravdepodobnosti berie do úvahy aj počet možných hodnôt atribútu.

3.3 Support Vector Machines (SVM)

Model metódy SVM vznikne optimálnym rozdelením inštancií v trénovacích dátach na dve časti lineárnym separátorom (nadrovinou) podľa hodnoty cieľovej klasifikácie. Jednotlivé súradnice inštancií v rozdeľovanom priestore sú definované hodnotami ich atribútov. Z toho dôvodu musia byť všetky atribúty kontinuálneho charakteru. Úloha nájdenia optimálneho separátora je úlohou kvadratického programovania, čiže nájsť také α_i, ktoré maximalizujú nasledujúci výraz:

[6] http://en.wikipedia.org/wiki/Additive_smoothing

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

pričom platí obmedzenie

$$\alpha_i \geq 0, \sum_i \alpha_i y_i = 0$$

kde $\boldsymbol{x_i}$ je inštancia definovaná svojimi atribútmi a y_i je cieľové ohodnotenie inštancie [7]. Namiesto skalárneho súčinu $\boldsymbol{x_i} \cdot \boldsymbol{x_j}$ je pri neexistencii lineárneho separátora možné použiť jadrovú transformáciu priestoru inštancií do priestoru vyššej dimenzie. Tým sa prevedie pôvodne lineárne neseparovateľná úloha na lineárne separovateľnú [8]. Ak aj napriek tomu nie sú dáta oddeliteľné, do vzťahu pre nájdenie separátoru pribudne penalizácia za náležanie do nesprávnej časti.

4 Praktická stránka riešenia

4.1 Implementácia

Na počiatočné rozdelenie trénovacích dát slúži nástroj naprogramovaný v jazyku Perl `divide-in-two-rand.pl`. Zvyšok úlohy bol riešený v štatistickom programe R. V súbore `koref.r` sa nachádzajú všetky potrebné funkcie na načítanie dát, ich transformáciu, selekciu atribútov, trénovanie modelov, klasifikáciu testovacích dát a vyhodnocovanie výsledkov. Na konkrétne metódy strojového učenia som použil tieto funkcie:

- Rozhodovacie stromy – funkcia `rpart` z knižnice `rpart`
- Naive Bayes – funkcia `naiveBayes` z knižnice `e1071`
- SVM – funkcia `svm` z knižnice `e1071`

Okrem spomínaných knižníc funkcie vyžadujú prítomnosť knižnice `cba` (obsahuje funkciu `as.dummy`, ktorá sa používa na prevod kategoriálnych atribútov na kontinuálne) a `proxy` (knižnica `cba` je na nej závislá).

4.2 Hodnotenie úspešnosti

Pri výbere množiny atribútov a pri krokoch, ktorými som chcel zlepšiť kvalitu výsledných modelov, som potreboval mieru, pomocou ktorej by som túto kvalitu kvantifikoval. Použil som tzv. **F-measure**, ktorá je definovaná ako

$$F = \frac{2PR}{P + R}$$

kde

$$P = \frac{\#\ správne\ klasifikovaných\ kladne}{\#\ správne\ klasifikovaných\ kladne + \ \#\ nesprávne\ klasifikovaných\ kladne}$$

$$R = \frac{\#\ správne\ klasifikovaných\ kladne}{\#\ správne\ klasifikovaných\ kladne + \ \#\ nesprávne\ klasifikovaných\ záporne}$$

Hodnoty P a R sú miery **precision** (presnosť) a **recall** (úplnosť). Vyžadujú, aby cieľová klasifikácia bola binárna, čiže inštancia môže byť ohodnotená kladne alebo záporne.

V nasledujúcom texte okrem pojmu F-measure používam slovo úspešnosť. Týmto termínom myslím takisto hodnotu F-measure[7]. Rovnako sa v ďalšom texte predpokladá, že všetky testy úspešností, pri ktorých nie je určené, na akom type dát boli vykonávané, prebiehali na vývojových testovacích dátach. Testy na trénovacích a evaluačných testovacích dátach sú explicitne zdôraznené.

[7] nie je tým myslená úspešnosť (accuracy) vyjadrená ako pomer správne klasifikovaných inštancií a všetkých inštancií

5 Analýza riešenia

Pôvodné dáta obsahovali 54 atribútov a binárne klasifikačné ohodnotenie, či je dvojica anafora a kandidát - antecedent koreferenčná. Takéto množstvo atribútov ponúka veľký priestor na modifikáciu pôvodných dát za účelom zlepšenia úspešnosti klasifikácie.

Základná modifikácia dát, ktorú som spravil pri všetkých ďalších testoch, je odstránenie prvého atribútu, a to identifikátoru anafory (`anaph_id`). Použitie tohto atribútu pri trénovaní modelov nezlepšuje alebo dokonca zhoršuje úspešnosť klasifikácie a podstatne predlžuje výpočet. Z toho dôvodu sa v ďalšom texte tento atribút neberie vôbec do úvahy a termínom **pôvodné dáta** budem označovať dáta upravené touto základnou modifikáciou, podobne **kompletná sada atribútov** tento atribút neobsahuje.

Pre metódu SVM je potrebné, aby všetky atribúty, z ktorých sa trénuje model, boli kontinuálneho charakteru. Je nutné pretransformovať kategoriálne atribúty. Na to som naprogramoval funkciu `transform_to_continuous`, ktorá na transformáciu interne používa funkciu `as.dummy` z knižnice `cba`. Z každého kategoriálneho atribútu vznikne toľko nových atribútov, koľko hodnôt môže pôvodný atribút nadobúdať. Každý z týchto nových atribútov už nadobúda iba hodnoty 0 a 1, je možné s ním pracovať ako s kontinuálnym. V nasledujúcich experimentoch pri použití metódy SVM bola vždy ako posledný krok pred trénovaním modelu uskutočnená transformácia dátových súborov na čisto kontinuálne.

Všetky experimenty popísané po kapitolu 6 boli robené s implicitnými hodnotami parametrov pre jednotlivé metódy strojového učenia. Bližšie informácie o parametroch je možné získať v R dokumentácii k jednotlivým metódam.

Výsledky všetkých testov počas hľadania optimálnej množiny atribútov a modifikácie ich hodnôt sú v tabuľke 5.3.

5.1 Úvodný test

Na začiatku som uskutočnil úvodný test úspešnosti klasifikácie za použitia pôvodných dát. Tento test mi stanovil hodnoty, s ktorými som mohol porovnávať výsledky ďalších testov a sledovať, či sa natrénované modely skvalitňujú alebo nie. Úspešnosti primárneho testu je vidieť v tabuľke 5.3, riadok 1. Znepokojujúca je najmä výrazne malá hodnota F-measure klasifikácie pri použití naivného Bayesovského klasifikátoru, čo ukazuje na cieľ nasledujúcich modifikácii – okrem celkového zlepšenia, zvýšiť úspešnosť Bayesovského klasifikátoru bližšie k hodnotám úspešností zostávajúcich dvoch metód.

Pri pohľade na frekvencie hodnôt kategoriálnych atribútov v trénovacích dátach som odpozoroval dva dôležité problémy:

1. niektoré atribúty majú veľmi veľa možných hodnôt, čo zhoršuje rozhodovanie pri klasifikácii
2. početnosť niektorých hodnôt atribútov je veľmi malá, štatisticky nevýznamná

5.2 Automatická modifikácia dát

Rozhodol som sa zoskupiť príbuzné hodnoty atribútov (riešenie problému 1 v časti 5.1) a odstrániť jednotvárne atribúty, teda také, kde jedna hodnota výrazne prevažuje nad ostatnými (riešenie problému 2). Moja požiadavka bola, aby tieto modifikácie prebehli čo najviac automaticky. Za týmto účelom som naprogramoval funkcie `get_auto_group_model` a `get_monotone_attribs`. Spustenie funkcií v tomto poradí vytvorí schému, pomocou ktorej je potom možné rovnakým spôsobom modifikovať jednotlivé dátové súbory.

5.2.1 Automatické zoskupenie hodnôt

Navrhnutým riešením problému 1 v časti 5.1 je funkcia `get_auto_group_model`. Tá prechádza postupne všetky kategoriálne atribúty a zlučuje ich hodnoty do skupín. Toto zoskupovanie je založené na podobnosti podmienených pravdepodobností, že pri danej hodnote v_k atribútu x je zodpovedajúca inštancia klasifikovaná ako koreferenčná ($y = 1$). Takto vznikne zoznam pravdepodobností (pre každú hodnotu atribútu jedna pravdepodobnosť), hodnoty sa podľa nich vzostupne usporiadajú a postupne sa zhlukujú do skupín G_j tak, že vzdialenosť aritmetického priemeru pravdepodobností pre hodnoty patriacej do rovnakej skupiny a hodnoty najmenšej pravdepodobnosti patriacej do nasledujúcej skupiny je rovná alebo väčšia voliteľnému parametru $\varepsilon \in \langle 0,1 \rangle$. Zapísané formálne:

1. $v_1 \in G_1$
2. $v_{i+1} \in G_j \quad ak \quad v_i \in G_j \ \wedge \ P(y = 1|x = v_{i+1}) - \frac{\sum_{v_k \in G_j} P(y=1|x=v_k)}{|G_j|} < \varepsilon$
   $v_{i+1} \in G_{j+1} \quad inak$

Čím je parameter ε vyšší, tým je počet nových hodnôt menší a zahrňuje v sebe viac pôvodných hodnôt atribútu. Nulový ε znamená zachovanie pôvodných hodnôt, teda žiadne zoskupovanie.

> Pr. *Atribút X nadobúda hodnoty U, V a W. Y je klasifikácia inštancie. Dáta majú 100 inštancií a ich distribúcia je popísaná v tabuľke 5.1. V tabuľke je vypočítaná aj požadovaná podmienená pravdepodobnosť, podľa ktorej sa atribúty vzostupne zoradia do postupnosti U, W a V.*

v	Počet inštancií		$P(Y = 1\|X = v)$
	Celkom	$Y = 1$	
U	60	10	0,1666
V	5	4	0,8
W	35	7	0,2

Tabuľka 5.1 Príklad distribúcie klasifikácie *Y* voči hodnotám atribútu *X*

Pre parameter $\varepsilon = 0{,}04$ algoritmus postupuje nasledovne:

1. *U: vytvorí sa nová skupina GROUP_U, a inicializuje sa hodnotou U*
2. *W: vzdialenosť pravdepodobnosti pre hodnotu W (0,2) od aritmetického priemeru pravdepodobností v GROUP_U (0,1666; obsahuje iba hodnotu U) je menšia ako parameter ε, takže do GROUP_U sa vloží aj hodnota W*
3. *V: vzdialenosť pravdepodobnosti pre hodnotu V (0,8) od aritmetického priemeru pravdepodobností v GROUP_U (0,1833) je väčšia ako parameter ε, takže sa vytvorí nová skupina GROUP_V, do ktorej sa vloží hodnota V*
4. *Skupiny GROUP_U a GROUP_V sú nové hodnoty atribútu X, viď. tabuľka 5.2*

v	Počet inštancií		$P(X = v)$	$P(Y = 1 \mid X = v)$
	Celkom	$Y = 1$		
GROUP_U	95	17	0,95	0,1666
GROUP_V	5	4	0,05	0,8

Tabuľka 5.2 Distribúcia hodnôt atribútu *X* a klasifikácie *Y* voči hodnotám atribútu *X*

5.2.2 Odstránenie jednotvárnych atribútov

Ako riešenie problému 2 v časti 5.1 som navrhol funkciu `get_monotone_attribs`, ktorá prechádza cez všetky kategoriálne atribúty a určuje tie, ktoré sa majú vyhodiť. Konkrétne vyberie tie atribúty, u ktorých frekvencia jednej hodnoty prevyšuje $(1 - \theta)$, kde θ je opäť voliteľný parameter. Vyšší θ spôsobí odstránenie viacerých atribútov. Nulová hodnota parametru θ ponecháva všetky pôvodné atribúty, teda aj tie, ktoré po zoskupení hodnôt môžu nadobúdať iba jednu hodnotu.

> Pr. *Atribút X nadobúda hodnoty GROUP _U a GROUP _V podľa distribúcie uvedenej v tabuľke 5.2. Pre $\theta = 0{,}06$ bude atribút zo sady odstránený, pretože frekvencia práve jednej jeho hodnoty (0,95; hodnota GROUP_U) prevyšuje hodnotu $1 - \theta$ (0,94).*

Na predchádzajúcom príklade je vidieť, že ak by bola volaná funkcia `get_monotone_attribs` ako prvá, čiže distribúcia hodnôt by zodpovedala tabuľke 5.1, atribút by nebol odstránený. Z toho dôvodu je dôležité zachovanie poradia volania popísaných funkcií.

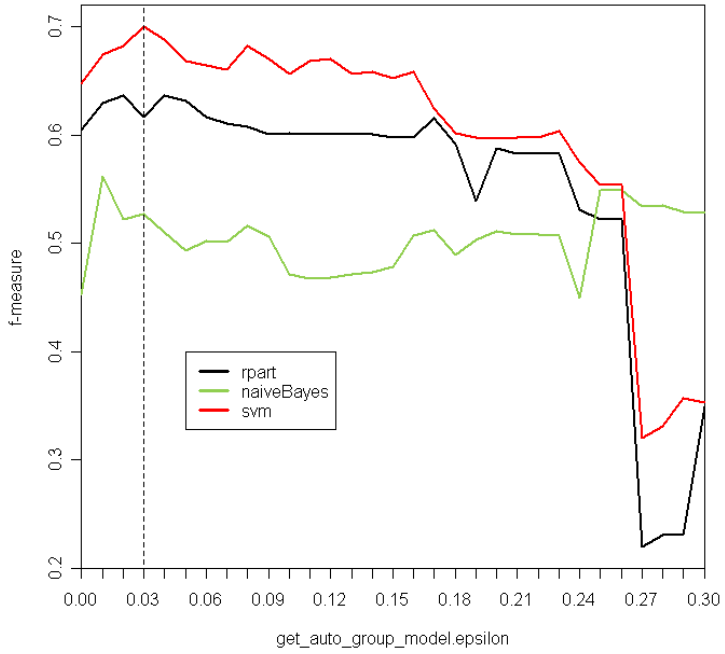
5.2.3 Výsledky použitého postupu

Práve nastavenie týchto dvoch parametrov ε a θ určuje spôsob modifikácie dát, zvyšok prebieha automaticky. Testoval som úspešnosť metód strojového učenia pri použití takto modifikovaných dát pre rôzne hodnoty týchto parametrov. Zistil som, že rozumná zmena ($< 0{,}3$) parametru θ funkcie `get_monotone_attribs` nemá veľký vplyv na úspešnosť klasifikácie. Vyššie hodnoty tohto parametru vypustia príliš mnoho atribútov, čo spôsobí razantný pokles úspešnosti. Výraznejšie zmeny som zaznamenal pri modifikácii parametru ε (v rozmedzí $\langle 0; 0{,}3 \rangle$) funkcie `get_auto_group_model`, ako ukazuje obrázok 5.1.

5.2.4 Poznatky z výsledkov použitého postupu

Z týchto výsledkov môžem vyvodiť tieto závery pre popísanú automatickú modifikáciu dát:

- pri vhodnej voľbe parametrov sa úspešnosť zlepší, nie však nejako dramaticky
- klasifikácia metódou SVM dosahuje veľmi dobré výsledky, najlepšie pre dvojicu parametrov ($\varepsilon = 0{,}03$; $\theta = 0{,}01$), a to 70,06% (tabuľka 5.3, riadok 2). Pri tomto modeli sa používa 37 atribútov, z toho 20 má hodnoty zmenené zoskupovaním, viď. príloha 1
- naivný Bayesov klasifikátor si výrazne polepšil, ale stále nedosahuje dostatočné úspešnosti, ktoré by sa blížili úspešnostiam zvyšných dvoch metód
- na dobré výsledky je potrebné veľké množstvo atribútov (najlepšia klasifikácia SVM potrebuje 37 atribútov)



Obrázok 5.1 Hodnoty F-measure pri testovaní modelov vytvorených z automaticky modifikovaných dát v závislosti na metóde a parametri ε funkcie `get_monotone_attribs` (najlepší výsledok pre $\varepsilon = 0,03$)

5.3 Automatický výber atribútov

Stále pretrvávajúce problémy ma donútili zmeniť postup. Problémom predchádzajúcich riešení je, že pri klasifikácii sa riadia príliš veľkým množstvom atribútov. Prítomnosť niektorých atribútov nemusí mať vplyv na úspešnosť klasifikácie, prípadne ju môže aj zhoršovať, ak spôsobia tzv. pretrénovanie. Z tohto dôvodu je nutné obmedziť počet atribútov a pri trénovaní modelu použiť iba tie skutočne relevantné.

5.3.1 Popis algoritmu

Na riešenie tejto úlohy som naprogramoval funkciu – `choose_best_attribs`. Princíp jej fungovania spočíva v hladovom algoritme. Na začiatku vyberie všetky

kombinácie dvojíc atribútov, čiže $\binom{53}{2} = 1378$ atribútov. Pre každý výber natrénuje zadanou metódou model a otestuje na vývojových testovacích dátach. Do ďalšej iterácie postúpia kombinácie s najvyššou hodnotou úspešnosti a ku každej dvojici sa postupne pridáva tretí zo zostávajúcich 51 atribútov. Najlepšie trojice sú vstupom do ďalšej iterácie atď. Pokračuje sa buď do parametrom stanoveného maximálneho počtu atribútov alebo dokiaľ ešte pridanie ďalšieho atribútu zlepší úspešnosť klasifikácie. Výsledkom volania funkcie je zoznam atribútov, ktoré pre zadanú metódu, trénovacie a testovacie dáta dávajú pravdepodobne najlepšie výsledky. Slovo pravdepodobne preto, lebo táto funkcia neprechádza všetky kombinácie atribútov, takže teoreticky môže existovať nevyskúšaná kombinácia, ktorá vracia lepšie výsledky ako kombinácia vybraná touto funkciou. Pravdepodobnosť optimálneho výsledku by sa zvýšila, ak by sa nevyberali iba tie n-tice s najlepším výsledkom, ale aj s druhým, tretím atď. najlepším výsledkom, na druhej strane by sa tým však enormne predlžil výpočet. Z tohto dôvodu funkcia `choose_best_attribs` v jednotlivých iteráciách vyberá iba tie najlepšie n-tice.

5.3.2 Výsledky na pôvodných dátach výberom metódou `rpart`

Vychádzal som opäť z pôvodných dát. Funkcia `choose_best_attribs` vybrala metódou rozhodovacích stromov tieto atribúty (*AttrRpart1*):

`gen_agree, num_agree, cand_fun, cand_afun, cand_asubpos, cand_acase, sent_dist, cand_ord`.

Neprekvapilo, že najlepšie dopadla metóda `rpart`, na ktorej bola množina atribútov *AttrRpart1* vyberaná, ostatné dve metódy zaznamenali zhoršenie, viď. tabuľka 5.3, riadok 3.

5.3.3 Výsledky na modifikovaných dátach výberom metódou `rpart`

Vyskúšal som natrénovať modely automaticky modifikovanými dátami s parametrami $\varepsilon = 0{,}03$ a $\theta = 0{,}01$, pričom som použil iba vyššie spomenuté atribúty (automatická modifikácia ani jeden z týchto parametrov neodstránila). Výsledky tejto varianty sú v tabuľke 5.3, riadok 4.

Takisto som skúsil použiť iba atribúty, ktoré boli výsledkom volania funkcie `choose_best_attribs` na modifikovaných dátach, a to konkrétne (*AttrRpart2*):

`gen_agree, num_agree, cand_afun, cand_apos, file_deepord_dist, sibl`.

Modely som natrénoval tiež na modifikovaných dátach s rovnakými parametrami ε a θ ako v predchádzajúcom prípade (tabuľka 5.3, riadok 5). Oba spôsoby priniesli zlepšenie u metód Naive Bayes a SVM, u rozhodovacích stromov sa naopak úspešnosť zhoršila.

5.3.4 Výsledky na pôvodných dátach výberom metódou Naive Bayes

Kvôli posledne spomenutému problému z predchádzajúceho odseku som opäť nechal vybrať optimálnu množinu atribútov funkciou `choose_best_attribs`,

tentoraz však pomocou metódy Naive Bayes. Výsledkom bolo týchto 11 atribútov (*AttrBayes1*):

```
gen_agree, num_agree, app_in_coord, cand_epar_sempos,
anaph_epar_sempos, fun_agree, cand_agen, clause_dist,
tfa_agree, sibl.
```

Testovanie úspešnosti klasifikácie na základe tohto výberu atribútov prinieslo doteraz najlepšie výsledky pre metódu Naive Bayes. Avšak pre zvyšné dve metódy boli tieto výsledky doteraz najhoršie (tabuľka 5.3, riadok 6).

5.3.5 Poznatky z výsledkov použitých postupov

Nadobudnuté poznatky z tohto postupu a z neho plynúcich výsledkov boli nasledovné:

- bol dosiahnutý zatiaľ najlepší výsledok u metódy Naive Bayes, ktorej úspešnosť sa dostala nad hranicu 60%
- očakávane sa zlepšila úspešnosť pomocou metódy `rpart`
- naopak, od výberu atribútov na modifikovaných dátach (*AttrRpart2*) som očakával zlepšenie úspešnosti pri použití rozhodovacích stromov; nastala opačná situácia, zvyšné dve metódy však zlepšenie zaznamenali
- úspešnosti boli oproti pokusom s automaticky modifikovanými dátami na kompletnej sade atribútov stále nízke

Výbery atribútov pomocou funkcie `choose_best_attribs` ukazujú na jeden dôležitý poznatok:

- všetky výbery obsahujú atribúty `gen_agree` (zhoda v sémantickom rode) a `num_agree` (zhoda v sémantickom čísle)

Z lingvistického hľadiska sú tieto dva atribúty naozaj veľmi dôležité pri určovaní koreferencie, navyše sa jedná o binárne atribúty, takže rozhodovanie pri nich je najjednoduchšie možné. O dôležitosti týchto atribútov svedčí aj ich použitie hneď v prvej fáze algoritmu popísanom v práci [4].

5.3.6 Ďalší možný postup

Automatický výber atribútov použitím príslušnej metódy vždy zlepšil dovtedajšie výsledky. Podobným postupom by bolo možné vybrať aj najlepšiu množinu atribútov pre metódu SVM. Úspešnosť takto natrénovaného modelu by možno prevýšila doteraz najlepší výsledok na automaticky modifikovaných pôvodných dátach (70,06%). Túto variantu som skúšal, ale vzápätí som ju aj opustil, pretože trénovanie SVM modelov trvá výrazne dlhšie ako pri ostatných dvoch metódach, a tým sa celý algoritmus stáva veľmi časovo náročný.

Ďalšou možnosťou je vyskúšať výber atribútov funkciou `choose_best_attribs` za použitia automaticky modifikovaných dát. Tento postup som aplikoval pomocou metódy rozhodovacích stromov (kapitola 5.3.3), čo zlepšilo výsledky pre zvyšné dve metódy (pre rozhodovacie stromy nastalo zhoršenie), ale napriek tomu boli tieto

výsledky horšie ako výsledky aktuálnych optimálnych riešení. Navyše, ako je spomenuté v predchádzajúcom odseku, z dôvodov časových nárokov som automatický výber atribútov pomocou metódy SVM neskúšal ani len na pôvodných dátach. To sú dôvody, prečo som túto variantu opustil a vydal sa iným smerom.

5.4 Manuálna modifikácia dát

Zlučovanie hodnôt atribútov automatickou metódou (kapitola 5.2) prinieslo doteraz najlepšiu úspešnosť. Avšak metodika zlučovania nemusela byť tá najsprávnejšia, rozhodol som sa preto zvoliť inú, ktorá by eventuálne mohla priniesť lepšie výsledky. Ďalej popísaný postup zlučovania hodnôt je založený na štruktúre rozhodovacích stromov, konkrétne rozhodovacieho stromu natrénovaného z atribútov *AttrRpart1* (jeho textový popis je na obrázku 5.2).

5.4.1 Odstránenie nepoužitého atribútu

Zo štruktúry rozhodovacieho stromu na obrázku 5.2 vidieť, že pri jeho konštrukcii nebol použitý vôbec atribút `sent_dist`. Môžem ho preto odstrániť. Nazvem túto novú množinu atribútov *AttrRpart3*. Úspešnosť rozhodovacích stromov ostala nezmenená, úspešnosť zvyšných metód sa nepatrne zvýšila (tabuľka 5.3, riadok 7).

5.4.2 Zlučovanie hodnôt na základe štruktúry rozhodovacieho stromu

Z obrázku 5.2 je možno pozorovať, že v niektorých uzloch stromu je rozhodovanie založené na disjunkcii viacerých hodnôt nejakého atribútu. Takéto hodnoty sú vhodným kandidátom na zlúčenie do jednej hodnoty. Je však nutné si dávať pozor, pretože upravovaný atribút môže byť použitý aj na inom mieste rozhodovacieho stromu. Na tomto mieste môže byť rozhodovanie založené na inej podmnožine hodnôt alebo môže tvoriť s predchádzajúcou množinou hodnôt neprázdny prienik.

```
1) root 10001 1342 0 (0.86581342 0.13418658)
    2) gen_agree=0 6529  149 0 (0.97717874 0.02282126) *
    3) gen_agree=1 3472 1193 0 (0.65639401 0.34360599)
      6) cand_afun=Adv,Atr,AtrAdv,AtrAtr,AtrObj,Atv,AtvV,AuxP,ExD,Obj,Pnom 2235  482 0 (0.78434004 0.21565996)
       12)  cand_fun=ACMP,AIM,CAUS,CNCS,COMPL,COND,CPHR,CPR,CRIT,DIFF,DIR1,DIR2,EXT,ID,MANN,MAT,MEANS,PAR,REG,RESTR,
            RSTR,SUBS,TFRWH,THL,THO,TOWH,TPAR,TSIN,TTILL,TWHEN 760   26 0 (0.96578947 0.03421053) *
       13) cand_fun=ACT,ADDR,APP,AUTH,BEN,DENOM,DIR3,EFF,LOC,ORIG,PAT 1475  456 0 (0.69084746 0.30915254)
         26) num_agree=0 361   15 0 (0.95844875 0.04155125) *
         27) num_agree=1 1114  441 0 (0.60412926 0.39587074)
           54) cand_ord>=5.5 436   94 0 (0.78440367 0.21559633) *
           55) cand_ord< 5.5 678  331 1 (0.48820059 0.51179941)
            110) cand_asubpos=7,8,9,A,D,K,N,U,Z 573  260 0 (0.54624782 0.45375218)
              220) cand_acase=1,6,7,X 164   46 0 (0.71951220 0.28048780) *
              221) cand_acase=2,3,4 409  195 1 (0.47677262 0.52322738)
                442) cand_ord>=3.5 94   33 0 (0.64893617 0.35106383) *
                443) cand_ord< 3.5 315  134 1 (0.42539683 0.57460317)
                  886) cand_fun=APP,AUTH 72   27 0 (0.62500000 0.37500000) *
                  887) cand_fun=ACT,ADDR,BEN,DIR3,EFF,LOC,ORIG,PAT 243   89 1 (0.36625514 0.63374486) *
            111) cand_asubpos=1,4,5,6,H,l,n,P,S 105   18 1 (0.17142857 0.82857143) *
      7) cand_afun=AdvAtr,AuxT,empty,Sb 1237  526 1 (0.42522231 0.57477769)
       14) num_agree=0 189   15 0 (0.92063492 0.07936508) *
       15) num_agree=1 1048  352 1 (0.33587786 0.66412214)
         30) cand_ord>=8.5 255  107 0 (0.58039216 0.41960784) *
         31) cand_ord< 8.5 793  204 1 (0.25725095 0.74274905) *
```

Obrázok 5.2 Textový popis rozhodovacieho stromu natrénovaného z množiny atribútov *AttrRpart1*

Toto manuálne zlúčenie hodnôt do skupín a výber atribútov patriacich do množiny *AttrRpart3* zabezpečuje funkcia `rpart_best_data_transformation`. V nej sa zoskupujú hodnoty kategoriálnych atribútov `cand_fun`, `cand_afun`, `cand_asubpos`, `cand_acase` a kontinuálny atribút `cand_ord` sa prevádza na kategorický na základe deliacich hodnôt z rozhodovacieho stromu na obrázku 5.2.

Modely postavené na takto manuálne modifikovaných dátach a množine atribútov *AttrRpart3* dosiahli veľmi dobrú a hlavne relatívne vyrovnanú úspešnosť (tabuľka 5.3, riadok 8). Pre rozhodovací strom bol potvrdený zatiaľ najlepší výsledok. Metóda Naive Bayes sa oproti výsledku na nemodifikovaných dátach výrazne zlepšila, prekročila métu 60%, hoci najlepší výsledok u tejto metódy (s množinou atribútov *AttrBayes1*) nebol prekonaný. Metóda SVM síce nedosiahla maximálnu úspešnosť, ale aj tak klasifikuje lepšie v porovnaní s modelom natrénovaným na pôvodných dátach (s 53 atribútmi).

5.4.3 Včlenenie niektorých zvyšných hodnôt do skupín

Niektoré atribúty nadobúdajú v inštanciách hodnoty, ktoré sa na výstavbe rozhodovacieho stromu nepodieľajú, takže sa v predchádzajúcom kroku nevčlenili do žiadnej skupiny. Pridruženie niektorých z týchto samostatných hodnôt do existujúcich skupín by teoreticky mohlo vylepšiť úspešnosť.

V tomto prípade sa jednalo o atribúty `cand_acase` a `cand_asubpos`, ktoré s nezanedbateľnou frekvenciou nadobúdali ďalšie hodnoty nezaradené do vytvorených skupín. V prípade prvého atribútu akékoľvek pričlenenie samostatných hodnôt do existujúcich skupín nezlepšilo úspešnosť. V druhom prípade pridruženie dvoch samostatných hodnôt do tej istej skupiny úspešnosť zas o čosi vylepšilo (tabuľka 5.3, riadok 9). Pravidlá zoskupenia sú popísané v prílohe 2.

5.4.4 Automatický výber atribútov na manuálne modifikovaných dátach

Zlúčenie hodnôt atribútov na základe štruktúry rozhodovacieho stromu výrazne pomohlo. Dalo by sa uvažovať, že týmto spôsobom je možné vytvoriť ešte lepší model. Stačilo by na takto manuálne modifikovaných kompletných dátach (53 atribútov) spustiť funkciu `choose_best_attribs` za použitia metódy rozhodovacích stromov a z výslednej množiny atribútov postaviť rozhodovací strom. V ňom by sa dalo opäť rozumne manuálne zoskupiť hodnoty niektorých atribútov. Podobne som aj postupoval. Keďže by mohla funkcia `choose_best_attribs` vybrať množinu atribútov takú, že z nej postavený rozhodovací strom by mal horšiu úspešnosť ako pôvodný model (pretože táto funkcia nemusí nutne vybrať optimálne atribúty), umožnil som vo funkcii `choose_best_attribs` určenie počiatočnej množiny atribútov. K tejto množine algoritmus ďalej pridáva ďalšie atribúty pokiaľ sa úspešnosť výsledného modelu zlepšuje. Výsledkom bolo rozšírenie počiatočnej množiny *AttrRpart3* o atribúty `epar_fun_agree` a `cand_subj` (*AttrRpart4*). Oba nové atribúty sú binárne, preto nie je potrebné zoskupovať hodnoty.

Pridanie týchto atribútov ešte o trochu vylepšilo výsledky metódy `rpart`. Naopak, metóda SVM zaznamenala oproti predchádzajúcemu experimentu nepatrný pokles a najviac sa táto zmena prejavila na metóde Naive Bayes, u ktorej nastalo rapídne zníženie úspešnosti, viď tabuľka 5.3, riadok 10.

5.5 Výsledné modifikácie

Zo všetkých modifikácií som sa rozhodol pre každú metódu strojového učenia vybrať na ďalšie experimenty tú, pri ktorej použití dosiahla príslušná metóda

najlepšiu úspešnosť. U metódy rozhodovacích stromov je to manuálna modifikácia dát a atribútová množina *AttrRpart4* (tab. 5.3, riadok 10). Pre metódu Naive Bayes som vybral pôvodné nemodifikované dáta a sadu atribútov *AttrBayes1* (tab. 5.3, riadok 6) a pre metódu SVM automaticky modifikované dáta s použitím všetkých atribútov, ktoré ostali po aplikovaní tejto modifikácie (tab. 5.3, riadok 2).

Počas všetkých doterajších experimentov zaznamenávala metóda Naive Bayes najväčšie fluktuácie a nestabilitu a navyše pri najlepšom výsledku u metódy Naive Bayes boli zároveň dosiahnuté najhoršie výsledky u zvyšných dvoch metód (horšie ako na pôvodných dátach, viď. tabuľka 5.3, riadky 6 a 1). Naopak modifikácia popísaná v kapitole 5.4.3 zaznamenala u všetkých metód podobne vysokú úspešnosť, čo signalizuje možnú vyššiu stabilitu takéhoto výberu a príslušnej modifikácie. Preto som sa rozhodol do ďalšieho kroku nechať postúpiť aj model, ktorý vznikol natrénovaním metódou Naive Bayes na takto modifikovaných dátach, čiže druhý najúspešnejší model u metódy Naive Bayes.

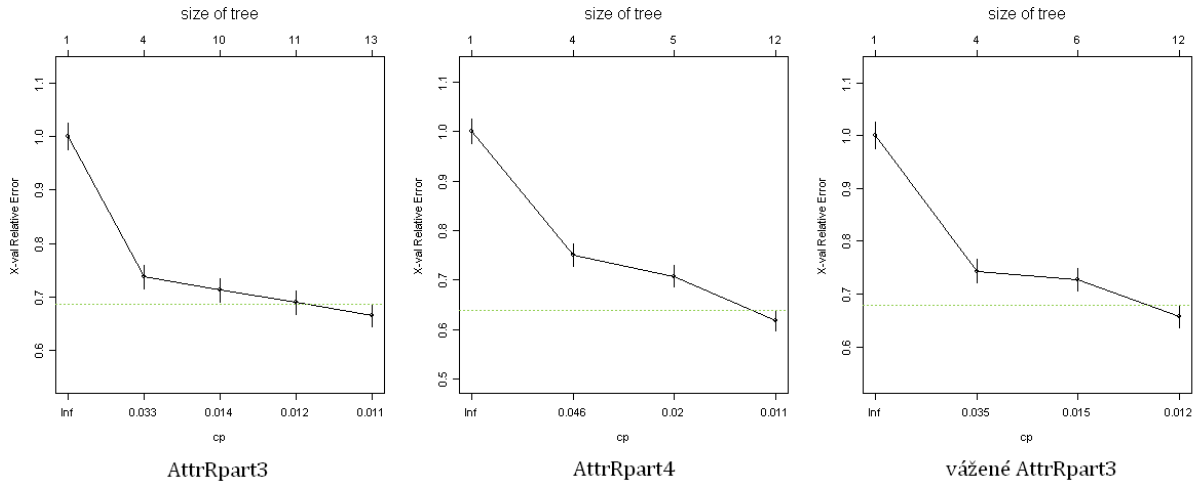
	Odkaz na kapitolu	Výber atribútov, modifikácia dát	rpart	naiveBayes	svm	Počet atrib.
1.	5.1	Pôvodné dáta	60,53%	42,08%	64,13%	53
2.	5.2	Autom. modifikované dáta (0,03; 0,01)	61,70%	52,76%	70,06%	37
3.	5.3.2	Pôvodné dáta, autom. výber atribútov metódou `rpart` (*AttrRpart1*)	66,11%	44,10%	60,06%	8
4.	5.3.3	Autom. modifikované dáta (0,03; 0,01), autom. výber atribútov metódou `rpart` na pôvodných dátach (*AttrRpart1*)	58,79%	50,54%	62,81%	8
5.	5.3.3	Autom. modifikované dáta (0,03; 0,01), autom. výber atribútov metódou `rpart` na autom. modifikovaných dátach (*AttrRpart2*)	62,82%	55,41%	63,13%	6
6.	5.3.4	Pôvodné dáta, autom. výber atribútov metódou `naiveBayes` (*AttrBayes1*)	57,22%	63,11%	59,56%	11
7.	5.4.1	Pôvodné dáta, autom. výber atribútov metódou `rpart`, bez `sent_dist` (*AttrRpart3*)	66,11%	44,79%	60,22%	7
8.	5.4.2	Manuálne modifikované dáta, autom. výber atribútov metódou `rpart`, bez `sent_dist` (*AttrRpart3*)	66,11%	61,40%	65,56%	7
9.	5.4.3	Manuálne modifikované dáta + modifikácia `cand_asubpos`, autom. výber atribútov metódou `rpart`, bez `sent_dist` (*AttrRpart3*)	66,11%	61,74%	65,92%	7
10.	5.4.4	Manuálne modifikované dáta + modifikácia `cand_asubpos`, autom. výber atribútov metódou `rpart` na modifikovaných dátach (*AttrRpart4*)	66,30%	46,77%	65,58%	9

Tabuľka 5.3 Úspešnosti testov (metrikou F-measure) počas hľadania optimálnej množiny atribútov a modifikácie dát (červené výsledky sú najlepšie, zeleno podfarbené sú vybraté)

6 Optimalizácia metód

Po vybraní najvhodnejších atribútov som ešte zisťoval, či zmena voliteľných parametrov u jednotlivých metód strojového učenia prinesie zlepšenie úspešnosti na vývojových dátach.

U rozhodovacích stromov som experimentoval s váhami priradenými atribútom. Týmito váhami sa delí miera, podľa ktorej sa určuje podmienka pre výstavbu podstromov, takže zvyšovaním váhy atribútu sa znižuje jeho dôležitosť (presne naopak, ako by sa dalo očakávať). Predvolené nastavenie je váha rovná 1. Zistil som, že optimálny výsledok (67,21%) zabezpečia vysoké váhy u `epar_fun_agree` a `cand_subj` a zároveň vhodne, v závislosti na veľkosti váh predchádzajúcich atribútov, zvolená váha atribútu `cand_ord` v rozmedzí intervalu ⟨12,100⟩. Z pozorovania štruktúry rozhodovacieho stromu pre toto nastavenie som však zistil, že atribúty `epar_fun_agree` a `cand_subj` v ňom nie sú vôbec použité, takže pri trénovaní tohto modelu stačí atribútová množina *AttrRpart3*. Úspešnosť bez váženia je na tejto množine síce nižšia ako na množine *AttrRpart4*, ale s váhou v spomínanom rozmedzí pre atribút `cand_ord` je výsledok najlepší (pre rozhodovacie stromy).

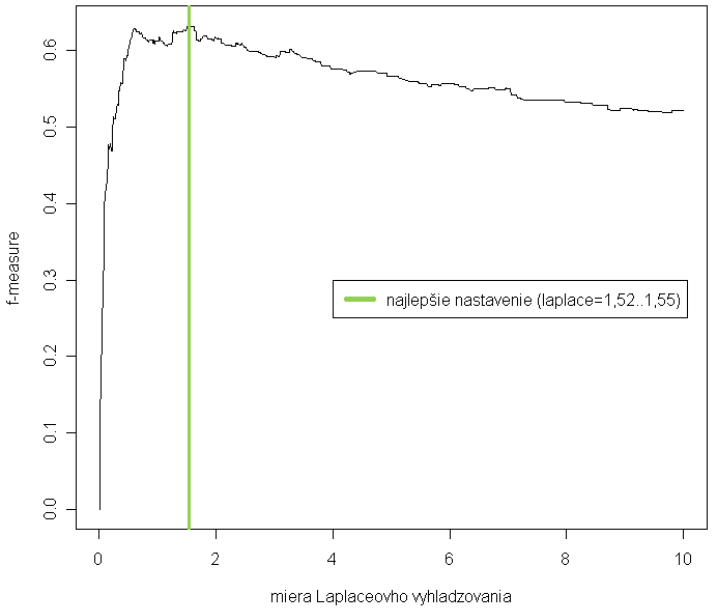


Obrázok 6.1 Relatívna chyba klasifikácie v závislosti na veľkosti stromu (danej hodnotou complexity parametru) pri troch variantách modelu Rpart. Optimálnu veľkosť stromu určuje najľavejší bod, ktorý sa nachádza pod zelenou čiarou

Ešte lepších výsledkov by bolo možné dosiahnuť orezávaním rozhodovacieho stromu. Mieru orezania určuje tzv. complexity parameter (`cp`). Grafy na obrázku 6.1 sú výsledkom volania funkcie `plotcp` na modeloch natrénovaných na príslušných atribútových množinách (prípadne s vážením popísaným vyššie). Všetky zobrazené parametre sú vypočítané z výsledkov 10-násobnej krížovej validácie[8] na trénovacích dátach, kedy tak pre rôzne hodnoty `cp` vznikli odhady stredných hodnôt a smerodajných odchýlok chýb. Tieto chyby sa počítajú ako jednotkový doplnok miery accuracy[9] aplikovanej na výsledky[10]. Grafy zobrazujú závislosť práve

[8] http://en.wikipedia.org/wiki/Cross-validation
[9] viď. poznámku 7 v kapitole 4.2
[10] Celý tento výpočet prebieha pri vytváraní modelu príkazom `rpart`.

stredných hodnôt týchto chýb na miere orezania stromu, čiže parametri `cp`. Okrem toho je v grafoch zelenou čiarou znázornená hladina, ktorá určuje súčet strednej hodnoty a smerodajnej odchýlky chýb pre orezanie s najmenšou strednou hodnotou chýb. Maximálne orezanie také, že stredná hodnota jeho chýb je menšia ako táto hladina, je optimálne orezanie. V grafoch na obrázku 6.1 je to tá hodnota `cp`, u ktorej bodka (určujúca strednú hodnotu chýb) leží pod zelenou čiarou a zároveň čo najviac vľavo. [9, str. 260] Pre váženú atribútovú množinu *AttrRpart3* to platí pre hodnotu `cp` takú, že veľkosť stromu je 12, čo zodpovedá veľkosti stromu bez orezania. Z toho plynie, že nie je možné orezať použitý rozhodovací strom tak, aby sa razantne neznížila jeho úspešnosť klasifikácie.



Obrázok 6.2 Závislosť F-measure na nastavení Laplaceovho vyhladzovania pri modeli natrénovanom metódou Naive Bayes

Metóda Naive Bayes (trénovaná na atribútovej množine *AttrBayes1*) umožňuje pozmeniť aposteriórne pravdepodobnosti tzv. Laplaceovým vyhladzovaním. Graf na obrázku 6.2 ukazuje funkciu hodnôt F-measure v závislosti na miere vyhladzovania. Vidieť z neho, že optimálna hodnota tohto parametru je v intervale $\langle 1{,}52; 1{,}55 \rangle$. Vyhladzovanie nespôsobilo výrazné vylepšenie, iba o 0,19%.

Na druhý model natrénovaný metódou Naive Bayes (vznikol modifikáciou dát popísanou v kapitole 5.4.3) som sa rovnako pokúšal aplikovať Laplaceovo vyhladzovanie. Tento model však najlepšiu úspešnosť dosiahol bez parametrizácie, čiže bez vyhladzovania.

Metóda SVM ponúka viac možností parametrizácie. Jednak je možné zameniť jadrovú transformačnú funkciu (a modifikovať ju ďalšími parametrami). Predvolené nastavenie u používanej funkcie `svm` je radiálna bázová funkcia (radial basis function).[11] Druhý spôsob parametrizácie je prostredníctvom nastavenia miery penalizácie za neseparovateľnosť inštancií (predvolená hodnota je 1). Po niekoľkých

[11] http://en.wikipedia.org/wiki/Radial_basis_function

experimentoch s rôznymi hodnotami parametrov sa ukázalo, že metóda SVM dosahuje maximálne zlepšenie pri voľbe polynomiálnej jadrovej funkcie stupňa 2 a s hodnotou penalizácie 13,5. Úspešnosť pri takomto nastavení je na vývojových testovacích dátach výborných 75%.

Výsledky parametrizácie modelov sú v tabuľke 6.1. Tieto na vývojových dátach optimalizované modely označím *rpart.best*, *bayes.best, bayes.alt* a *svm.best*.

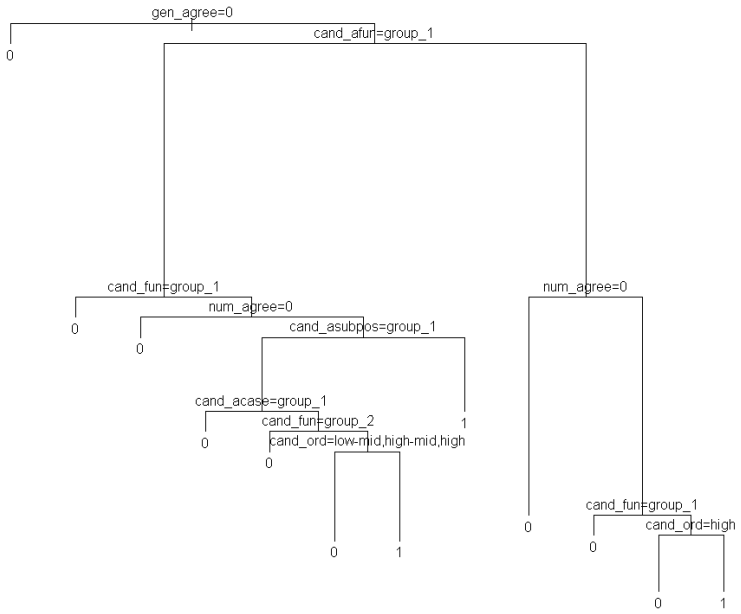
Použitá modifikácia (odkaz na kapitolu)	**Metóda**	**Parametrizácia**	**Označenie modelu**	**Úspešnosť**	
				Predvolené param.	**Nové param.**
5.4.3	rpart	$cost(cand\_ord) \in \langle 12; 100 \rangle$	rpart.best	66,11%	67,21%
5.3.4	naiveBayes	$Laplace \in \langle 1{,}52; 1{,}55 \rangle$	bayes.best	63,11%	63,30%
5.4.3	naiveBayes	-	bayes.alt	61,74%	61,74%
5.2	svm	$polynomial\ kernel;$ $degree = 2; cost = 13{,}5$	svm.best	70,06%	75%

Tabuľka 6.1 Úspešnosti metód pri predvolených a zmenených parametroch

7 Vyhodnotenie modelov

Dáta pre trénovanie modelov jednotlivými metódami a parametre týchto metód boli upravené nasledovne:

- Rozhodovacie stromy:
  - *rpart.best*
    - vybrané atribúty podľa množiny *AttrRpart3*
    - zoskupené hodnoty atribútov podľa prílohy 2
    - váha atribútu `cand_ord` na 12 (štruktúra finálneho rozhodovacieho stromu je na obrázku 7.1)
- Naive Bayes:
  - *bayes.best*
    - vybrané atribúty podľa množiny *AttrBayes1*
    - koeficient Laplaceovho vyhladzovania na 1,52
  - *bayes.alt*
    - vybrané atribúty podľa množiny *AttrRpart3*
    - zoskupené hodnoty atribútov podľa prílohy 2
    - predvolené nastavenia parametrov
- SVM:
  - *svm.best*
    - dáta modifikované automaticky s parametrami $\varepsilon = 0{,}03$ a $\theta = 0{,}01$ (viď. kapitola 5.2), prípadne tomu ekvivalentný manuálny výber atribútov a zoskupenie ich hodnôt podľa prílohy 1
    - použitie polynomiálnej jadrovej funkcie stupňa 2 a cena penalizácie nastavená na 13,5



Obrázok 7.1 Štruktúra rozhodovacieho stromu finálneho modelu *rpart.best*

Modely vytvorené na základe týchto kritérií boli otestované na evaluačných testovacích dátach, ktoré sa do tohto momentu ešte nepoužili. Pre porovnanie boli vykonané testy aj na trénovacích dátach. Úspešnosti na vývojových dátach sú k dispozícii z predchádzajúcich experimentov. Okrem F-measure, boli zaznamenané aj hodnoty precision a recall. Súhrnné výsledky sú v tabuľke 7.1.

Model	Trénovacie data			Vývojové dáta			Evaluačné dáta		
	P	R	F	P	R	F	P	R	F
rpart.best	71,39%	63,41%	67,17%	68,89%	65,61%	67,21%	68,29%	67,63%	67,96%
bayes.best	64,91%	61,62%	63,23%	55,87%	73,02%	63,30%	65,22%	50,72%	57,07%
bayes.alt	62,20%	59,84%	61,00%	53,49%	73,02%	61,74%	58,21%	75,36%	65,68%
svm.best	77,20%	76,97%	77,09%	72,41%	77,78%	75,00%	72,27%	76,81%	74,47%

Tabuľka 7.1 Hodnoty úspešností (precision, recall, F-measure) finálnych modelov na všetkých typoch dát

Medzi úspešnosťami na evaluačných dátach sú markantné rozdiely. Model *svm.best* klasifikuje najlepšie, ale zatiaľ čo úspešnosť u tohto modelu oproti vývojovým dátam klesla nepatrne, u modelu *bayes.best* klesla razantne. Naopak, ukázalo sa, že model *bayes.alt* je stabilnejší ako model *bayes.best*, keďže jeho úspešnosť sa priblížila k výsledku modelu *rpart.best* (jeho F-measure na evaluačných dátach tiež nepatrne vzrástla) a vznikol tak takmer 9%-ný rozdiel medzi modelmi trénovanými metódou Naive Bayes, hoci na vývojových dátach bol tento rozdiel iba necelé 2%. Takéto chovanie je možné vysvetliť pravdepodobným pretrénovaním modelu *bayes.best*, teda tým, že atribútová množina *AttrBayes1* je až príliš prispôsobená vývojovým dátam. Zvyšné modely sú v tomto smere konzistentné.

Zaujímavý je enormný vzrast úspešnosti u modelu *bayes.alt*. Zrejme však ide iba o náhodu, že evaluačné testovacie dáta sú o toľko lepšie prispôsobené tomuto modelu. Podiel na tom môže mať aj vyššia distribúcia pozitívnych inštancií v evaluačných dátach oproti tým vývojovým (viď. tabuľka 2.1).

Z tabuľky 7.1 je rovnako vidieť, že pri vyhodnocovaní modelov na trénovacích dátach sa lepšia znalosť týchto dát oproti iným prejavila v najvyšších hodnotách miery precision, teda najvyššou presnosťou pri klasifikácii pozitívnych inštancií.

7.1 Skutočná úspešnosť a rozdiel v kvalite metód

Z predchádzajúcej kapitoly je zjavné, že model *svm.best* má na evaluačných dátach najvyššiu úspešnosť 74,47% s veľkými rozdielmi v porovnaní s ostatnými. Bude mať však tento model takéto výsledky aj na úplne odlišných dátach? Aká je teda skutočná úspešnosť na akejkoľvek kombinácii dát? Podobne, budú rozdiely medzi modelmi také značné aj pri iných dátach, inak povedané – sú tieto rozdiely štatisticky významné?

Odpoveďou na prvé dve otázky je interval spoľahlivosti na hladine významnosti napr. 95%. Tento interval je možné získať viacerými spôsobmi. Jeden možný spôsob je výpočtom pomocou kvantilov normálneho rozdelenia tak, ako je popísané v [10]. Ja som však použil metódu bootstrapping. Metóda bootstrapping spočíva v mnohonásobnom (v mojom prípade $N = 1000$) výbere vzoriek z testovacej množiny veľkosti n tak, že vzorka má tiež veľkosť n, avšak jednotlivé inštancie sa

v nej môžu opakovať. Je to teda náhodný[12] výber kombinácie s opakovaním veľkosti n z množiny rovnakej veľkosti. Na každej takto vybratej vzorke sa spočíta zvolená štatistika (v mojom prípade F-measure) a týchto N hodnôt sa usporiada podľa veľkosti. V tomto bode je už možné získať interval spoľahlivosti, ale opäť viacerými spôsobmi. Ja som zvolil spôsob, ktorého výsledkom bol obojstranný percentilový interval spoľahlivosti na hladine významnosti 95%. Z usporiadanej množiny N hodnôt F-measure som zahodil prvých 2,5% a posledných 2,5% hodnôt (u mňa teda 25 hodnôt z každej strany) a to, čo ostalo, je požadovaný interval spoľahlivosti. Tento aj iné spôsoby konštrukcie intervalov spoľahlivosti sú popísané v [9, str. 133-138]. Výsledkom mojich experimentov pre optimalizovanú metódu SVM je interval v tabuľke 7.2. To znamená, že s pravdepodobnosťou 95% tento interval pokrýva skutočnú hodnotu úspešnosti modelu natrénovaného metódou SVM (optimalizovanou).

Model	95%-ný interval spoľahlivosti
svm.best	(69,42%; 79,11%)

Tabuľka 7.2 Interval spoľahlivosti F-measure modelu SVM na hladine významnosti 95%

Odpoveď na otázku štatistickej významnosti (signifikancie) rozdielu medzi metódami som našiel analogickým postupom. Použil som opäť bootstrapping s jednou odlišnosťou: vyhodnocoval som súčasne 2 metódy na rovnakých vzorkách a výsledky jednej metódy som odčítaval od výsledkov druhej. Vznikla tým opäť postupnosť N čísel, z ktorej som obojstranné intervaly spoľahlivosti skonštruoval rovnako ako v predchádzajúcom prípade. Ak výsledný interval pokrýva číslo 0, nie je rozdiel medzi úspešnosťou metód signifikantný. Ak interval nulu nepokrýva, rozdiel je štatisticky významný a môžeme tvrdiť (s 95% istotou), že jedna metóda je lepšia ako druhá. Signifikanciu rozdielov som testoval pre tieto dvojice modelov: *bayes.best – bayes.alt, bayes.alt – rpart.best* a *rpart.best – svm.best*. Z tabuľky 7.3 vidieť, že nulu pokrýva jedine interval spoľahlivosti rozdielu metód *bayes.alt* a *rpart.best*, čiže pre túto dvojicu nemôžeme zamietnuť hypotézu, že ich úspešnosť je v skutočnosti rovnaká. Intervaly spoľahlivosti pre zvyšné dvojice nepokrývajú nulu, takže pre výsledné modely platí:

$$bayes.best < bayes.alt \approx rpart.best < svm.best,$$

pričom operátor $<$ a $\approx$ označujú reláciu usporiadania na úspešnostiach modelov (vyjadrených hodnotou F-measure).

Pri popísaných experimentoch som používal implementáciu bootstrappingu v R knižnici `boot`, a to konkrétne funkcie `boot` a `boot.ci`.

Rozdiel modelov	95%-ný interval spoľahlivosti
bayes.best – bayes.alt	(−15,43%; −2,07%)
bayes.alt – rpart.best	(−5,90%; 1,92%)
rpart.best – svm.best	(−10,82%; −2,45%)

Tabuľka 7.3 Intervaly spoľahlivosti rozdielu F-measure natrénovaných modelov na hladine významnosti 95%

[12] Ďalej uvedené výsledky sú pre random seed inicializovaný číslom 1986

8 Záver

V tejto práci som riešil úlohu automatického určovania koreferencií. Na to som používal metódy strojového učenia, a to rozhodovacie stromy, metódu Naive Bayes a metódu SVM. Mojim cieľom bolo vybrať vhodné atribúty a modifikovať dáta tak, aby všetky použité metódy dosahovali čo najlepšie výsledky. Výsledné modely dosiahli na evaluačných testovacích dátach nasledovné hodnoty F-measure:

- Rozhodovacie stromy: 67,96%
- Naive Bayes: 65,68%
- SVM: 74,47%

Rozdiel medzi SVM a zvyšnými dvomi metódami je veľký a ukázalo sa, že tento rozdiel je aj štatisticky významný, t.j. i reálne je model natrénovaný metódou SVM (*svm.best*) z nich najlepší. Model *svm.best* navyše dosahuje porovnateľný výsledok F-measure s tým, uverejneným v práci [4]. Naopak, nemôžeme zamietnuť to, že model vytvorený pomocou rozhodovacích stromoch je rovnako úspešný ako ten, ktorý vznikol metódou Naive Bayes.

V tejto práci som uviedol metódu automatickej modifikácie dát, ktorá funguje na princípe zlučovania hodnôt atribútov s podobnou distribúciou voči cieľovej klasifikácii v trénovacích inštanciách a následnom odstraňovaní atribútov s výraznou početnou prevahou jednej hodnoty oproti ostatným. Vďaka tejto modifikácii som mohol vytvoriť model, ktorý pri trénovaní metódou SVM dosiahol vyššie spomenutý najlepší výsledok.

Rovnako sa v priebehu práce na tejto úlohe ukázalo, že najväčšie problémy robí metóda Naive Bayes. Dosahovala vždy najhoršie úspešnosti a musel som odskúšať množstvo alternatív (jedna z nich, ktorá sa javila ako najlepšia, nakoniec trpela „pretrénovaním“) na to, aby som získal finálny model, ktorý je už porovnateľný s modelom natrénovaným metódou rozhodovacích stromov.

Napriek tomu, že model *svm.best* dosahuje vynikajúce výsledky, na jeho vytvorenie je potrebné veľké množstvo atribútov, čo je aj vidieť v prílohe 1. Jeho vhodným znížením by sa model zjednodušil a pritom by sa mohla zachovať jeho kvalita. Tým sa však už táto práca zaoberať nebude…

Použitá literatúra

[1] Gigel, M. (2009). Upečte si s nami sypané žemle a kaiserky. *Sme.sk*, from <http://domacnost.sme.sk/c/4329652/upecte-si-s-nami-sypane-zemle-a-kaiserky.html>

[2] Vidová-Hladká, B. (2008). *Určování koreference*. Univerzita Karlova, Praha. from <http://ufal.mff.cuni.cz/~hladka/ML/PFL054_2008_09_zapocet.pdf>

[3] Linh, N. G. (2006). *Návrh souboru pravidel pro analýzu anafor v českém jazyce.* Univerzita Karlova, Praha. from <http://ufal.mff.cuni.cz/~linh/theses/aca-diplomka.pdf>

[4] Linh, N. G. and Žabokrtský, Z. (2007). *Rule-based Approach to Pronomial Anaphora Resolution Applied on the Prague Dependency Treebank 2.0 Data*. Paper presented at the DAARC 2007 (6th Discourse Anaphora and Anaphor Resolution Colloquium). from http://ufal.mff.cuni.cz/~zabokrtsky/papers/daarc-2007.pdf

[5] Vidová-Hladká, B. (2008). *Rozhodovací stromy*. Univerzita Karlova, Praha. from <http://ufal.mff.cuni.cz/~hladka/ML/LECTURES/decision_trees_22_10_cs.pdf>

[6] Vidová-Hladká, B. (2008). *Bayesovské učení*. Univerzita Karlova, Praha. from <http://ufal.mff.cuni.cz/~hladka/ML/LECTURES/bayes_learning_12_11_cs.pdf>

[7] Žižka, J. (2004). *Support Vector Machines (SVM)*. Masarykova Univerzita, Brno. from <http://is.muni.cz/el/1433/podzim2006/PA034/09_SVM.pdf>

[8] Wikipedia. (2008). Support vector machines. from <http://cs.wikipedia.org/wiki/Support_vector_machines>

[9] Venables, W. N. and Ripley, B. D. (2002). *Modern Applied Statistics with S*. Springer-Verlag, New York.

[10] Vidová-Hladká, B. (2007). *Vyhodnocování hypotéz*. Univerzita Karlova, Praha. from <http://ufal.mff.cuni.cz/~hladka/ML/LECTURES/hypothesis_evaluation_5_11_cs.pdf>

Prílohy

V nasledujúcich tabuľkách sú uvedené atribútové množiny a modifikácie hodnôt atribútov pre použité finálne modely *svm.best, rpart.best* a *bayes.alt*.

Názov atribútu	Popis atribútu	Nová hodnota	Pôvodné hodnoty
cand_gen	Rod kandidáta	GROUP_inan	inan, nr, fem
		GROUP_neut	neut
		GROUP_anim	anim, empty
		GROUP_inher	inher
cand_num	Číslo kandidáta	GROUP_nr	nr
		GROUP_sg	sg, pl
		GROUP_empty	empty
		GROUP_inher	inher
anaph_gen	Rod anafory	GROUP_neut	neut, inan, fem, anim
		GROUP_nr	nr
gen_agree	Zhoda kandidáta a anafory v rode	GROUP_0	0
		GROUP_1	1
num_agree	Zhoda kandidáta a anafory v čísle	GROUP_0	0
		GROUP_1	1
cand_coord	Je kandidát v koordinácii?	GROUP_0	0
		GROUP_1	1
cand_epar_fun	Funktor efektívneho rodiča kandidáta	GROUP_ADDR	ADDR, ACMP, LOC, DIR1, PAR, DIR3, CPHR, COMPL, APP, DENOM, ACT, CPR, MANN, CRIT, TWHEN
		GROUP_ATT	ATT, AUTH, DIFF, DPHR, EXT, ID, INTT, RESTR, SUBS, TFHL, TFRWH, THL, THO, TSIN, TTILL
		GROUP_BEN	BEN
		GROUP_CAUS	CAUS
		GROUP_HER	HER, VOCAT, DIR2
		GROUP_ORIG	ORIG, TOWH, TPAR
		GROUP_RESL	RESL, EFF, PRED, AIM, MAT, MEANS, COND, CONTRD
		GROUP_RSTR	RSTR, PAT, empty, CNCS, REG
cand_epar_sempos	Sémantický slovný druh efektívneho rodiča kandidáta	GROUP_adjspec_dotpronspec_dotindef	adjspec_dotpronspec_dotindef, adjspec_dotquantspec_dotdef, adjspec_dotquantspec_dotgrad, advspec_dotdenotspec_dotgradspec_dotneg, advspec_dotdenotspec_dotgradspec_dotnneg, advspec_dotdenotspec_dotngradspec_dotnneg, nspec_dotpronspec_dotindef, nspec_dotquantspec_dotdef, adjspec_dotdenot
		GROUP_adjspec_dotquantspec_dotindef	adjspec_dotquantspec_dotindef, nspec_dotpronspec_dotdefspec_dotdemon, advspec_dotpronspec_dotindef, advspec_dotpronspec_dotdef, nspec_dotpronspec_dotdefspec_dotpers
		GROUP_nspec_dotdenot	nspec_dotdenot, empty
		GROUP_nspec_dotdenotspec_dotneg	nspec_dotdenotspec_dotneg
		GROUP_v	v
anaph_epar_fun	Funktor efektívneho rodiča anafory	GROUP_COMPL	COMPL, ADDR, CPHR, SUBS, DIR3, MEANS
		GROUP_ID	ID, INTT
		GROUP_LOC	LOC, ACT, PRED, TTILL, CAUS
		GROUP_PAR	PAR, COND, MAT, RESL, TWHEN, CONTRD, TPAR, CPR, PAT, CRIT, ACMP, RSTR, EFF, AIM, DIR1, MANN, APP, CNCS, EXT, REG
		GROUP_TSIN	TSIN, BEN, HER, DPHR, DIFF, DENOM, ORIG, THL, RESTR, DIR2, THO
epar_fun_agree	Zhoda kandidáta a anafory vo funktore efektívneho rodiča	GROUP_0	0
		GROUP_1	1
epar_sempos_agree	Zhoda kandidáta a anafory v sémantickom slovnom druhu efektívneho rodiča	GROUP_0	0
		GROUP_1	1
epar_lemma_agree	Zhoda kandidáta a anafory v lemme efektívneho rodiča	GROUP_0	0
		GROUP_1	1
cand_fun	Funktor kandidáta na tektogramatickej rovine	GROUP_ACT	ACT
		GROUP_ADDR	ADDR
		GROUP_CAUS	CAUS, CNCS, COMPL, COND, CRIT, DIFF, DIR2, EXT, INTT, RESL, RESTR, TFHL, TFRWH, THL, THO, TOWH, TPAR, TSIN, TTILL, TWHEN, RSTR, ID, MEANS, AIM, PAR, MANN, ACMP, CPR
		GROUP_DENOM	DENOM, BEN, AUTH
		GROUP_HER	HER, VOCAT
		GROUP_ORIG	ORIG, PAT, APP, SUBS
		GROUP_REG	REG, CPHR, LOC, DIR1, EFF, DIR3, MAT
anaph_fun	Funktor anafory na tektogramatickej rovine	GROUP_CPHR	CPHR, COND, REG, ORIG, TWHEN, ACMP, EFF, BEN, DIR1, PAT
		GROUP_DIR3	DIR3
		GROUP_LOC	LOC, ADDR, APP, CRIT, ACT, CPR, RESTR, MAT, AUTH, INTT
		GROUP_MEANS	MEANS, HER, CAUS, SUBS, AIM, EXT, RESL, MANN, DIR2
		GROUP_TPAR	TPAR
fun_agree	Zhoda kandidáta a anafory vo funktore na tektogramatickej rovine	GROUP_0	0
		GROUP_1	1

cand_afun	Funktor kandidáta na analytickej rovine	GROUP_Atr	Atr
		GROUP_AtrAdv	AtrAdv, AtrAtr, AtrObj, AtvV, AuxP, AuxY, Coord, Pnom
		GROUP_Atv	Atv, Adv, ExD
		GROUP_AuxC	AuxC, ObjAtr, AuxG, AuxZ
		GROUP_AuxO	AuxO
		GROUP_AuxT	AuxT
		GROUP_Obj	Obj, AdvAtr
		GROUP_Sb	Sb
		GROUP_empty	empty
anaph_afun	Funktor anafory na analytickej rovine	GROUP_AdvAtr	AdvAtr
		GROUP_AtrAtr	AtrAtr, AtrAdv, AtrObj, ObjAtr
		GROUP_AuxP	AuxP
		GROUP_ExD	ExD
		GROUP_Pnom	Pnom, Atv, Atr, Obj, Adv, empty
		GROUP_Sb	Sb
afun_agree	Zhoda kandidáta a anafory vo funktore na analytickej rovine	GROUP_0	0
		GROUP_1	1
cand_apos	Slovný druh kandidáta	GROUP_A	A
		GROUP_C	C
		GROUP_D	D, I, R, T
		GROUP_J	J
		GROUP_N	N
		GROUP_P	P
		GROUP_empty	empty
cand_asubpos	Poddruh slovného druhu kandidáta	GROUP_2	2, C, E, G, I, L, O, Q, R, T, W, b, spec_eq, y
		GROUP_5	5
		GROUP_6	6
		GROUP_8	8
		GROUP_A	A, 1, 4
		GROUP_J	J, K, D
		GROUP_N	N, Z
		GROUP_S	S, 7, l
		GROUP_U	U, 9
		GROUP_empty	empty, P
		GROUP_n	n, H
		GROUP_spec_comma	spec_comma, spec_head, V
cand_agen	Rod kandidáta	GROUP_I	I, undef, F
		GROUP_M	M
		GROUP_N	N
		GROUP_X	X, Z
		GROUP_Y	Y, H
		GROUP_empty	empty
cand_anum	Číslo kandidáta	GROUP_D	D, undef
		GROUP_S	S, P, X
		GROUP_empty	empty
cand_acase	Pád kandidáta	GROUP_1	1
		GROUP_3	3
		GROUP_4	4, X
		GROUP_5	5
		GROUP_7	7, 2
		GROUP_empty	empty
		GROUP_undef	undef, 6
cand_apossgen	Privlastňovací rod kandidáta	GROUP_F	F
		GROUP_M	M
		GROUP_X	X
		GROUP_Z	Z
		GROUP_empty	empty
		GROUP_undef	undef
cand_apossnum	Privlastňovacie číslo kandidáta	GROUP_P	P
		GROUP_S	S
		GROUP_empty	empty
		GROUP_undef	undef
cand_apers	Osoba kandidáta	GROUP_3	3, empty
		GROUP_undef	undef
cand_akt	Je kandidát aktantom?	GROUP_0	0
		GROUP_1	1
akt_agree	Zhoda kandidáta a anafory v tom, či sú aktantom	GROUP_0	0
		GROUP_1	1
cand_subj	Je kandidát subjektom?	GROUP_0	0
		GROUP_1	1
subj_agree	Zhoda kandidáta a anafory v tom, či sú subjektom	GROUP_0	0
		GROUP_1	1
sent_dist	Sú kandidát a anafora v rovnakej vete?	GROUP_0	0
		GROUP_1	1
clause_dist	Počet klauzúl medzi kandidátom a anaforou	continuous	continuous
file_deepord_dist	Rozdiel hĺbkového poradia kandidáta a anafory	continuous	continuous
cand_ord	Poradie kandidáta od anafory	continuous	continuous
cand_tfa	Aktuálne členenie kandidáta	GROUP_c	c, t
		GROUP_f	f
tfa_agree	Zhoda v aktuálnom členení kandidáta a anafory	GROUP_0	0
		GROUP_1	1
sibl	Sú kandidát a anafora súrodenci	GROUP_0	0
		GROUP_1	1

Príloha 1 Výber atribútov s popisom a pravidlá pre modifikáciu ich hodnôt pre model *svm.best*

Názov atribútu	Nová hodnota	Pôvodné hodnoty
gen_agree	0	0
	1	1
num_agree	0	0
	1	1
cand_fun	group_1	ACMP, AIM, CAUS, CNCS, COMPL, COND, CPHR, CPR, CRIT, DIFF, DIR1, DIR2, EXT, ID, MANN, MAT, MEANS, PAR, REG, RESTR, RSTR, SUBS, TFRWH, THL, THO, TOWH, TPAR, TSIN, TTILL, TWHEN
	group_2	APP, AUTH
	group_3	ACT, ADDR, BEN, DIR3, EFF, LOC, ORIG, PAT
cand_afun	group_1	Adv, Atr, AtrAdv, AtrAtr, AtrObj, Atv, AtvV, AuxP, ExD, Obj, Pnom
	group_2	AdvAtr, AuxT, Sb, empty
cand_asubpos	group_1	7, 8, 9, A, D, K, N, U, Z, Q, E, G, J
	group_2	1, 4, 5, 6, H, P, S, l, n, spec_eq, empty
cand_acase	group_1	1, 6, 7, X
	group_2	2, 3, 4
cand_ord	low	$(-\infty; 3{,}5)$
	low-mid	$\langle 3{,}5; 5{,}5)$
	high-mid	$\langle 5{,}5; 8{,}5)$
	high	$\langle 8{,}5; \infty)$

Príloha 2 Výber atribútov a pravidlá pre modifikáciu ich hodnôt pre model *rpart.best* a *bayes.alt*