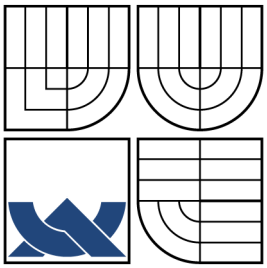
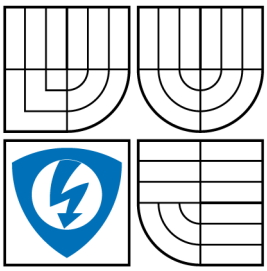


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA ELEKTROTECHNIKY
A KOMUNIKAČNÍCH TECHNOLOGIÍ
ÚSTAV AUTOMATIZACE A MĚŘICÍ TECHNIKY

FACULTY OF ELECTRICAL ENGINEERING AND
COMMUNICATION
DEPARTMENT OF CONTROL AND INSTRUMENTATION

SELEKCE PŘÍZNAKŮ POMOCÍ NEKORELOVANÝCH CHARAKTERISTIK

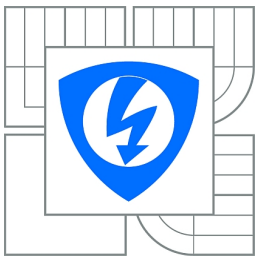
FEATURE SELECTION BASED ON COMBINATION OF UNCORRELATED EVALUATION FUNCTIONS

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — Bc. KAREL VACULÍK
AUTHOR

VEDOUCÍ PRÁCE — Ing. PETR HONZÍK, Ph.D.
SUPERVISOR

BRNO 2013



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky a komunikačních technologií

Ústav automatizace a měřicí techniky

Bakalářská práce

bakalářský studijní obor
Automatizační a měřicí technika

Student:	Bc. Karel Vaculík	***ID:***	134649
Ročník:	3	***Akademický rok:***	2012/2013

NÁZEV TÉMATU:

Selekce příznaků pomocí nekorelovaných charakteristik

POKYNY PRO VYPRACOVÁNÍ:

- seznamte se s metodami selekce příznaků používanými ve strojovém učení a vypracujte jejich stručný přehled; zaměřte se na tzv. filter feature selection (FFS) metody
- proveďte rešerši studií srovnávajících přesnost FFS metod
- zopakujte obdobné experimenty a srovnejte vlastní a publikované výsledky
- na základě vlastních experimentů vytvořte korelační matici FFS metod (na základě pořadí vybraných příznaků)
- navrhněte novou FFS metodu kombinující metody původní, využijte zjištěné přesnosti a korelační matici
- srovnejte novou FFS metodu s metodami původními

DOPORUČENÁ LITERATURA:

Isabelle Guyon and André Elisseeff. 2003. An introduction to variable and feature selection. J. Mach. Learn. Res. 3 (March 2003), 1157-1182.

Termín zadání:	11.2.2013	***Termín odevzdání:***	27.5.2013

Vedoucí práce: Ing. Petr Honzík, Ph.D.
Konzultanti bakalářské práce:

doc. Ing. Václav Jirsík, CSc.
Předseda oborové rady

UPOZORNĚNÍ:

ABSTRAKT

Ke zpracování nadměrného množství dat v digitální podobě je zapotřebí použít prostředků výpočetní techniky. V některých případech je možné použít statistické metody nebo strojové učení. V obou případech mohou být data reprezentována velkým počtem příznaků. Pro efektivní zpracování může hrát důležitou roli výběr pouze určité množiny příznaků, které jsou relevantní. Tato práce zkoumá podskupinu metod pro výběr příznaků, tzv. filter metody. Tyto metody jsou mezi sebou porovnány a na základě výsledků je navržena nová metoda, která je kombinací metod původních.

KLÍČOVÁ SLOVA

strojové učení, selekce příznaků, filter metody, vyhodnocení

ABSTRACT

In order to process large amount of data, it is necessary to use computers. It is possible to use statistical methods or machine learning in some cases. In either case, data can be represented with large number of features. Selection of suitable subset of features can be crucial for efficient processing. This thesis explores a subgroup of feature selection methods which are called filter methods. Comparison of such methods is carried out and the results are used in the design of a new method. This new method uses a combination of existing methods.

KEYWORDS

machine learning, feature selection, filter methods, evaluation

VACULÍK, Karel *Selekce příznaků pomocí nekorelovaných charakteristik*: bakalářská práce. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav automatizace a měřicí techniky, 2013. 42 s. Vedoucí práce byl Ing. Petr Honzík, Ph.D.

PROHLÁŠENÍ

Prohlašuji, že svou bakalářskou práci na téma „Selekce příznaků pomocí nekorelovaných charakteristik“ jsem vypracoval samostatně pod vedením vedoucího bakalářské práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené bakalářské práce dále prohlašuji, že v souvislosti s vytvořením této bakalářské práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a/nebo majetkových a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon), ve znění pozdějších předpisů, včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

Brno ....

(podpis autora)

PODĚKOVÁNÍ

Na tomto místě bych rád poděkoval Ing. Petru Honzíkovi, Ph.D. za cenné rady a připomínky, které mi byly poskytnuty během vytváření této práce.

Brno ....

(podpis autora)

OBSAH

ÚVOD

Technologický pokrok stále přináší zvyšování výkonu výpočetních zařízení. Vedle narůstajícího výkonu jednotlivých zařízení masivně roste také množství vytvořených dat. Nejmarkantnějším příkladem je asi World Wide Web, jehož velikost se dá pouze odhadovat. Pro zajímavost lze uvést, že v roce 2008 na oficiálním blogu [1] společnosti Google dvojice inženýrů zveřejnila výsledek, podle kterého byla pokořena hranice 1 bilionu objevených jedinečných URL adres.

Obrovské rozsahy dat není možné zpracovat manuálně, je zapotřebí využít výpočetní techniku. Ke zpracování dat se běžně používají statistické metody, ale také metody strojového učení, případně kombinace obou[1]. Datové záznamy jsou často charakterizovány množinou *příznaků* (*atributů*). Těchto atributů však může být uvedeno v některých případech mnoho, řádově tisíce a víc. Ne vždy jsou všechny atributy užitečné pro další zpracování. Nadměrné množství příznaků navíc vynucuje práci s mnohadimenzionálním prostorem, který však klade vyšší požadavky na výpočetní výkon a z hlediska statistického způsobuje řídkost dat, což se může projevit nižší přesností algoritmů strojového učení[2].

Případy s obrovským množstvím příznaků lze řešit technikami spadajícími do kategorie *redukce dimenzí*. Tuto kategorii lze dále rozdělit na dvě skupiny, jak uvádí např. Cunningham [5]. Jednou skupinou je *transformace příznaků*, kdy dochází k vytvoření nové množiny příznaků na základě existující množiny pomocí určité transformace. Jinou skupinou je *selekce příznaků*, při níž dochází k výběru vhodné podmnožiny příznaků z dané množiny.

Prvním cílem v rámci této práce bylo porovnat metody provádějící selekci příznaků, konkrétně se jedná o metody, které se v angličtině označují jako *filter*. Dalším cílem bylo vytvořit novou metodu jako vhodnou kombinaci porovnávaných metod a srovnat ji s původními metodami.

Text práce je rozčleněn následovně. V první kapitole je čtenář seznámen se základními pojmy strojového učení a selekce příznaků. Obsaženo je také rozdělení metod selekce příznaků a stručná charakteristika některých filter metod. Další kapitola je věnována srovnání filter metod. Nejdříve jsou prozkoumány způsoby vyhodnocení v existujících pracích, následuje vlastní vyhodnocení. To je provedeno dvěma způsoby. V prvním případě se jedná o vyhodnocení pomocí algoritmu strojového učení, v druhém případě jsou metody porovnávány podle pořadí uspořádání příznaků. Výsledky tohoto vyhodnocení jsou využity v kapitole třetí. Zde je proveden návrh nové metody na základě kombinace již existujících metod. Vyzkoušeno je více kombinací,

[1] Tato práce se zaměřuje spíše na oblast strojového učení; tomu bude odpovídat i použitá terminologie.

[2] V literatuře se lze setkat s pojmem *curse of dimensionality*, který má původ v [2].

které jsou vyhodnoceny. Ve čtvrté kapitole jsou uvedena doporučení na výběr kombinace metod, případně na samostatnou filter metodu. V závěru je uvedeno shrnutí práce.

1 STROJOVÉ UČENÍ A SELEKCE PŘÍZNAKŮ

Strojové učení se běžně uvádí jako podoblast umělé inteligence. Definice strojového učení není vymezena zcela přesně, avšak jedna, o kterou se autoři v literatuře opírají asi nejčastěji, je uvedená v [15]. V této definice se hovoří o učení počítačového programu, což však není považováno za problém. Originální znění v angličtině je:

> *A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E.*[1]

Jako příklad použití této definice uvažujme problém rozpoznávání číslic z obrázků. Úloha T bude označovat proces přiřazení číslice počítačovým programem k obrázku. Míra výkonu P může být stanovena jako podíl počtu správně přiřazených (rozpoznaných) číslic ku celkovému počtu. Zkušenost E může označovat množinu obrázků se správnými přiřazeními, které jsme poskytli programu k naučení.

Použití metod strojového učení je vhodné především u problémů, kde není možné deterministicky zadat podobu vhodného řešení bez znalosti použitých dat, tedy kde je podoba řešení získána automatizovaně zpracováním vstupních dat. Vhodnými oblastmi jsou také ty, kde je výhodná generalizace nad existujícími daty, která může být následně využita pro predikci hodnot u nových případů. Příklady použití jsou např. klasifikace dokumentů (anebo jiných objektů) do určitých tříd, zpracování obrazu, zpracování přirozeného jazyka, doporučovací systémy (např. v internetových obchodech), analýza časových řad (např. ve finančnictví) atd.

Jako u příkladu rozpoznávání číslic výše, data jsou velmi často reprezentována jako množina příkladů. Tyto příklady jsou charakterizovány množinou atributů. V takových případech si můžeme data představit jako tabulku podobnou těm u relačních databází, kde každý řádek určuje jeden příklad a každý sloupec označuje určitý atribut. Jeden atribut může být navíc označen jako výstupní[2] (tedy jako závislý na ostatních atributech). V takovém případě se jedná o tzv. učení s učitelem. V opačném případě jde o tzv. učení bez učitele. Příklad s rozpoznáváním číslic patří do skupiny učení s učitelem, konkrétně se jedná o *klasifikaci*, ve které výstupní atribut nabývá hodnot z pevně dané konečné množiny nominálních hodnot. Vedle klasifikace existuje ještě *regrese*, kde výstupní atribut nabývá číselných hodnot. V rámci této práce byly využívány klasifikační algoritmy. Aby bylo možné porovnat různá nastavení těchto algoritmů, je zapotřebí použít nějakou metriku k ohodnocení

[1] Volně přeloženo: *Řekneme, že počítačový program se učí ze zkušenosti E vzhledem k nějaké třídě úloh T a míře P, pokud se jeho výkon v úlohách z T měřený mírou P zlepšuje pomocí zkušenosti E.*

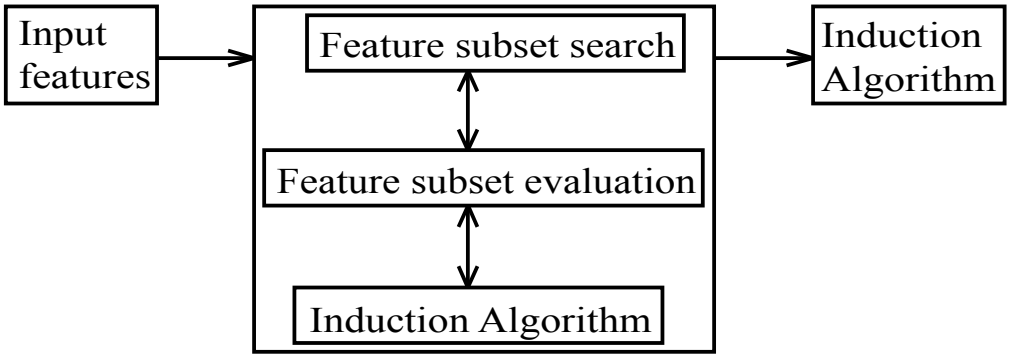
[2] Používá se také výraz *cílový atribut*.

výsledků. Základní metrikou klasifikačních algoritmů je *správnost*, která udává procento správně klasifikovaných příkladů. Dalšími metrikami jsou *přesnost*, udávající pro danou třídu podíl správně zařazených příkladů do této třídy ku celkovému počtu všech příkladů zařazených do této třídy, a *úplnost*, udávající pro danou třídu podíl správně zařazených příkladů do této třídy ku celkovému počtu příkladů, které by správně do této třídy zařazeny být měly. Za zmínku též stojí vyhodnocení pomocí křivky *ROC* a plochy pod touto křivkou označené jako *AUC*.

Vyhodnocení probíhá tak, že algoritmus se nejdříve naučí na *trénovacích* datech, kde má k dispozici správné přiřazení do tříd, následně je pak ohodnocen na *testovacích* datech, u kterých třídu predikuje. Aby nedocházelo ke zkreslení výsledků, trénovací i testovací data by měla být odlišná. Mohlo by totiž dojít k tomu, že algoritmus se dobře naučil na trénovacích datech, ale přitom má slabé výsledky při predikci na nových, dosud neviděných datech[3]. Tento způsob testování je dále rozšířen do používanější podoby nazvané jako *křížová validace*. V základní podobě této metody se stanoví počet složek n. Vstupní data se následně rozdělí na n částí. Jedna z těchto částí se odloží a zbytek se použije na učení. Následně je provedeno testování na odložené části. V dalším kroku se vybere na testování jiná složka a opět zbytek na učení. Takto se to opakuje celkem n-krát, takže pro testování jsou použity postupně všechny složky. Na závěr se výsledky z jednotlivých kroků zprůměrují.

Některá data mohou být charakterizována pouze několika atributy, někdy však jejich počet dosahuje řadově tisíců. Častými zástupci s velkým počtem atributů jsou data z oblasti biologie a chemie. Jak již bylo uvedeno dříve, velké množství atributů může představovat problém z hlediska výkonu algoritmů strojového učení. Stojí také za zmínku, že velký počet atributů (dimenzí) může zhoršovat generalizaci. Jedním z řešení je vybrat pouze určitou podmnožinu atributů, které budou použity v dalších fázích zpracování. Pro tyto metody se používá označení *selekce příznaků*. Zkoušet všechny kombinace příznaků a hledat jednu optimální je značně neefektivní, protože všech kombinací je 2^m, kde m je počet příznaků. Pro hledání vhodné množiny příznaků se tedy používají nejrůznější heuristiky, které jsou výpočetně méně náročné. Tyto techniky obvykle pracují v dopředném, nebo zpětném režimu. První jmenovaný začíná s prázdnou množinou a postupně atributy přidává, druhý pracuje na opačném principu, tedy začíná se všemi atributy, ze kterých pak odebírá. Počet vybíraných atributů je buď zadán pevně předem, nebo se průběžně provádí ohodnocení na nějakém učicím modelu a ve vhodném okamžiku se výběr ukončí. V případě dopředného výběru se nepřidávají další atributy, pokud nepřinášejí výrazné zlepšení, a v případě zpětného výběru se přestanou atributy ubírat, pokud by to ohodnocení výrazně zhoršilo.

[3]V takové situaci došlo u algoritmu k tzv. přeučení.



Obr. 1.1: Wrapper model[4]

Tři typy metod pro selekci příznaků jsou uvedeny v [7], a to wrapper, embedded a filter. V další části kapitoly budou tyto metody krátce představeny.

1.1 Wrapper metody

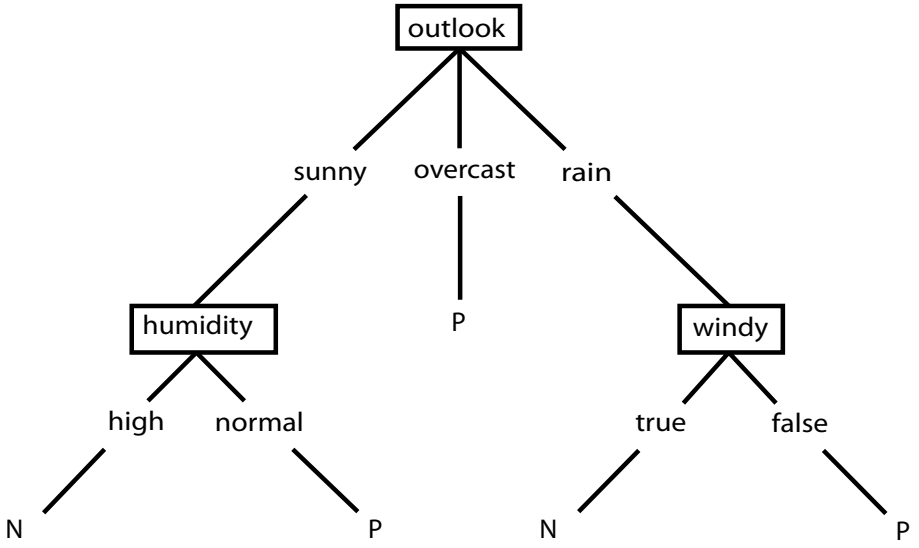
Metoda nazvaná *wrapper* je předložená v [10]. Název je dán tím, že tento model „obaluje“ použitý algoritmus strojového učení, který v modelu slouží k vyhodnocování vybrané podmnožiny atributů. Toto schéma je pro lepší představu uvedeno na Obr. 1.1. Tento algoritmus je brán jako „černá skříňka“, důležité jsou jen výsledky vyhodnocení, které jsou získány použitím křížové validace. Pro výběr příznaků jsou používány heuristiky *forward stepwise selection* a *backward stepwise elimination*, které přidávají nebo odebírají příznaky hladovým způsobem s tím, že jednou přidané příznaky mohou být zase odstraněny a odstraněné naopak přidány. Existují však i jiné strategie, např. v [11] je uvedeno použití technik *hill climbing* a *best-first search*.

1.2 Embedded metody

Jako *embedded* jsou souhrnně označeny takové metody, které jsou přímo součástí učícího algoritmu, a tedy neprobíhají zvlášť ve fázi předzpracování dat. Jako příklad je možné uvést rozhodovací stromy, které slouží pro klasifikaci. Ukázka takového stromu převzatého z [19] je na Obr. 1.2, který je sestaven z dat s atributy Outlook, Temperature, Humidity, Windy[5]. Klasifikace nových příkladů probíhá tak, že se začne od kořene stromu a podle hodnot atributů klasifikovaného příkladu se v každém

[4]Schéma je převzaté z [10]. Algoritmus má přívlastek *indukční* díky své schopnosti indukovat obecné závěry z konkrétních příkladů.

[5]Překlad: předpověď počasí, teplota, vlhkost, větrno.



Obr. 1.2: Příklad rozhodovacího stromu

uzlu vybírá potomek pro další sestup, dokud se nedospěje až do některého listového uzlu, který označuje výslednou třídu (zde P označuje obecně *pozitivní třídu* a N *negativní*). Strom je konstruován od kořene technikou rozděl a panuj. Při konstrukci se podle určitého kritéria (např. Information Gain, viz dále) vybírají nejvhodnější atributy na pozice uzlů. Jak lze vidět z ilustrace, strom nemusí použít ve všech větvích všechny atributy a zpravidla to ani nedělá, aby nedošlo k přeučení. V případě více atributů dokonce použije jen zlomek z nich. Tímto způsobem je implicitně provedena selekce příznaků.

1.3 Filter metody

Filter metody nejsou závislé, na rozdíl od wrapper metod, na výběru modelu strojového učení. Většinou pracují tak, že provedou pouhé seřazení atributů od nejlepšího po nejhorší. Vhodná podmnožina atributů je poté získána výběrem počátečního úseku tohoto seznamu. Výpočetně jsou méně náročné oproti wrapper metodám, takže jsou praktičtější na datech s vysokým počtem atributů. Na druhou stranu dávají obecně horší výsledky, viz např. [18].

Dále budou představeny vybrané filter metody na základě jejich dostupnosti v programu RapidMiner [14], který bude použit v experimentální části práce.

1.3.1 Information Gain

Information gain, neboli *informační zisk*, není přímo metoda, ale spíše míra, která provádí ohodnocení atributů. Rozšířila se hlavně díky algoritmu ID3 pro tvorbu rozhodovacích stromů od R. Quinlana [19]. Tato míra, nebo také charakteristika atributu staví na poznatcích z oblasti teorie informace. Klíčovou roli zde hraje *entropie*, která vyjadřuje míru neuspořádanosti.

Hodnota entropie je definována pro náhodnou proměnnou (v našem případě pro atribut) X jako funkce s předpisem

$$H(X) = -\sum_i P(x_i) \log_2(P(x_i)). \tag{1.1}$$

Pro proměnnou se dvěma hodnotami je entropie znázorněna na Obr. 1.3. Vodorovná osa v grafu udává relativní četnost jedné hodnoty (četnost druhé je doplněk do 1), vertikální osa hodnotu entropie. Jak lze vidět, nejvyšší hodnota entropie je při rovnoměrném rozložení hodnot.

Dále je definována entropie proměnné Y za předpokladu pozorování hodnot jiné proměnné X jako

$$H(Y|X) = -\sum_j P(x_j) \sum_i P(y_i|x_j) \log_2(P(y_i|x_j)). \tag{1.2}$$

Informační zisk se poté spočítá jako redukce entropie cílového atributu Y za předpokladu volby X

$$IG(Y;X) = H(Y) - H(Y|X). \tag{1.3}$$

Informační zisk je symetrická metrika, neboť platí

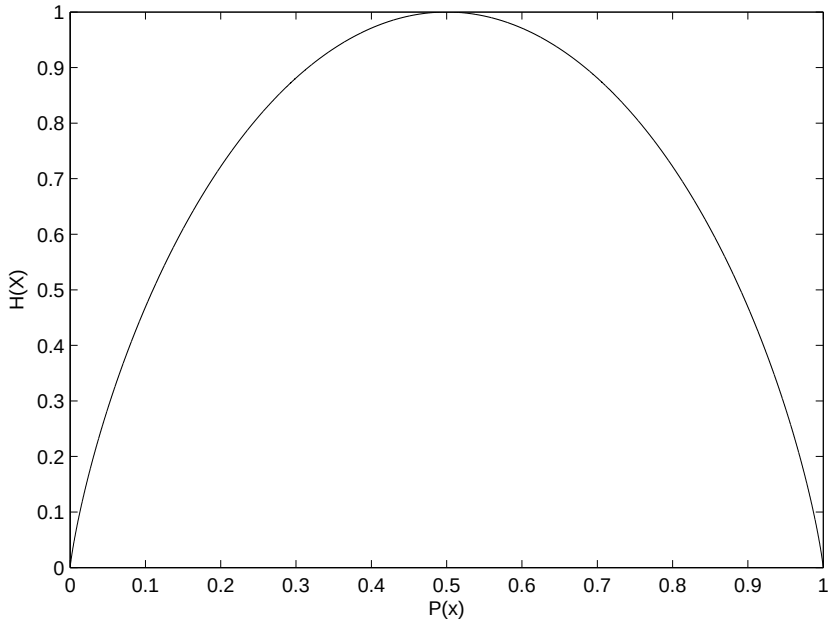
$$IG(Y;X) = H(Y) - H(Y|X) = H(X) - H(X|Y) = IG(X;Y). \tag{1.4}$$

Nejlepší atribut je ten s nejvyšším informačním ziskem. Ten totiž udává, jak dobře hodnoty atributu rozdělují příklady do tříd. Nejvyšší informační zisk budou mít ty atributy, pro jejichž každou hodnotu existuje pokud možno jen jedna třída (entropie je blízká 0).

Pro selekci příznaků je každému atributu přiřazena hodnota informačního zisku a podle toho jsou příznaky setříděny.

1.3.2 Information Gain Ratio

Předchozí míra má však nevýhodu v tom, že nebere v úvahu počet hodnot atributů. Pokud by nějaký atribut měl pro každý příklad jedinečnou hodnotu (např.



Obr. 1.3: Entropie

identifikátor příkladu), informační zisk by nabýval maximální možné hodnoty. Takový atribut je však nevhodný pro klasifikaci, neboť neumožňuje příliš generalizovat. Upravenou verzí informačního zisku je *poměrný informační zisk* (*information gain ratio*), který uvažuje i počet hodnot atributů přidáním entropie zkoumaného atributu do původního vztahu[6]:

$$IGR(Y;X) = \frac{IG(Y;X)}{H(X)}. \tag{1.5}$$

Tato metoda patří mezi normalizované varianty informačního zisku, tzn., že nabývá hodnot mezi 0 a 1.

1.3.3 Symmetrical Uncertainty

Symmetrical uncertainty je další míra, která se snaží jistým způsobem odstranit nedostatky informačního zisku:

$$SU(Y;X) = \frac{2IG(Y;X)}{H(Y) + H(X)}, \tag{1.6}$$

kde IG je informační zisk a H entropie podle vztahů uvedených dříve. Jedná se také o normalizovanou variantu informačního zisku jako v případě IGR, avšak tato varianta je symetrická. Vyšší hodnotě SU je přikládána vyšší váha jako v předchozích případech. Vzorce pro SU, IG a IGR byly převzaty z [8] a [3].

[6]V české literatuře, např. v [3], je možné se setkat s označením *větvení* pro výraz ve jmenovateli.

1.3.4 Relief

Zcela jiný způsob ohodnocení atributů přináší algoritmus Relief navržený v článku [9]. Tato první verze algoritmu pracuje s nominálními a numerickými atributy a je omezena pouze na data se dvěma třídami cílového atributu. Nová vylepšení jsou navržena a srovnána v [12]. Z těchto variant přináší nejvíce rozšíření a v testech vychází nejlíp verze označená ReliefF. Ta přináší oproti původní verzi schopnost pracovat se šumem v datech, chybějícími atributy a především s více třídami. Pseudokód algoritmu je následující:

```
1  set all weights W[A] := 0.0;
2  for i := 1 to m do
3      begin
4          randomly select an instance R;
5          find nearest hits H and nearest misses M;
6          for A := 1 to all_attributes do
7              W[A] := W[A] − diff(A, R, H)/m
                           + Σ_{C≠class(R)} [P(C) × diff(A, R, M(C))]/m;
8          end
9      end
10 end
```

Algoritmus 1: ReliefF

V poli W jsou ukládány váhy označující relevanci jednotlivých atributů. Iniciálně se nastaví tyto váhy na nulu. Parametrem m je zvolena velikost vzorku. Tím je i zadáno, kolikrát se provede vnější cyklus. V rámci jedné iterace se provede nejdříve náhodný výběr příkladu R z trénovacích dat. Dále se nalezne k nejbližších sousedů[7] ze stejné třídy jako je R, které uloží do H jako *nearest hits*, a k nejbližších sousedů pro ostatní třídy, které uloží do M jako *nearest misses*. Pro výběr nejbližších sousedů je použita metrika $diff$, viz dále. Nyní ve vnitřním cyklu aktualizuje váhu pro každý atribut. Metrika $diff(A, X, Y)$ počítá normalizovanou vzdálenost mezi příklady X a Y pro atribut A. Při použití k nejbližších sousedů se počítá průměrná vzdálenost. Metrika $diff$ umí počítat i s chybějícími hodnotami, detaily viz [12]. Od aktuální váhy se odečítá vzdálenost příkladu od *nearest hits* a přičítá vzdálenost od *nearest misses*. Obě hodnoty jsou navíc normalizovány parametrem m, takže výsledná váha je v intervalu [-1, 1].

Idea algoritmu je taková, že pro relevantní atributy bude vzdálenost od *nearest hits* průměrně menší než pro *nearest misses* a tedy výsledná váha se bude blížit k 1.

[7] V původním algoritmu to byl jeden soused, k sousedů zde pomůže odstranit problém se šumem.

Naopak pro irelevantní atributy budou tyto vzdálenosti celkem podobné a výsledná váha se bude pohybovat kolem 0, případně i v záporných hodnotách.

1.3.5 χ^2 test

Pro testování nezávislosti dvou náhodných veličin X a Y je možné použít χ^2 test [4]. Uvažujme, že četnosti obou veličin jsou dány kontingenční tabulkou:

	y	$y_{[1]} \dots y_{[s]}$	$n_{j.}$
x	n_{jk}		
$x_{[1]}$		$n_{11} \dots n_{1s}$	$n_{1.}$
$\vdots$		$\dots\dots\dots$	$\dots$
$x_{[r]}$		$n_{r1} \dots n_{rs}$	$n_{r.}$
$n_{.k}$		$n_{.1} \dots n_{.s}$	n

Testuje se nulová hypotéza H_0, která předpokládá nezávislost X a Y. Použije se testová charakteristika:

$$\chi^2 = \sum_{j=1}^{r} \sum_{k=1}^{s} \frac{(n_{jk} - \frac{n_{j.}n_{.k}}{n})^2}{\frac{n_{j.}n_{.k}}{n}}. \tag{1.7}$$

Pokud platí H_0, χ^2 se řídí rozložením $\chi^2((r-1)(s-1))$. H_0 je zamítnuta na asympt. hladině významnosti α, pokud $\chi^2 \geq \chi^2_{1-\alpha}((r-1)(s-1))$.

Čím vyšší hodnota statistiky, tím lepší důkaz proti platnosti nulové hypotézy. Pro použití k selekci příznaků provedeme test na nezávislost mezi každým atributem a cílovým atributem. Atributy lze tedy seřadit podle hodnot této statistiky, přičemž vyšší hodnota označuje lepší atribut. Použití této metody je možné nalézt např. v práci [13].

2 SROVNÁNÍ FILTER METOD

Jedním z cílů této práce je i porovnání *filter* metod, přičemž pozornost bude věnována primárně pěti metodám zmíněným v předchozí kapitole. Charakteristické rysy těchto metod již byly zmíněny, v této kapitole budou porovnány výsledky učicích algoritmů při použití atributů vybraných těmito metodami a také seřazení těchto atributů. Nejdříve budou prozkoumány výsledky srovnávacích testů v již existujících pracích, následovat budou dva způsoby vlastního porovnání.

2.1 Srovnání v literatuře

Všech pět zmíněných metod je porovnáno v [16]. K ohodnocení na dvou datových souborech byly použity klasifikační algoritmy IB1, Naive Bayes, C4.5 a RBF network. Použité datové soubory čítají 20, resp. 14, příznaků. Všechny tyto příznaky byly nejdříve seřazeny každou metodou a vyhodnocení poté proběhlo pro všechny možné výběry n nejlepších atributů. Jako metrika vyhodnocení byla použita správnost odhadnutá pomocí křížové validace. Závěry vyhodnocení jsou takové, že v případně jednoho datového souboru je dosaženo zhruba stejných výsledků u všech metod, v případě druhého souboru má o něco lepší výsledky metoda Relief při použití Naive Bayes a RBF network.

Další výsledky přináší [20]. V této práci je prezentována nová metoda založená na kombinaci metod filter a wrapper, nicméně je uvedeno i vyhodnocení pro čisté filter metody. Konkrétně jde o poměrný informační zisk, χ^2 a ReliefF. Na osmi datových souborech bylo provedeno ohodnocení pomocí klasifikačního algoritmu 1NN využitím křížové validace. Opět byly vyzkoušeny různé počty vybraných nejlepších atributů, ve výsledku jsou však uvedeny hodnoty správnosti jen nejlepší případy pro každou metodu a datový soubor. V jednom případě měl poměrný informační zisk se správností přibližně 90 % lepší výsledek oproti 87 a 83 % metod χ^2 a Relief. V jiném případě Relief a χ^2 svými 65 % překonaly poměrný informační zisk asi o 6 %. Avšak v dalších šesti případech byly výsledky velmi podobné pro všechny tři metody.

Rozsáhlé testování dvanácti různých metod pro selekci příznaků je popsáno v [6]. Z těchto metod však pro tuto práci byly podstatné pouze dvě, informační zisk a χ^2. Testování bylo provedeno na textových datech určených ke klasifikaci pomocí učicího algoritmu support vector machine. Kromě správnosti byla měřena ještě přesnost, úplnost a F-míra. Počty vybraných atributů byly z rozsahu 10 až 2000. Ve všech čtyřech metrikách informační zisk překonával χ^2.

2.2 Experimentální srovnání na modelu 1NN

Výběr množiny atributů s jejím ohodnocením byl proveden v prostředí RapidMiner [14]. RapidMiner patří mezi nejrozšířenější open-source systémy pro strojové učení, analýzu dat a další zpracování dat. Vedle neplacené verze existují i další, které doplňují funkcionalitu a podporu. Tato aplikace byla vybrána, neboť poskytuje většinu potřebných nástrojů pro tuto práci a její grafické rozhraní je celkem intuitivní na ovládání. Efektivitu práce navíc zvyšuje možnost sestavit schéma celého procesu z jednodušších bloků, které se označují jako *operátory* a kterých je k dispozici více než 500.

Tato část vyhodnocení probíhala na 10 datových souborech, jejichž základní údaje jsou uvedeny v tab. 2.1.

ID	Název	Počet příznaků	Počet tříd	Počet příkladů
1	ISOLET	617	26	7797
2	SECOM	590	2	1567
3	Madelon	500	2	2000
4	Musk (Version 2)	166	2	6598
5	Multiple Features	649	10	2000
6	optdigits	64	10	3823
7	Statlog (Landsat Satellite)	37	6	6435
8	Scene Classification	295	6	1137
9	SensIT Vehicle	101	3	15764*
10	Spectrometer	102	5	509

Tab. 2.1: Datové soubory pro vyhodnocení. * zde označuje z původní velikosti pětinu, která byla pro účely vyhodnocení postačující; data byla vybrána se zachováním rozložení tříd

Pro tuto část experimentu byla vybrána data s desítkami až stovkami příznaků. Všechna data jsou určena pro klasifikaci. Základním zdrojem pro tato data je repozitář UCI[1] pro strojové učení. Data byla upravena a převedena jednotně do vhodnějšího formátu pro účely automatizace zpracování pomocí RapidMiner.

K vyhodnocení vybraných množin atributů jednotlivými metodami byl zvolen algoritmus 1NN. Jedná se o instanci klasifikačního algoritmu k-NN (k-nearest neighbor), který patří do skupiny *instance-based learning*. Tento druh algoritmu nevytváří učením generalizovaný model, ale nové příklady jsou klasifikovány na základě porovnání s příklady v trénovací množině. V případě 1NN se volí třída na základě jednoho nejbližšího souseda. Ten je vybrán použitím smíšené Eukleidovské metriky. V případě numerických hodnot používá tato metrika stejný vzorec pro výpočet

[1] http://archive.ics.uci.edu/ml/

vzdálenosti jako základní verze, tedy

$$D(X,Y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}, \tag{2.1}$$

navíc však umí pracovat i s nominálními atributy, u kterých je vzdálenost rovna 1 (resp. 0), pokud se hodnoty rovnají (resp. nerovnají). Před vstupem do algoritmu jsou numerické atributy normalizovány do normálního rozložení se střední hodnotou 0 a rozptylem 1.

Kritériem hodnocení na zvolených podmnožinách příznaků byla správnost algoritmu. K vyhodnocení byla použita křížová validace s 10 složkami. Data byla do jednotlivých složek rozdělena se zachováním distribuce tříd.

Vývojový diagram vyhodnocení je uveden na obr. 2.1. Podle tohoto diagramu jsou v RapidMiner vytvořeny jednotlivé procesy. Ukázku procesů z RapidMiner lze nalézt v příloze A.

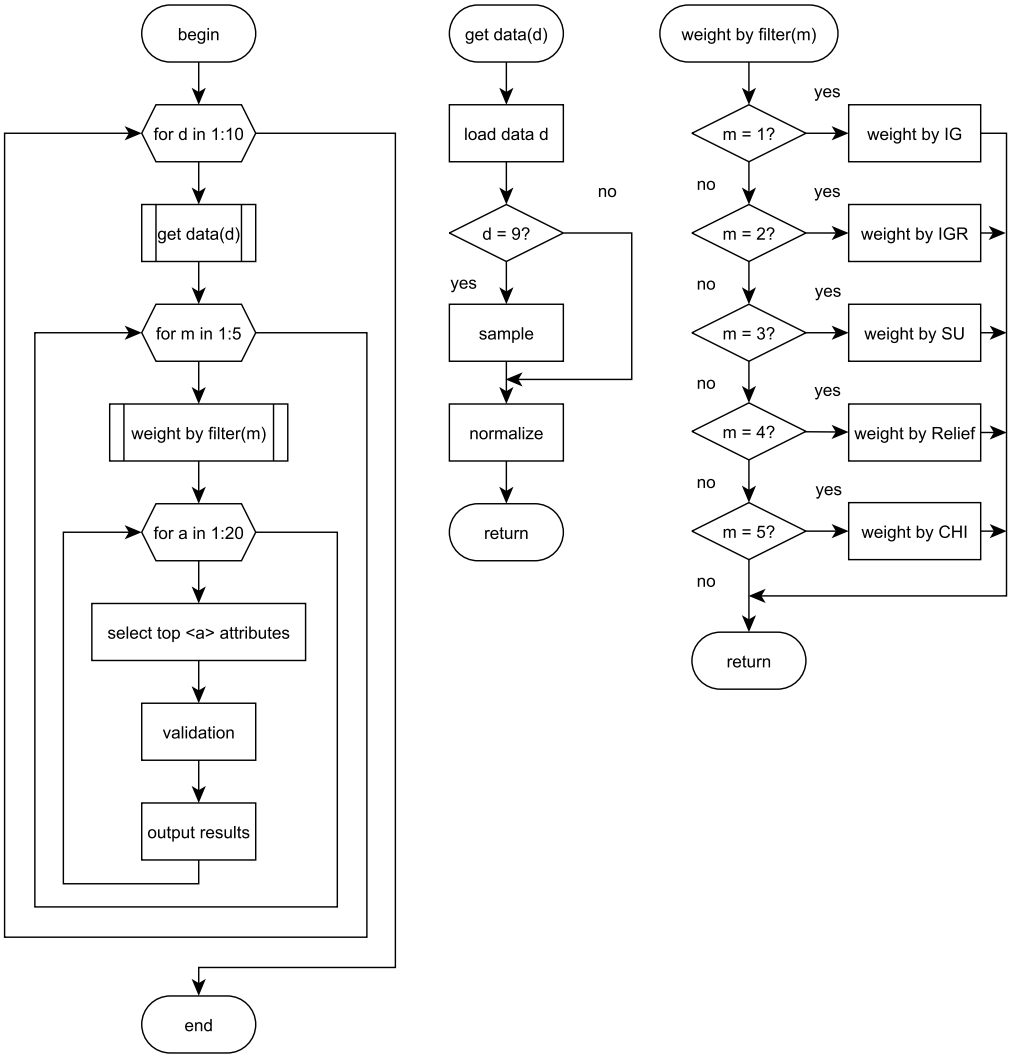
Iterace přes všechny datové soubory je uvedena ve vnějším cyklu, aby nemuselo docházet k opakovanému čtení stejného souboru z pomalého pevného disku. Pokud je načten 9. datový soubor (SensIT Vehicle), použije se jen pětina vzorku, jak je uvedeno taky v tabulce 2.1. U každého datového souboru dochází k normalizaci atributů, jak bylo uvedeno výše. Je to z toho důvodu, aby z použitých atributů přispívaly všechny stejným dílem ke klasifikaci pomocí 1NN. Tyto kroky jsou na obrázku uvedeny jako proces *get data(d)*.

Na každém datovém souboru jsou atributy seřazeny pomocí všech pěti filter metod. Přesněji řečeno, seřazeny jsou podle vah určených jednotlivými metodami, proto je krok označen jako *vážení* (weight).

V nejvnitřnějším cyklu je poté postupně vybíráno 1 až 20 nejlepších atributů podle vytvořeného řazení. Na každém výběru je pomocí křížové validace vyhodnocen model 1NN. Získaných 20 hodnot správnosti je nakonec uloženo do souboru. Takových souborů je tedy vytvořeno celkem 50. Pro souhrnné zpracování byly tyto soubory zkombinovány do jednoho souboru pomocí jednoduchého skriptu napsaného v jazyce Python[2].

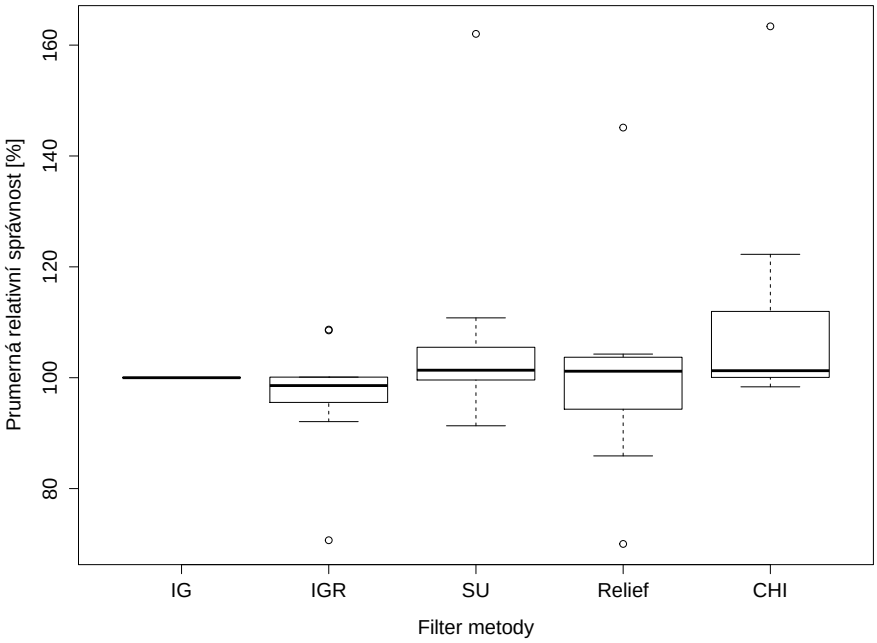
Cílem bylo srovnat jednotlivé metody mezi sebou. K tomu bylo potřeba zvolit vhodný způsob srovnání na základě získaných správností. Správnost je míra udávaná v procentech a její doplněk udává chybu klasifikace. Pro různě velké množiny zvolených atributů se správnost mohla lišit i o desítky procent. Pouhé rozdíly mezi hodnotami správnosti u jednotlivých metod nejsou příliš vypovídající, neboť rozdíl 5 % u 90% hranice představuje daleko menší relativní chybu než třeba u 10% hranice. Tento problém byl vyřešen zvolením jedné metody jako etalonu. Tou metodou

[2]http://www.python.org/



Obr. 2.1: Vyhodnocení filter metod

je informační zisk, neboť je velmi rozšířený a dá se považovat za jednu ze základních metod. Pro každý datový soubor a každý zvolený počet vybraných atributů byla určena relativní správnost všech metod vzhledem ke správnosti informačního zisku. Např. pokud byla správnost metody IG 65 % a IGR 68 % pro určitý datový soubor a počet vybraných atributů, pak relativní správnost IG byla 100 % a IGR přibližně 105 %. Pro každý datový soubor a filter metodu byl nakonec vypočten průměr z těchto relativních správností. Tyto výpočty byly provedeny v programu MS Excel. Výsledky pro jednotlivé metody jsou uvedeny krabicovými diagramy na obr. 2.2. Mediány metod se příliš neliší, v průměru však nejlépe vychází χ^2 a Symmetrical Uncertainty. Postupně pak následuje Relief, IG a IGR.



Obr. 2.2: Výsledky na datech s větším počtem příznaků

2.3 Stanovení korelační matice metod

Vedle vyhodnocení správnosti po aplikaci jednotlivých metod byla ještě zkoumána pořadí seřazených atributů. Pro tento účel byla použita data s menším počtem atributů. Pro úplnost byla změřena i průměrná relativní správnost pro každou z metod. Přehled těchto datových souborů dává tab.2.2.

ID	Název	Počet příznaků	Počet tříd	Počet příkladů
1	Letter Recognition	16	26	20000
2	Waveform Database Generator	21	3	5000
3	Parkinsons	23	2	197
4	Cardiotocography	23	10	2126
5	Steel Plates Faults	27	7	1941
6	Ionosphere	34	2	351
7	Statlog (Landsat Satellite)	36	6	6435
8	Spambase	57	2	4601

Tab. 2.2: Datové soubory použité pro výpočet korelací

Pro každý datový soubor a každou metodu byl z programu RapidMiner do souboru vypsán seznam atributů ve vypočteném pořadí. V tomto seznamu byly zahrnuty všechny atributy. Každý atribut má též přiřazenu váhu, která v podstatě určuje i pořadí atributu. Získané výstupní soubory byly opět zpracovány pomocí jednoduchého skriptu v jazyce Python. Ten prošel vytvořené seznamy atributů a uspořádal je podle jejich původního pořadí. Tím vznikly seznamy vah uspořádané pro všechny metody stejným způsobem podle názvu atributů. Pro každý datový soubor byly tyto seznamy vah zkombinovány a uloženy do samostatného souboru.

Cílem bylo porovnat pořadí atributů vytvořená jednotlivými metodami. K tomuto účelu se používají testy nezávislosti ordinálních veličin. Pro porovnání dvou ordinálních veličin se používá Spearmanův koeficient pořadové korelace, viz např. [4]. Seznamy vah, vytvořené výše popsaným způsobem, je právě možné použít k výpočtu tohoto koeficientu.

Nechť $(X_1, Y_1), ..., (X_n, Y_n)$ je dvourozměrný náhodný výběr a R_i je pořadí náhodné veličiny X_i a Q_i pořadí náhodné veličiny Y_i. Pak Spearmanův koeficient pořadové korelace je dán výrazem

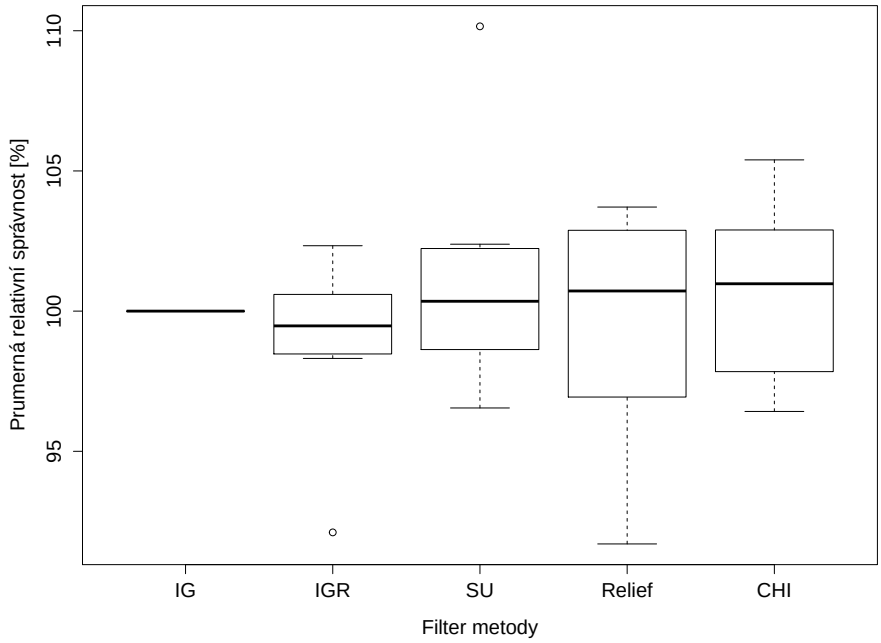
$$r_s = 1 - \frac{6}{n(n^2-1)} \sum_{i=1}^{n} (R_i - Q_i)^2.$$

Koeficient nabývá hodnot z intervalu [-1, 1]. Čím bližší je hodnotě 1 (resp. -1), tím silnější je přímá (resp. nepřímá) pořadová závislost veličin X, Y. Hodnoty blízké 0 vyjadřují závislost slabou nebo žádnou.

Dosazením seznamů vah za veličiny R a Q je tak možné vypočítat korelaci pořadí atributů dvou filter metod. Pomocí skriptovacího jazyka R [17] byly vypočteny korelace pro všechy dvojice metod a pro každý datový soubor. Data z těchto korelačních matic bylo potřeba jistým způsobem sloučit. K tomu navíc byly hledány nekorelované charakteristiky (tzn., že nás nezajímají metody, mezi nimiž existuje pozitivní nebo negativní korelace). Na základě těchto požadavků byl vypočten průměr z absolutních hodnot jednotlivých korelací. Výsledná korelační matice je uvedena v tab 2.3.

	IG	IGR	SU	$RELIEF$	CHI
IG	$1,00$	$0,64$	$0,77$	$0,48$	$0,65$
IGR	$0,64$	$1,00$	$0,59$	$0,34$	$0,60$
SU	$0,77$	$0,59$	$1,00$	$0,57$	$0,92$
$RELIEF$	$0,48$	$0,34$	$0,57$	$1,00$	$0,59$
CHI	$0,65$	$0,60$	$0,92$	$0,59$	$1,00$

Tab. 2.3: Souhrnná korelační matice



Obr. 2.3: Výsledky na datech s menším počtem příznaků

Pro srovnání je na těchto datech provedeno stejné vyhodnocení jako v předchozí části, tzn. použitím 1NN s křížovou validací jako je na obr. 2.1, nyní však se všemi možnými velikostmi množin vybraných atributů. Výsledky znázorňuje obr. 2.3. Jak lze vidět, na těchto datech se výsledky jednotlivých metod neliší příliš ve srovnání s předchozím případem. To může být dáno i tím, že byly vybírány množiny atributů o všech možných velikostech a navíc celkový počet atributů nebyl příliš vysoký. Tzn., že z toho malého počtu výběrů mohly poslední výběry o velikostech $n, n-1, n-2, ...$ znatelně zkreslit výsledky, neboť takové množiny vybraných atributů se už příliš neliší a výsledky jsou téměř shodné.

3 NÁVRH NOVÉ METODY

Návrh nové metody se zaměřuje na vytvoření metody, která kombinuje již existující filter metody. Myšlenka je taková, že jednotlivé metody se obecně mohou odlišovat na množinách vybraných atributů, resp. na seřazení atributů podle použitých metrik. Vhodnou kombinací určitých metod by tak mohlo být dosaženo zajímavých výsledků. Cílem je tedy vybrat vhodné metody, určitý počet nejlepších atributů na základě ohodnocení těmito metodami a také způsob výběru těchto atributů. Co se týče posledního bodu, při výběru atributů je nutné brát v úvahu pořadí metod. To z toho důvodu, že pro některé metody mohou být množiny k nejlepších atributů podobné. Např. nechť je použita kombinace jistých dvou metod pro výběr 2 atributů, přičemž podle každé metody se vybírá jeden atribut. Dále nechť tyto metody vytvořily uspořádání atributů od nejlepšího následujícím způsobem: $x, y, ...$ v případě první metody a $x, z, ...$ v případě druhé metody. Pokud by se začínalo první metodou, došlo by výběru x, z. Avšak pokud by první výběr učinila druhá metoda, vybrány by byly atributy x, y.

V této kapitole jsou provedeny experimenty na různých kombinacích. Všech možných způsobů, jak zkombinovat filter metody a provést výběr atributů, je však příliš mnoho. Pokud je např. počet vybíraných atributů k, celkový počet atributů datového souboru M a $M \gg k$, pak je možné předpokládat, že může nastat situace, kdy množiny s k nejlepšími atributy jsou pro jednotlivé metody navzájem disjunktní. V takovém případě při výběru n_1, n_2, n_3, n_4, n_5 nejlepších atributů podle jednotlivých metod ($n_i \geq 0$ a $\sum_{i=1}^{n} n_i = k$) je možné provést $\binom{n+k-1}{k}$ různých výběrů, kde $n = 5$[1]. Např. výběr 20 atributů bude možný 10626 způsoby. Problém s mnoha kombinacemi je vyřešen tak, že jsou vybrány k vyzkoušení jen některé na základě výsledků z předchozí kapitoly.

Celkově bylo vyzkoušeno 128 způsobů, jak vybrat určitý počet atributů. Prvních 8 kombinačních metod je založeno na výsledcích správnosti. Zbývajících 120 metod provádí kombinace za využití korelační matice, vypočtené v kapitole 2.3. Jednotlivé způsoby kombinací budou podrobněji popsány v dalších částech kapitoly. V první řadě však bude popsána přípravná fáze.

3.1 Příprava experimentů

Aby bylo provedení experimentů rychlejší, byl pomocí RapidMiner vytvořen proces pro výpis ohodnocených atributů každého datového souboru podle každé z pěti

[1]Uvedený počet výběrů tedy předpokládá nejhorší situaci, kdy jsou nejlepší atributy u jednotlivých metod odlišné. V reálných případech jsou však tyto množiny atributů podobné a počet různých výběrů je tedy menší.

filter metod. Tak je možné u každé kombinační metody využít již dopředu vypočítané uspořádání atributů a nemusí se počítat opakovaně. Tento proces je modifikací procesu uvedeného na obr. 2.1. Rozdíl je v tom, že po zvážení atributů (weight by filter) dochází už jenom k výpisu tohoto váženého seznamu atributů. RapidMiner umožňuje výpis vah atributů do souborů s formátem xml. Ukázka části takového souboru xml je na obr. 3.1.

```
<?xml version="1.0" encoding="windows-1250"?>
<attributeweights version="5.3">
    <weight name="isolet.data (461)" value="1.0"/>
    <weight name="isolet.data (460)" value="0.965392768942813"/>
    <weight name="isolet.data (459)" value="0.9377905296930781"/>
    <weight name="isolet.data (462)" value="0.9272516958814719"/>
    ...
    <weight name="isolet.data (369)" value="0.05659061914345907"/>
    <weight name="isolet.data (583)" value="0.05526301885746221"/>
    <weight name="isolet.data (288)" value="0.05193220416606998"/>
    <weight name="isolet.data (473)" value="0.0"/>
</attributeweights>
```

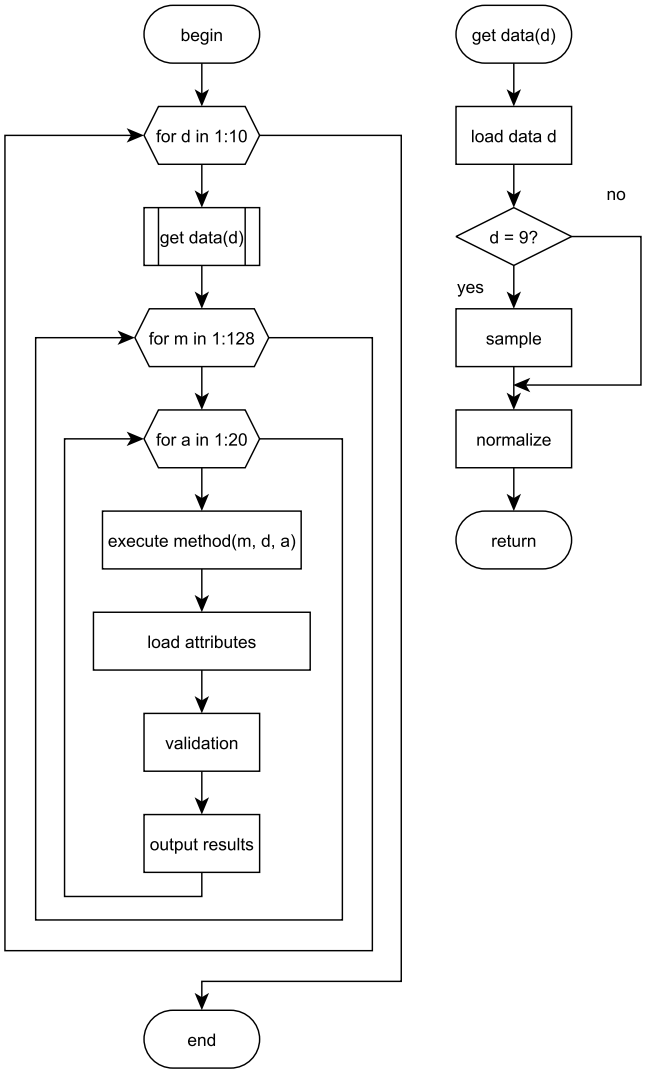
Obr. 3.1: Příklad souboru s atributy a jejich váhami

Tento soubor je vytvořen z dat ISOLET (viz tab. 2.1), použitá metoda byla IG. Pro každý atribut je přítomen jeden prvek *weight*, který v parametru *name* obsahuje jméno atributu a v parametru *value* váhu atributu. Jak lze vidět z obrázku, váhy mají hodnotu od 0 do 1, přičemž vyšší číslo označuje lepší atribut.

Co se týče samotných experimentů, opět probíhá vyhodnocení správnosti stejně jako v části 2.2. V RapidMiner byl však vytvořen trochu jiný proces, který je pomocí vývojového diagramu zachycen na obr. 3.2. Nyní je hodnoceno 128 metod, které jsou spouštěny v části *execute method*. Každá metoda načte potřebné soubory s váhami a poté určitým způsobem vytvoří nový soubor s váhami, který obsahuje pouze stanovený počet atributů. Tento nový soubor je načten v procesu a na základě této podskupiny atributů dojde k vyhodnocení. Provedení kombinační metody není možné v tomto procesu uskutečnit vně cyklus pro výběr počtu atributů. To z toho důvodu, že princip je zde trochu jiný, než u klasických filter metod. U nich jsou totiž seřazeny všechny atributy a pak je možné vybrat jakýkoliv počet. Kombinovaná metoda však sestaví atributy podle požadovaného celkového počtu. Není tedy možné udělat jedno uspořádání na všech atributech a z toho pak brát různé počty atributů[2].

Jednotlivé metody jsou implementovány v jazyce Python a jsou volány pomocí funkce, která je v kroku *execute method* externě spouštěna. Ke spuštění je nutné mít

[2]Technicky vzato to možné je, avšak do takového výběru se nemusí promítnout fakt, že byla použita kombinace metod.

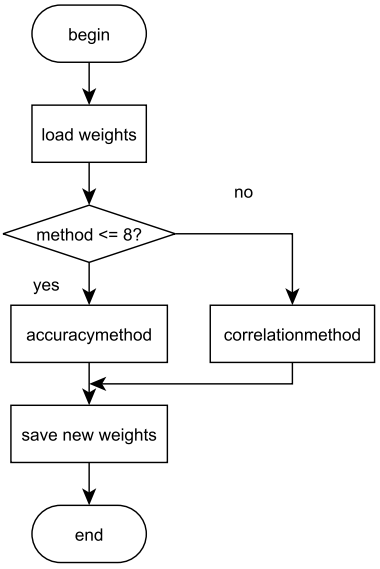


Obr. 3.2: Vyhodnocení kombinačních metod

Python nainstalován. Přesněji řečeno, v RapidMiner je nutné zadat příkaz, který je následně proveden v konzoli operačního systému. Nastavení tohoto operátoru je tedy závislé na použitém OS. V další části kapitoly bude popsána zmíněná funkce i volané metody.

3.2 Kombinace na základě správnosti filter metod

Předtím, než je v externí funkci možné vytvořit vhodnou kombinaci atributů, je nutné načíst atributy a jejich váhy ze souborů. Teprve následně je spuštěna určitá kombinační metoda, která pracuje na základě zjištěných správností nebo korelace filter metod, a získaný seznam atributů je vypsán. Tyto kroky jsou zachyceny na



Obr. 3.3: Externí funkce pro spuštění kombinačních metod

obr. 3.3, přičemž metody na základě korelační matice budou popsány až v další podkapitole.

Funkce podle obr. 3.2 přebírá parametry s číslem kombinační metody, datového souboru a počtem požadovaných atributů. Podle zadaného čísla datového souboru jsou načteny soubory s váhami, které byly sestaveny podle pěti používaných filter metod. Jak již bylo zmíněno, každý soubor s váhami je ve formátu xml. Programovací jazyk Python však poskytuje knihovnu pro zpracování takového formátu. Pomocí této knihovny je načten celý soubor jakožto jeden řetězec znaků a převeden na stromovou strukturu, kterou je možné procházet pomocí jazyka XPath[3]. Použitím XPath je extrahován seznam všech prvků *weight*, z nichž jsou následně vybrány hodnoty parametrů *name*. Výsledkem je tedy seznam jmen atributů, který je uspořádán podle původního uspořádání v souboru, tzn. podle vah od nejlepšího.

Číslo kombinační metody určuje, jakým způsobem se budou atributy vybírat. Výběr na základě správnosti je možné provést osmi způsoby, jejichž odlišující znaky jsou uvedeny v tab. 3.1. Zkombinovat lze 2 až 5 filter metod. Vždy se vybírají nejlepší metody podle průměrné relativní správnosti, jak je uvedeno v kapitole 2.2. Např. kombinace č. 2 a 6 použije χ^2, SU a Relief. Další rozdíl je v tom, zda se atributy vybírají nejdříve od nejlepší metody, nebo od nejhorší. U kombinace č. 2 jsou vybírány nejdříve podle χ^2, u kombinace č. 6 nejdříve podle Relief. Při výběru samozřejmě dochází ke kontrole, zda uvažovaný atribut nebyl již do výběru přidán.

[3]Specifikace jazyka je spravována konsorciem W3C, viz např. `http://www.w3.org/TR/xpath20/`.

Číslo metody	Počet použitých filter metod	Výběr atributů
1	2	od nejlepší filter metody
2	3	
3	4	
4	5	
5	2	od nejhorší filter metody
6	3	
7	4	
8	5	

Tab. 3.1: Metody na základě výsledků správnosti

Poslední věc, kterou je nutné vyřešit, je počet atributů, který má být podle každé filter metody vybrán. Byl navržen způsob, který bere v úvahu výsledky průměrných správností. A to tak, že použité metody podle těchto správností uspořádá a podle každé metody vybere takový počet atributů, aby byl polovinou počtu u bezprostředně lepší metody. Tzn., že pokud budou vybírány atributy podle χ^2 a SU, pak podle χ^2 budou vybrány 2/3 atributů a podle SU 1/3. Pokud např. bude vybíráno podle χ^2, SU, Relief a IG, pak rozdělení atributů bude postupně 8/15, 4/15, 2/15 a 1/15.

Pseudokód celého algoritmu výběru atributů pro metody 1 až 8 je uveden jako alg. 2. Vstupem do algoritmu jsou jednak uspořádané seznamy atributů pro jednotlivé metody (*Attributes*), jednak počet požadovaných atributů v novém seznamu ($Total$). Další parametry jsou podle tab. 3.1, tedy počet kombinovaných filter metod ($NumberOfMethods$) a příznak určující, zda budou atributy přidávány od nejlepší metody ($StartWithBest$). Výstupem je seznam nových atributů ($NewAttributes$).

V prvním kroku jsou vypočteny relativní počty atributů, které budou jednotlivými metodami vybrány. Tento krok se v kódu nachází na řádcích 2 až 4. Např. pro dvě metody (m=2) budou tyto počty 2/3 a 1/3, pro tři metody 4/7, 2/7 a 1/7, atd. Tyto relativní počty jsou na dalším řádku pomocí funkce compute_distribution() převedeny na absolutní vzhledem k celkovému počtu Total.

Funkce pokračuje na 6. řádku tak, že seznam filter metod je uspořádán podle jejich průměrné správnosti a v toto pořadí je zaznamenáno v seznamu $Methods$. Vybráno je jen m nejlepších metod. Jelikož jsou v každé kombinaci alespoň 2 metody, bude tento seznam začínat vždy metodou χ^2 a SU vzhledem k jejich výsledkům. Pokud je zvolen výběr atributů od nejhorší metody, bude seznam $Methods$ obrácen, viz řádky 7 až 10. Stejně tak budou obráceny i počty v $Distr$, takže bude zachována korespondence mezi metodou a jejím počtem atributů. Z toho je zřejmé, že nejlepší metoda bude vždy vybírat největší počet atributů.

Vstup : Attributes, Total, NumberOfMethods, StartWithBest
Výstup: NewAttributes

```
1  m := NumberOfMethods;
2  for i := m to 1 do
3  |  RelDistr[i] := (2^(i-1))/(2^m-1);
4  end
5  Distr := compute_distribution(RelDistr, Total);
6  Methods := uspořádaný seznam m nejlepších filter metod;
7  if StartWithBest = false then
8  |  reverse(Distr);
9  |  reverse(Methods);
10 end
11 for i := 1 to m do
12 |  added := 0;
13 |  j := 0;
14 |  while added < Distr[i] do
15 |  |  attr := j-tý atribut podle metody Methods[i];
16 |  |  if attr ∉ NewAttributes then
17 |  |  |  přidej attr do NewAttributes;
18 |  |  |  added++;
19 |  |  end
20 |  |  j++;
21 |  end
22 end
```

Algoritmus 2: Kombinace atributů podle výsledků správnosti

Nakonec je v pseudokódu přítomen cyklus, uvnitř kterého jsou přidávány atributy. Tento cyklus proměnnou i určuje filter metodu, která je zrovna používána. Atributy jsou v cyklu, uvedeného na řádcích 14 až 21, vybírány podle dané metody do té doby, než je splněn požadavek na počet pro tuto metodu. Počet se udržuje v proměnné *added*. Do seznamu $NewAttributes$ se přidávají jen takové atributy, přesněji jejich jména, které ještě nebyly přidány jinou metodou.

Tímto způsobem je seznam atributů získán. Na závěr je nutné tento seznam vypsat do souboru, aby mohl být načten procesem v RapidMiner, kde výpočet pokračuje. Formát výstupního souboru byl zvolen stejný jako u vstupního. Jelikož jsou v $NewAttributes$ uložena pouze jména atributů, je potřeba doplnit hodnoty vah. V případě vstupního souboru nejsou mezi sousedními hodnotami vah obecně stejné rozdíly. U výstupního souboru však na konkrétních váhách nezáleží a navíc ani uspořádání atributů není podstatné, neboť se pro validaci v RapidMiner berou všechny tyto atributy. Z tohoto důvodu jsou atributy do souboru vypsány v získaném pořadí a jsou jim přiřazeny váhy rovnoměrně z intervalu od 1 do 0.

3.3 Využití korelační matice

Vedle zmíněných osmi metod je testováno dalších 120, které těží z hodnot výsledné korelační matice uvedené v kap. 2.3. Načtení a výpis atributů s váhami se provádí stejným způsobem, jako bylo právě popsáno. Budou zde tedy uvedeny pouze metody pro výběr atributů.

Přechozích osm metod bylo odlišeno dvěma parametry. Nyní jsou použity parametry čtyři; jejich účel bude patrnější z popisu algoritmu uvedeného později. Celkový počet požadovaných atributů se k nim nyní nepočítá. Prvním parametrem je *StartingMethod*, který určuje, kterou z pěti filter metod se bude začínat. Tento parametr nabývá hodnot 0 pro IG, 1 pro IGR, 2 pro SU, 3 pro Relief a 4 pro χ^2. Kterou filter metodou se bude pokračovat, je dáno jednak vypočtenou korelační maticí a jednak parametrem *ContinueWithMostCorrelated*, který svou pravdivostní hodnotou určuje, zda další použitá metoda bude ta, která s tou počáteční nejvíce koreluje, nebo ta, která s ní koreluje nejméně. Úkolem této práce bylo primárně vyzkoušet metody, které spolu nekorelují, z experimentálních důvodů jsou však zkoušeny i kombinace korelujících metod. Další parametr *SortBy* udává, zda se atributy budou vybírat podle počtu atributů (hodnota SmallestDistribution nebo LargestDistribution), jež se mají podle jednotlivých filter metod přidávat, případně zda je výběr dán pořadím metod určeného podle *ContinueWithMostCorrelated* (hodnota DontSort). Posledním parametrem je *NumberOfMethods*, který stejně jako v předchozím případě udává, kolik filter metod bude zkombinováno.

Počet kombinací parametrů je celkem 120, každá kombinace tedy určuje jednu z metod. Z procesu v RapidMiner je skriptu s metodami předáno pouze číslo kombinační metody. Pro přehlednost lze uvažovat, že toto číslo nabývá hodnot 0 až 119. Pak lze hodnoty všech čtyř parametrů jednoznačně určit podle postupu uvedeného jakožto algoritmus 3. Pro představu je přehled prvních 24 metod uveden v tabulce 3.2. Pokračování tabulky by se odlišovalo metodami ve sloupci StartingMethod.

Metody	Starting-Method	ContinueWith-MostCorrelated	SortBy	NumberOfMethods
0 – 3	IG	false	DontSort	2 – 5
4 – 7			SmallestDistribution	2 – 5
8 – 11			LargestDistribution	2 – 5
12 – 15		true	DontSort	2 – 5
16 – 19			SmallestDistribution	2 – 5
20 – 23			LargestDistribution	2 – 5

Tab. 3.2: Část seznamu metod založených na korelační matici

Vstup : Číslo kombinační metody M z rozsahu 0 až 119 (včetně)
Výstup: StartingMethod, ContinueWithMostCorrelated, SortBy, NumberOfMethods

```
1  StartingMethod = ⌊M/24⌋;
2  if M mod 24 > 11 then
3  |  ContinueWithMostCorrelated := true;
4  else
5  |  ContinueWithMostCorrelated := false;
6  end
7  if M mod 12 < 4 then
8  |  SortBy := DontSort;
9  else if M mod 12 < 8 then
10 |  SortBy := SmallestDistribution;
11 else
12 |  SortBy := LargestDistribution;
13 end
14 NumberOfMethods := (M mod 4) + 2;
```

Algoritmus 3: Určení atributů kombinačních metod

Po zjištění hodnot jednotlivých parametrů je spuštěn hlavní algoritmus pro výběr atributů. Pseudokód celého algoritmu je možné nalézt ve výpise 4. Atributy se budou vždy přidávat podle vybrané počáteční metody v parametru *StartingMethod* a podle několika dalších. Jak je uvedeno na řádku č. 2, všechny další metody jsou nejdříve uloženy do seznamu *Methods* v pořadí daném korelací s počáteční metodou. Např. pokud je počáteční metodou IG, pak podle tabulky 2.3 budou v tomto pořadí přidány: Relief, IGR, χ^2, SU. Pokud bylo zvoleno použití nejvíce korelujících metod, je tento seznam obrácen. V dalším kroku se z konce seznamu *Methods* odstraní tolik metod, aby i s počáteční metodou zůstal pouze požadovaný počet.

Sedmým řádkem začíná výpočet rozdělení, které bude určovat, kolik atributů bude podle které filter metody vybráno.[4] Na začátek seznamu *Distribution* je iniciálně přidána hodnota 1,0, která je určena pro počáteční metodu. Jak lze totiž vidět na řádku 15, tato metoda je na začátek *Methods* přidána. Namísto hodnoty 1,0 mohlo být zvoleno i jiné kladné číslo, neboť později budou všechna čísla normalizována na součet roven 1; tato první hodnota zde určuje jen počáteční bod, od kterého se budou další výpočty odvíjet. Druhá hodnota *Distribution* bude odrážet míru korelace druhé metody s počáteční. Konkrétně se použije rozdíl čísla 1 a hodnoty korelace, jak je uvedeno na řádku 9. Např. pokud v seznamu *Methods* byly metody IG a Relief, pak v *Distribution* budou hodnoty 1,0 a 0,52, protože podle

[4]Výpočty, které jsou dále uvedeny pro výběr atributů, je potřeba brát jen jako heuristiky, které byly vytvořeny spíše na základě intuice a nelze je brát jako korektní interpretaci korelačního koeficientu.

Vstup : Attributes, Total, CorrelationMatrix, StartingMethod, ContinueWithMostCorrelated, SortBy, NumberOfMethods
Výstup: NewAttributes

```
1  m := NumberOfMethods;
2  Methods := vzestupně uspořádaný seznam filter metod podle korelace se
   StartingMethod, vyjma StartingMethod;
3  if ContinueWithMostCorrelated = true then
4  |  reverse(Methods);
5  end
6  V Methods ponechej pouze prvních m − 1 metod;
7  Distribution := [1.0];
8  for i := 1 to m − 1 do
9  |  value := 1 - CorrelationMatrix[StartingMethod][Methods[i]];
10 |  for j := 1 to i − 1 do
11 |  |  value := value * (1 - CorrelationMatrix[Methods[i]][Methods[j]])
12 |  end
13 |  Přidej value do Distribution;
14 end
15 Na začátek Methods přidej StartingMethod;
16 if SortBy = SmallestDistribution then
17 |  Vzestupně uspořádej Distribution a podle toho i Methods;
18 else if SortBy = LargestDistribution then
19 |  Sestupně uspořádej Distribution a podle toho i Methods;
20 end
21 RelDistr := normalizovaný seznam Distribution;
22 Distr := compute_distribution(RelDistr, Total);
23 for i := 1 to m do
24 |  added := 0;
25 |  j := 0;
26 |  while added < Distr[i] do
27 |  |  attr := j-tý atribut podle metody Methods[i];
28 |  |  if attr ∉ NewAttributes then
29 |  |  |  přidej attr do NewAttributes;
30 |  |  |  added++;
31 |  |  end
32 |  |  j++;
33 |  end
34 end
```

Algoritmus 4: Kombinace atributů na základě korelační matice

tab. 2.3 je hodnota korelace Relief a IG 0,48. Intuitivně lze tato čísla brát tak, že pokud bylo podle IG vybráno např. 100 atributů, nebude už bráno 100 podle Relief, protože mezi těmito metodami je jistý stupeň korelace a vybrané množiny atributů by se mohly překrývat. Podle Relief se tudíž vybere menší počet, např. 52.

Pokud z požadovaného celkového počtu bude vybráno 100/152 atributů podle IG a 52/152 podle Relief, intuitivně bude takový výběr zajímavý, předpokládá-li se, že IG je považována za důležitější metodu. Tato heuristika pro určení počtu atributů k jednotlivým metodám pokračuje na řádcích 10 až 12 v případě použití třech a více filter metod. U třetí, čtvrté a páté metody je nutné do výpočtu zahrnout i další předcházející metody, nejen tedy počáteční. Rozšířením předchozího příkladu na 5 metod budou do *Distribution* k 1,0 a 0,52 dále přidány tyto tři hodnoty. Za filter metodu IGR to bude 0,36 * 0,66 = 0,2376, neboť s IG má hodnotu korelace 0,64 a s Relief 0,34. Další metodou vzhledem k nejmenší korelaci s IG je χ^2 a do *Distribution* bude přidána hodnota 0,35 * 0,41 * 0,40 = 0,0574. Poslední v tomto příkladě bude SU, za kterou se přidá 0,23 * 0,43 * 0,41 * 0,08 = 0,003.

Vytvořený seznam rozdělení může být dále uspořádán, a to buď vzestupně, nebo sestupně. Pokud dojde k uspořádání, jsou uspořádány zároveň i metody v *Methods*, aby související hodnoty obou seznamů byly ve výsledku opět na stejných pozicích. U příkladu výše vyšly hodnoty v sestupném pořadí. Avšak ne ve všech případech to musí takhle dopadnout. Mějme například počáteční metodu IG a tyto hodnoty korelací: s IGR 0,25, s SU 0,26 a s χ^2 0,27. Při výběru od nejmenší korelace by bylo zvoleno pořadí, v jakém jsou metody zmíněny. Pro IGR by se do *Distribution* přidala hodnota 0,75. Pokud by hodnota korelace IGR a SU byla 0,92, pak by do *Distribution* byla pro SU vložena hodnota 0,74 * 0,08 = 0,0592. Dále za předpokladu hodnoty korelace rovné 0,6 pro IGR a χ^2 a stejně tak pro SU a χ^2 by do Distribution byla pro χ^2 uložena hodnota 0,73 * 0,4 * 0,4 = 0,1168. Tento příklad ukazuje, že i když byly metody SU a χ^2 uspořádány vzestupně podle korelace, nemusí být výsledné hodnoty v seznamu *Distribution* uspořádány vůbec. Pokud by se postupovalo od nejvíce korelovaných metod, není uspořádání v *Distribution* také zaručeno; protipříklady by daly vytvořit obdobně.

V další části algoritmu je seznam *Distribution* normalizován, aby součet hodnot dával hodnotu 1. Poslední část je už stejná jako u předchozího algoritmu pro výběr atributů. Nejprve jsou vypočteny absolutní počty atributů vzhledem k požadovanému celkovému počtu *Total*. Nakonec jsou podle těchto počtů a podle uspořádání metod vybrány atributy, opět bez opakování.

3.4 Výsledky

Vyhodnocení správnosti algoritmu 1NN po použití každé ze 128 metod proběhlo opět na 10 datových souborech z tabulky 2.1 s výběrem 1 až 20 atributů. Blíže však byly zkoumány pouze výsledky, které v průměru překonaly výsledky metody χ^2, tedy výsledky vedoucí filter metody.

Co se týče prvních osmi metod založených na zjištěných relativních správnostech, výsledky jsou uvedeny v tab. 3.3. Pro stanovení výsledků byly opět vypočteny relativní správnosti, tentokrát vzhledem k χ^2. V tabulce jsou uvedeny průměrné relativní správnosti pro každý datový soubor a také celkový průměr. Hodnoty pro jednotlivé datové soubory nepochází z normální rozložení, proto jsou pro přehlednost uvedeny i mediány. Porovnáním metod mezi sebou se nejeví žádná z nich jako výrazně lepší.

dataset	m1	m2	m3	m4	m5	m6	m7	m8
1	113,89	112,53	106,53	113,07	115,52	113,79	111,30	113,58
2	99,34	56,48	55,90	56,67	56,25	56,83	57,21	57,00
3	122,64	121,06	121,87	121,46	122,01	119,66	120,21	120,57
4	86,67	104,51	104,22	104,31	105,41	104,21	104,07	104,00
5	87,06	89,46	87,79	84,43	86,02	86,62	86,86	83,18
6	131,50	118,72	118,28	117,92	113,88	113,88	113,88	113,90
7	111,99	112,06	111,92	111,50	112,16	112,21	112,40	112,21
8	240,52	253,97	253,74	253,74	253,54	254,41	254,00	254,14
9	135,36	133,18	131,17	130,30	138,87	137,62	136,35	135,65
10	87,30	98,44	97,67	97,45	102,98	102,84	102,73	102,65
průměr	121,63	120,04	118,91	119,09	120,66	120,21	119,90	119,69
medián	112,94	112,30	109,22	112,29	113,02	113,00	111,85	112,90
> 100	6	7	7	7	8	8	8	8

Tab. 3.3: Výsledky kombinačních metod založených na správnosti

Cílem bylo pokusit se nalézt takovou kombinaci metod, která by předčila jednotlivé filter metody. Jak ukazuje poslední řádek tabulky, některé kombinace dopadly i na osmi datových souborech lépe než χ^2. Je možné testovat nulovou hypotézu, která tvrdí, že vybraná kombinační metoda (např. m8) je stejně kvalitní jako χ^2. Za tohoto předpokladu by lepší výsledek pro daný datový soubor nastal s pravděpodobností p = 0,5 a četnost výskytů lepších výsledků by se řídil binomickým rozdělením. Alternativní hypotézou je, že vybraná metoda je lepší než χ^2. Pravděpobnost, že na 10 datových souborech je metoda lepší maximálně v 7 případech je $P(X \leq 7) = 0,945$. Pokud tedy vybraná metoda byla lepší v 8 případech, na hladině významnosti $\alpha = 0.05$ není zamítnuta nulová hypotéza.

Zbývajících 120 metod vytvořených na základě korelační matice takových výsledků nedosahují. Za zmínku stojí pouze ty, jež začínali podle χ^2 a další výběr pokračoval nejvíce korelovanými metodami. Na jednu stranu se neukázalo, že by bylo výhodnější vybírat nekorelované metody, na druhou stranu výsledek příliš nepřekvapuje, neboť u této skupiny jsou vždy primárně vybrány metody χ^2 a SU, tedy stejné jako u prvních osmi kombinačních metod založených na správnosti. Pro těchto 12 metod jsou v tab. 3.4 uvedeny už jen souhrnné výsledky . Všechny další výsledky jsou dostupné v elektronické příloze.

metoda	celkový průměr	medián	počet lepších výsledků
m117	100,89	100,00	5
m118	100,48	100,27	6
m119	100,96	100,12	6
m120	101,25	100,71	7
m121	100,72	100,00	1
m122	100,76	100,00	3
m123	100,91	100,00	2
m124	101,14	100,13	6
m125	100,89	100,00	5
m126	100,62	100,38	7
m127	101,05	100,27	7
m128	101,42	100,64	7

Tab. 3.4: Výsledky pro χ^2 v kombinaci s nejvíce korelovanými metodami

4 ZHODNOCENÍ METOD

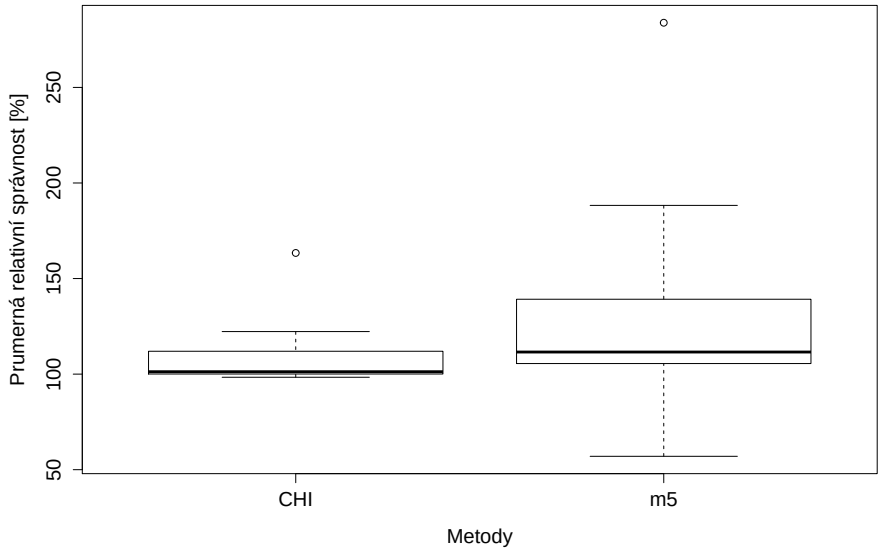
Tato kapitola se soustředí na shrnutí výsledků s možnými doporučeními. I přesto, že metody byly ohodnoceny na základě algoritmu 1NN a stejné výsledky nejsou zaručeny při použití jiných algoritmů, dává algoritmus 1NN ze své podstaty vcelku vhodný způsob ohodnocení vybraných příznaků.

V první řadě je možné na základě provedených výpočtů posoudit jednotlivé filter metody samostatně. Podle výsledků v kapitole 2.2 se obecně ukazuje jako nejlepší filter metoda χ^2. Z tohoto důvodu je možné tuto metodu doporučit jako první volbu pro selekci příznaků. Není však vyloučeno, že na určitém datovém souboru bude dávat lepší výběr příznaků jiná metoda.

Proti samotné filter metodě může být však zajímavější použít kombinaci více metod. Jako nejvhodnější se jeví spojení χ^2 s SU. Tato kombinace je podpořena výsledky experimentů, a to u obou typů kombinací. Konkrétně je vhodné použít metodu označenou v tab. 3.3 jako m5. U výsledků sice nebyla zjištěna statistická významnost, avšak v případě některých datových souborů může být kombinace přínosnější než samotná metoda χ^2. Výběr atributů podle m5 je shrnut jako alg. 5. Na obr. 4.1 je srovnání χ^2 a kombinace m5 podle relativní správnosti vůči IG.

1 t := Požadovaný celkový počet atributů;
2 NewAttributes := nejlepších t/3 atributů podle metody SU;
3 NewAttributes := nejlepších 2t/3 atributů podle metody CHI, které ještě nejsou v NewAttributes;
4 return NewAttributes;

Algoritmus 5: Postup kombinující metody SU a χ^2



Obr. 4.1: Srovnání χ^2 a m5 na datech z tabulky 2.1

5 ZÁVĚR

V práci je srovnáno 5 metod používaných na selekci příznaků, je stanovena jejich vzájemná podobnost pomocí korelační matice a nakonec je navržena skupina nových selekčních postupů kombinujících více původních metod dohromady. Zlepšení dosažené pomocí nových postupů je těsně pod hranicí statistické významnosti.

Porovnání metod je provedeno na základě literární rešerše, dále je experimentálně ověřeno pomocí algoritmu 1NN. Vlastní experimenty byly též využity na stanovení korelace mezi jednotlivými selekčními metodami. Korelace vyjadřují míru podobnosti, s jakou metody postupně vybírají atributy.

V dalším kroku byl na základě vykonaných experimentů proveden návrh a realizace nových postupů, které kombinují existující metody. Tyto nově navržené postupy byly experimentálně vyhodnoceny a srovnány s výsledky pro samotné metody. Nejlépe dopadly kombinace, které používaly primárně takové filter metody, pomocí nichž dosáhl algoritmus 1NN nejvyšších hodnot správnosti. Konkrétně se jedná o χ^2 a Symmetrical Uncertainty. Vhodný způsob zkombinování této dvojice metod je uveden v poslední části práce.

LITERATURA

[1] ALPERT, J. – HAJAJ, N. We knew the web was big... In: *The Official Google Blog* [online]. 2008 [cit. 2012-12-27]. Dostupné z: http://googleblog.blogspot.cz/2008/07/we-knew-web-was-big.html

[2] BELLMAN, R. *Adaptive control processes: a guided tour*. Princeton: Princeton Univ. Press, 1961. ISBN 978-069-1079-011.

[3] BERKA, P. *Dobývání znalostí z databází*. Vyd. 1. Praha: Academia, 2003, 366 s. ISBN 80-200-1062-9.

[4] BUDÍKOVÁ, M. – LERCH, T. – MIKOLÁŠ, Š. *Základní statistické metody*. 1. vyd. Brno: Masarykova univerzita v Brně, 2005. ISBN 978-802-1038-868.

[5] CUNNINGHAM, P. *Dimension Reduction*. Technická zpráva UCD-CSI. University College Dublin, 2007. Dostupné z: http://www.csi.ucd.ie/files/UCD-CSI-2007-7.pdf

[6] FORMAN, G. An extensive empirical study of feature selection metrics for text classification. *Journal of Machine Learning Research*. 2003, s. 1289–1305.

[7] GUYON, I. – ELISSEEFF, A. An introduction to variable and feature selection. *The Journal of Machine Learning Research*. 2003, s. 1157–1182.

[8] JIANG, B.-N. et all. A Hybrid Feature Selection Algorithm: Combination of Symmetrical Uncertainty and Genetic Algorithms. *Lecture Notes in Operations Research 9: The Second International Symposium on Optimization and Systems Biology*. 2008, s. 152–157.

[9] KIRA, K. – RENDELL, L. A. The feature selection problem: traditional methods and a new algorithm. *Proceedings of the tenth national conference on Artificial intelligence*. San Jose, California. AAAI Press, 1992, s. 129–134. ISBN 0-262-51063-4.

[10] JOHN, G. H. – KOHAVI, R. – PFLEGER, K. Irrelevant Features and the Subset Selection Problem. *MACHINE LEARNING: PROCEEDINGS OF THE ELEVENTH INTERNATIONAL*. 1994, s. 121–129.

[11] JOHN, G. H. – KOHAVI, R. Wrappers for Feature Subset Selection. *ARTIFICIAL INTELLIGENCE*. 1997, roč. 97, č. 1, s. 273–324.

[12] KONONENKO, I. Estimating Attributes: Analysis and Extensions of RELIEF. *Proceedings of the European conference on machine learning on Machine Learning*. Springer Verlag, 1994, s. 171–182. ISBN 3-540-57868-4.

[13] LIU, H. – SETIONO, R. Chi2: Feature Selection and Discretization of Numeric Attributes. *Proceedings of the Seventh International Conference on Tools with Artificial Intelligence*. Washington, DC, USA. IEEE Computer Society,1995, s. 388–391.

[14] MIERSWA, I. – WURST, M. – KLINKENBERG, R. – SCHOLZ, M. – EULER, T. YALE: Rapid Prototyping for Complex Data Mining Tasks. *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*. 2006, s. 935–940.

[15] MITCHELL, T. *Machine learning*. Boston: McGraw-Hill, 1997, xvii, 414 s. ISBN 00-704-2807-7.

[16] NOVAKOVIĆ, J. – ŠTRBAC, P. – BULATOVIĆ, D. Toward optimal feature selection using ranking methods and classification algorithms. *Yugoslav Journal of Operations Research*. 2011, roč. 21, č. 1, s. 119–135.

[17] R Core Team (2013). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL http://www.R-project.org/.

[18] SOMOL, P. – BAESENS, B. – PUDIL, P. – VANTHIENEN, J. Filter- versus wrapper-based feature selection for credit scoring. *International Journal of Intelligent Systems*. 2005, roč. 20, č. 10, s. 985-999. ISSN 0884-8173.

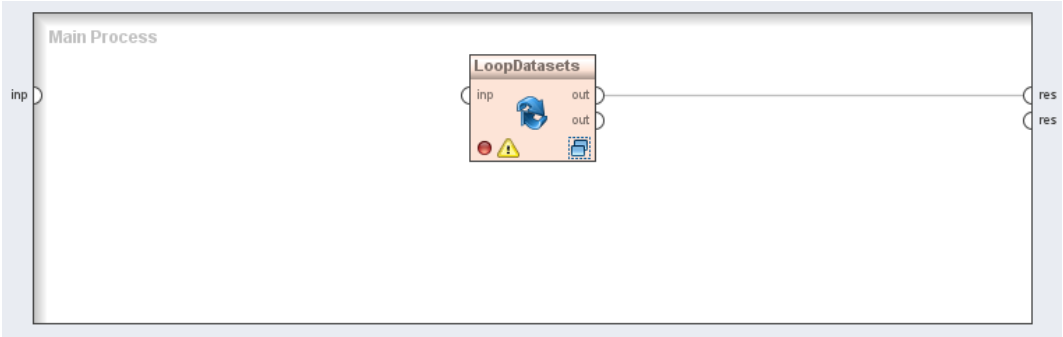
[19] QUINLAN, J. R. Induction of Decision Trees. *Machine Learning.* 1986, s. 81–106.

[20] ZHU, Z. – ONG, YS. – DASH, M. Wrapper-Filter Feature Selection Algorithm Using a Memetic Framework. *Transactions on Systems, Man, and Cybernetics*. 2007, roč. 37, č. 1, s. 70–76.

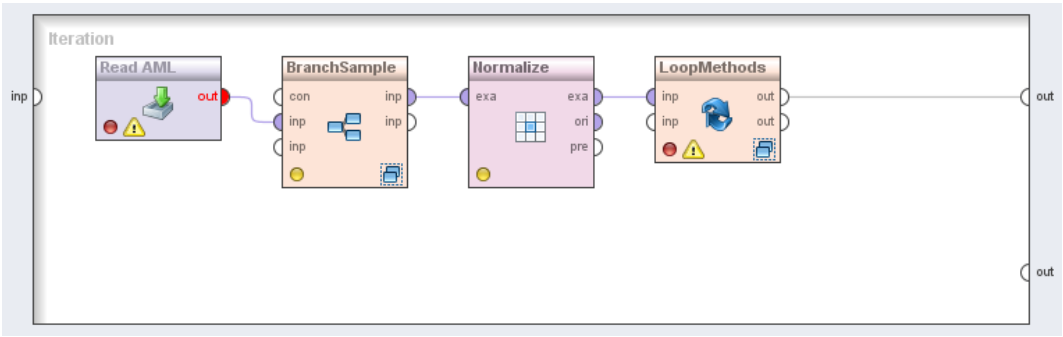
A PŘÍKLAD PROCESU V RAPIDMINER

V této části je pro ilustraci uvedeno schéma jednoho procesu z RapidMiner. Konkrétně se jedná o proces, ve kterém jsou vyhodnoceny jednotlivé filter metody. V textu byl tento proces uveden jako vývojový diagram na obr. 2.1.

Na nejvyšší úrovni probíhá cyklus, který iteruje přes všech 10 datových souborů. Tento cyklus je uveden na obr. A.1. Tělo cyklu je možné nalézt na obr. A.2.

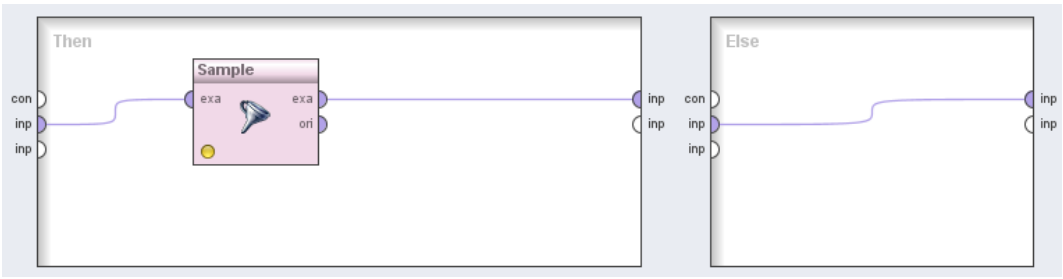


Obr. A.1: Nejvyšší úroveň procesu



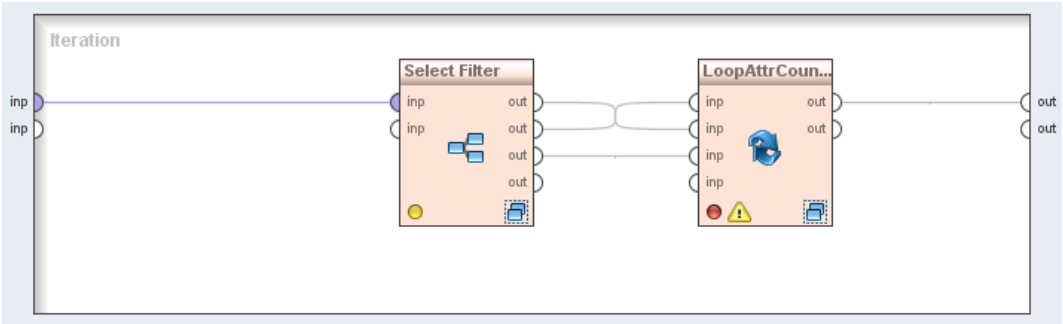
Obr. A.2: Tělo cyklu LoopDatasets

Cyklus nejprve provede načtení vybraného datového souboru pomocí operátoru Read AML. Následuje větvící operátor, jehož tělo je uvedeno na obr. A.3.



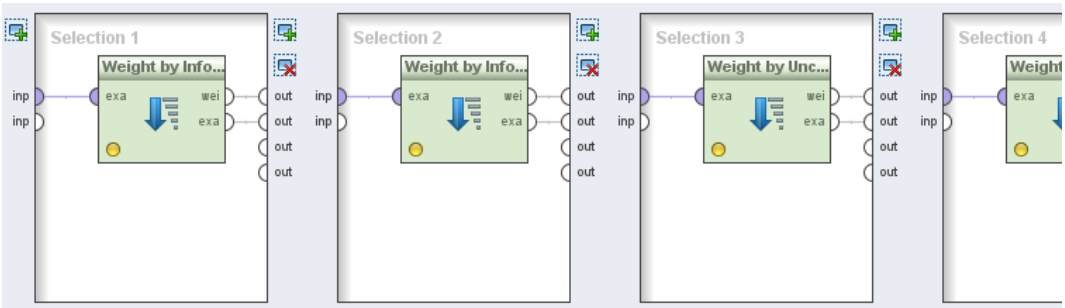
Obr. A.3: Podproces BranchSample

Operátor větvení vybírá určitý podproces k provedení na základě vytvořené podmínky. V tomto případě se testuje číslo datového souboru a u devátého souboru je proveden výběr vzorku dat. Dalším krokem je normalizace dat, po které se iteruje přes jednotlivé filter metody. Tento cyklus je označen jako LoopMethods a jeho strukturu ukazuje obr. A.4.

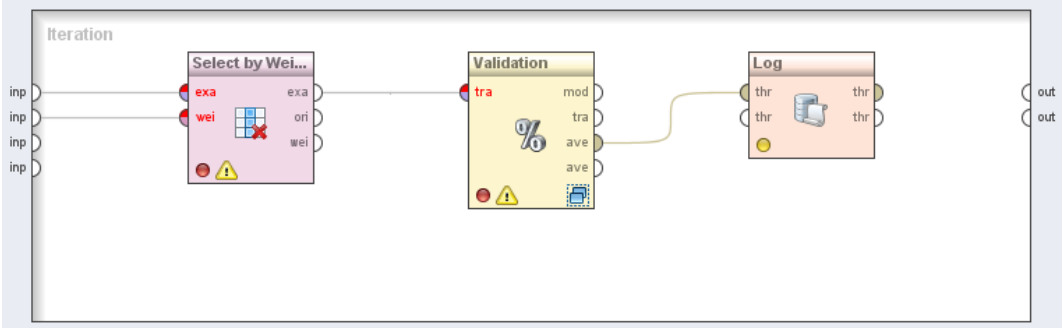


Obr. A.4: Tělo cyklu LoopMethods

Podle aktuální iterace LoopMethods se vybere určitá filter metoda pomocí operátoru větvení. Část větví je zachycena na obr. A.5. V každé větvi jsou data načtena určitou filter metodou a na výstup jsou poslány atributy a přiřazenými váhami.

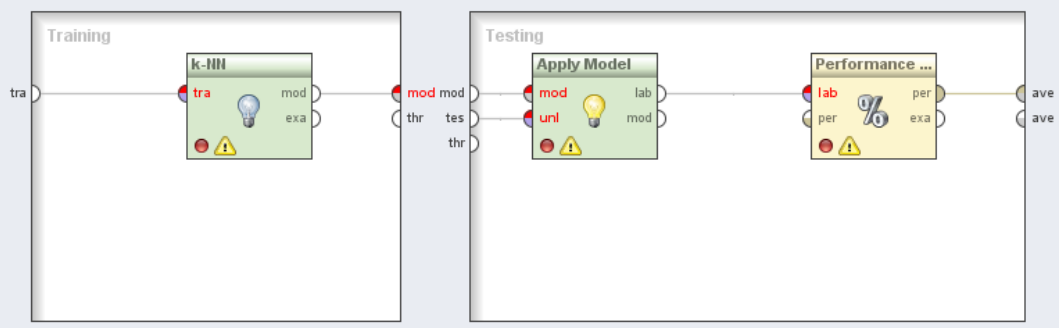


Obr. A.5: Podproces Select Filter



Obr. A.6: Tělo cyklu LoopAttrCount

V LoopMethods je vnořen další cyklus, který iteruje přes počty vybíraných atributů, tedy od 1 do 20. Jak lze vidět na obr. A.6, v tomto cyklu se nejdříve vybere určitý počet nejlepších atributů podle přiřazených vah. Konkrétně se vezme počet daný aktuální iterací cyklu LoopAttrCount. Následuje validace pomocí algoritmu 1NN a výpis výsledků do souboru pomocí operátoru Log. Proces validace zachycuje obr. A.7. Při validaci se střídají dvě fáze, první je trénovací a druhá testovací. V trénovací fázi je uveden algoritmus, který má být použit pro učení; zde je to algoritmus k-NN s parametrem k = 1. Vytvořený model je poté aplikován v testovací fázi a pomocí operátoru Performance je získáno vyhodnocení modelu.



Obr. A.7: Podproces Validation