Институт по математика и информатика
Българска академия на науките
София, България
2011

Извличане на клас-асоциативни правила чрез използване на многомерни номерирани информационни пространства

Автореферат на дисертация, представена за присъждане на образователна и научна степен „доктор"

Илия Митов

Научни ръководители:

проф. д-р Куун Ванхооф
Хаселтски университет, Белгия

доц. д-р Красимир Марков
Институт по математика и информатика,
Българска академия на науките

universiteit hasselt

Institute of Mathematics and Informatics

Институт по математика и информатика
Българска академия на науките
София, България
2011

Извличане на клас-асоциативни правила чрез използване на многомерни номерирани информационни пространства

Автореферат на дисертация, представена за присъждане
на образователна и научна степен „доктор”
по научна специалност 01.01.12 „Информатика”
направление 4.6 „Информатика и компютърни науки”

Илия Митов

Научни ръководители:

проф. д-р Куун Ванхооф
Хаселтски университет, Белгия
доц. д-р Красимир Марков
Институт по математика и информатика,
Българска академия на науките

Авторефератът представя следния дисертационен труд:

Iliya Mitov

Class Association Rule Mining Using Multi-Dimensional Numbered Information Spaces

A thesis submitted for the degree "Doctor of Science" in:

Hasselt University, Faculty of Science

Jury members:

Prof. Dr. Koen VANHOOF (Promoter),
University of Hasselt, Belgium

Assoc. Prof. Dr. Krassimir MARKOV (Copromoter),
Institute of Mathematics and Informatics, Bulgaria

Prof. D.Sci Stefan DODUNEKOV,
Institute of Mathematics and Informatics, Bulgaria

Prof. Dr. Bart GOETHALS,
University of Antwerp, Belgium

Assoc. Prof. Dr. Tom BELLEMANS,
University of Hasselt, Belgium

Assoc. Prof. Dr. Toon CALDERS,
Eindhoven University of Technology, The Netherlands

Assoc. Prof. Dr. Benoit DEPAIRE,
University of Hasselt, Belgium

The thesis defence took place on November 15th, 2011 at 13:00
in Auditorium H4/Building D; Hasselt University; Campus Diepenbeek;
Belgium

Резюме

Асоциативните класификатори имат специално място в семейството на класификационните алгоритми. Те предлагат редица предимства спрямо другите класификационни алгоритми като: ефективност на обучението; независимост от обучаващата извадка; лесно боравене с високо-размерни бази; бързо разпознаване; висока точност; разбираем за хората класификационен модел.

Процесът на създаване на класификационни модели силно зависи от използването на подходящи методи за достъп. В тази работа сме се фокусирали върху използването на една организация на паметта, наречена „Многомерни номерирани информационни пространства", позволяваща работа с контекстно-независими многомерни структури от данни. Целта е да използваме тези структури и операции над тях при реализирането на един асоциативен класификатор като доказателство за жизнеността на идеята за използване на този тип структури от данни като удобно средство за извличане на знания.

Бе разработен един нов асоциативен класификатор PGN, който поставя под въпрос общия подход на даване на приоритет на подкрепата (support) спрямо доверието (confidence), използван широко в досегашните алгоритми. PGN се фокусира първо върху доверието на асоциативните правила, а едва в по-късен етап – на подкрепата на правилата. Едно от основните предимства на PGN е, че е без параметри. При него асоциативните правила се създават от най-дългите към по-късите, докато е възможно създаването на нови правила в класа, като резултат на пресичането на предишно създадените (в частност инстанциите) правила. Във фазата на орязване всички противоречия и по-общи правила между класовете се изчистват, след което в рамките на всеки клас се премахват всички по-конкретни правила, за които има по-общи правила, които ги обхващат.

Допълнително бе разработен метод за ефективно изграждане и съхраняване на множеството от правила в многослойна структура MPGN, използвайки възможностите на многомерните номерирани информационни пространства.

Софтуерът на предложените алгоритми и структури са реализирани в рамките на програмната средата за извличане на знания PaGaNe [Mitov et al, 2009a].

Проведените експерименти доказаха жизнеността на предложените подходи, показвайки добро представяне на PGN и MPGN в сравнение с други класификатори от групите, базирани на правила, особено в случаите на множества от данни с много класове и с неравномерно разпределение.

Благодарности

Изпълнението на тази работа беше поддържано от Хаселтския университет чрез проект R-1876 "Intelligent systems' memory structuring using multidimensional numbered information spaces", както и от Националния фонд научни изследвания по проект ДО 02-308 "Спецификации и стандарти за автоматизирано генериране на метаданни от електронни документи".

Искам да изкажа своята благодарност към Хаселтския университет в Белгия и към Института по математика и информатика – БАН за предоставянето на благоприятни условия за осъществяването на тази дейност.

Специално искам да благодаря на моите научни ръководители проф. Куун Ванхооф и доц. Красимир Марков, а също и на колегите доц. Беноа Депеер от Хаселтския университет, проф. Левон Асланян и доц. Хасмик Сахакян от Института по информатика и автоматизация в Армения, проф. Виктор Гладун и доц. Виталий Величко от Института по кибернетика в Украйна, които ми оказаха голяма подкрепа по време на провеждането на изследванията ми и работата ми по тази разработка.

Също така съм безкрайно задължен за безусловната подкрепа, голямото търпение и постоянното насърчаване, което получавах от съпругата ми – Валерия, през времето на реализиране на тази разработка.

1 Въведение

1.1 Клас-асоциативни правила

В последно време количеството на натрупаната информация в аналогов, а вече и в цифров вид непрекъснато нараства. Поради бързото развитие на всички сфери на човешката дейност в съвременното общество производствените, икономическите и социалните процеси стават все по-комплексни. Повечето организации, които използват информационно-технологични ресурси, събират и съхраняват големи обеми от данни. Днешното предизвикателство е свързано не толкова с начините на събиране и съхраняване на необходимите данни, колкото с възможността да се извлекат смислени изводи от този огромен обем от информация. Решението на този проблем е в развитието на технологиите на извличане на данни и в частност, в използването на асоциативните правила.

Основната цел на извличането на асоциативни правила е да се открият закономерности в постъпващите данни. Проблемната област се заражда в задачите за анализ на потребителската кошница, където се търси наличието на интересни закономерности в големи масиви от данни [Agrawal et al, 1993]. По-късно се вижда, че може да бъде направена връзка между извлечените асоциативни правила и класовете, към които те принадлежат и това да се използва като елемент в машинното обучение [Bayardo, 1998]. Оттогава са предложени много асоциативни класификатори, които основно се различават по стратегиите за избор на класификационните правила и в евристичните методи, използвани за орязване на правилата.

Асоциативните класификатори предлагат нова алтернатива в схемите за класификация чрез производство на правила, основаващи се на съюза на двойки атрибут-стойности, които се срещат често в масивите от данни. Често срещаните патерни в тези данни характеризират интересни взаимоотношения между атрибутивните условия и етикетите на класовете, които позволяват използването им за по-ефективна класификация.

Асоциативните правила се извличат в двуетапен процес, състоящ се от търсене на често срещаните комбинации, последвано от формирането на правило, при което се ползват различни критерии за оценка на значимостта на потенциално формиращото се правило, с цел неговото включване в последващите стъпки на алгоритъма или отхвърлянето му.

1.2 Многомерни номерирани информационни пространства

Един най-общ преглед на използваните информационни структури в алгоритмите за извличане на асоциативни правила показва голямо разнообразие от различни решения, използвани за съхраняване и извличане на информация: графи, хеш-таблици, различни видове дървета, двоични матрици, масиви и пр. Всеки вид структура от данни носи както предимства, така и недостатъци. В [Liu et al, 2003] се обръща специално внимание на тези въпроси като се представя сравнение между дървета и масиви. Изследването показва, че дървовидните структури са в състояние да намалят броя обхождания чрез обединение на различните транзакции по техните префикси, но пък се заплаща по-скъпа цена при построяването на дървото, особено когато множествата от данни са разпръснати и големи. Масивите изискват по-малко разходи по време на изграждането, но пилеят повече време при обхожданията поради невъзможността за обединение на различните транзакции.

Като основен подход при организацията на паметта ние решихме да използваме номерирането, което позволява смяна на имената с номера и използването на математически функции и адресни вектори за достъп до информацията. Допълнително предимство е, че номерирането позволява използването на един и същ механизъм за адресиране както за основната, така и за външната памет. Използваният подход борави както с голям брой размери, така и с голям брой елементи от дадено измерение. Този тип организация на паметта се нарича "Многомерни номерирани информационни пространства". Предимствата й са доказани в множество реални изпълнения [Markov, 1984, 2004, 2005]. Досега този вид организация на паметта не е била използвана в областта на изкуствения интелект. Прилагането й в областта на системи за управление на бази от данни е показала редица удобства. Най-общо предимствата на номерираните информационни пространства са:

- възможност за изграждане на растящи пространствени йерархии от информационни елементи;
- мощност при изграждането на връзки между информационните елементи чрез използването на адресни индекси.

Основната идея е да се заменят стойностите на атрибутите на обектите с цели числа, които са поредните номера на стойностите на елемента в съответните подредени множества от допустими стойности. По този начин всеки обект се описва чрез вектор с целочислени стойности, който може да бъде използван като координатен адрес в многомерно информационно пространство.

1.3 Цели на дисертацията

Целите на дисертационната работа са ориентирани в две направления:

- да се представи един асоциативен класификатор без параметри, който се фокусира основно върху доверието при изграждането на асоциативните правила. Ние очакваме, че този подход ще осъществи висококачествено разпознаване, особено в рамките на небалансирани и с много класове множества от данни;
- да покаже предимствата от използването на многомерните номерирани информационни пространства като структури на паметта в процесите по извличане на знания като за целта се използва създадения асоциативен класификатор.

1.4 Структура на дисертацията

Дисертацията се състои от девет глави и приложение както следва:

1. Въведение.
2. Извличане на данни и откриване на знания.
3. Асоциативни класификатори.
4. Многомерни номерирани информационни пространства.
5. Алгоритмите PGN и MPGN.
6. Програмна реализация.
7. Пример върху множеството от данни "Lenses".
8. Анализ на чувствителността.
9. Заключение и планове за бъдеща работа.

По-долу е направен кратък преглед на съдържанието на главите.

Глава 2 въвежда в процеса на извличане на данни и неговото значение за общия процес на откриване на знания. Дадена е една таксономия на методите за извличане на данни със специален фокус върху методите за класификация. Направен е сбит преглед на основните видове класификатори, последвано от кратко описание на процеса на дискретизация, която е важна част от процеса на подготовка на данни в глобалната рамка на процеса на откриване на знания. В допълнение в тази глава се представят някои известни програмни среди с отворен код за откриване на знания.

Глава 3 представя преглед на областта на асоциативните класификатори. Представени са отделните стъпки в процеса на класификация, типични за тоя тип алгоритми: генериране на правила, орязване и разпознаване. Показани са различните видове техники, които се използват във фазата на генериране на правилата. Орязването, което е важен елемент от процеса на обучение на асоциативните класификатори, се прилага като предварителна стъпка, паралелно с извличането на правилата или след това. Обсъждат се и различни видове мерки за оценка на качеството на дадено правило и схеми на подреждане на правилата. Във фазата на разпознаване се използват различни видове стратегии: избор на едно правило или набор от правила с различни схеми на подредба. Разгледани са и няколко конкретни представителя на асоциативни класификатори, които показват разнообразието на предложените вече техники.

Глава 4 е фокусирана върху различните видове съществуващи методи за организация на данните в областта на извличане на данни и откриването на знания. В тази глава е представен и определен тип организация на паметта, наречена "Многомерни номерирани информационни пространства", както и тяхната програмна реализация ArM 32. Описани са основните структури на метода и операциите с тях.

Глава 5 съдържа описание на предложените нови алгоритми за класификация – PGN и MPGN.

Глава 6 се спира на програмната реализация на предложените алгоритми. Тя предоставя кратко описание на експериментална среда за извличане на данни – PaGaNe, която е резултат от съвместната работа на изследователи от България и Белгия.

Глава 7 описва конкретните стъпки, през които преминават алгоритмите PGN и MPGN на примера на множеството от данни "Lenses".

Глава 8 се фокусира върху представянето на няколко експеримента, извършени с разработените инструменти в PaGaNe. Направено е сравнение между PGN, MPGN и други класификатори, които имат подобно поведение за създаване на класификационен модел, базиран на правила.

Глава 9 съдържа изводи и преглед на насоките за бъдещи изследвания.

В допълнението са включени пълния набор от данни от допълнителни експерименти, включващи PGN, MPGN и широка гама от различни видове класификатори, реализирани в програмната среда WEKA.

Текстът на дисертацията е 200 стр. и съдържа 35 таблици, 46 фигури и 106 препратки към литературни източници.

1.5 Използвани означения

Обикновено в класификационните задачи се използват т.нар. правоъгълни множества от данни – $\mathbf{R} = \{R^i, i \in 1,...,r\}$. Всяка инстанция представя множество от двойки "атрибут-стойност" $R = \{C = c, A_1 = a_1,..., A_n = a_n\}$, където n е броят на атрибутите. Понеже в правоъгълните множества от данни позициите на атрибутите, съответно на класа, са фиксирани, инстанциите се представят като вектори, съдържащи само стойностите на атрибутите. За повишаване на четимостта, стойността на класа, който в настоящата работа е прието да е в първа позиция на вектора, се отделя от останалите атрибути с "|": $R = (c \mid a_1,...,a_n)$. Знакът "-" използваме когато искаме да означим, че стойността на даден атрибут може да приема произволни стойности.

Патерните имат подобна структура и представляват подмножества на инстанциите: $P = (c \mid a_1,...,a_n),\ R = (c \mid b_1,...,b_n),\ a_i = b_i \text{ or } a_i = \text{"–"} \Rightarrow P \subseteq R$.

(c) се нарича "глава", а $(a_1,...,a_n)$ – тяло на патерна. Всеки патерн дефинира правило по следния начин: ако някои атрибути имат определена специфична стойност (непроизволните стойности на атрибутите от тялото), то може да се заключи, че разглежданият обект принадлежи на класа, отбелязан в главата на патерна.

Мощност на патерна (cardinality): $|P| = \text{number of } a_k \neq \text{"–"};\ |P| \leq n$.

Пресичане на два патерна:

$$P^i \cap P^j = (c^l \mid a_1^l,...,a_n^l):\ c^l = \begin{cases} c^i : c^i = c^j \\ \text{"–"} : c^i \neq c^j \end{cases} \text{ and } a_k^l = \begin{cases} a_k^i : a_k^i = a_k^j \\ \text{"–"} : a_k^i \neq a_k^j \end{cases}.$$

Ако $|P^i \cap P^j| > 0$ и $c^i = c^j$, то $P = P^i \cap P^j$ е патерн. P се нарича наследник на P^i и P^j; съотв. P^i и P^j се наричат предходници на P.

За два патерна, принадлежащи на различни класове $P^i = (c^i \mid a_1^i,...,a_n^i)$ и $P^j = (c^j \mid a_1^j,...,a_n^j)$, $c^i \neq c^j$ съществуват следните възможности:

– P^i е изключение на P^j ако $(a_1^i,...,a_n^i) \supset (a_1^j,...,a_n^j)$;

– P^i противоречи на P^j ако $(a_1^i,...,a_n^i) = (a_1^j,...,a_n^j)$.

За един патерн P спрямо множество от данни $\mathbf{R} = \{R^i, i \in 1,...,r\}$:

– подкрепата (support) представлява броят инстанции, на които P е подмножество: $Supp(P,\mathbf{R}) = \text{number of } R^i : P \subseteq R^i,\ R^i \in \mathbf{R}, i \in 1,...,r$.

– доверието (confidence) е равно на съотношението на подкрепата на патерна спрямо подкрепата на тялото на патерна в множеството: $conf(P,\mathbf{R}) = \frac{Supp(P,\mathbf{R})}{Supp((-|a_1,...a_n),\mathbf{R})}$.

Заявката за разпознаване Q прилича на патерна, но стойността на класа е неизвестна: $Q = (?|b_1,...,b_n)$.

Процентът на пресичане между патерна P и заявката Q се дефинира като $IntPerc(P,Q) = 100 * \frac{|P \cap Q|}{|P|}$.

2 Асоциативни класификатори

Обикновено структурата на алгоритмите на асоциативните класификатори се състои от три основни стъпки: (1) Извличане на асоциативни правила, (2) Орязване, и (3) Разпознаване.

Извличането на асоциативните правила е типична задача от областта на извличане на данни (data mining), която работи по неконтролиран начин (unsupervised manner). Основно предимство на асоциативните правила е, че теоретично те могат да разкрият всички интересни връзки в базата от данни. В практическите приложения обаче, броят на извлечените правила обикновено е твърде голям за да бъдат използвани всичките. По тази причина се прилага орязване на част от правилата с цел изграждане на точни и компактни класификатори. Колкото по-малко на брой правила са необходими, за да се получи задоволително разпознаване, толкова по-човешки интерпретируем е резултатът от класификацията.

За пръв асоциативен класификатор се приема CBA, създаден през 1998г [Liu et al, 1998]. През последното десетилетие възникват множество други асоциативни класификатори, като CMAR [Li et al, 2001], ARC-AC и ARC-BC [Zaïane and Antonie, 2002], CPAR [Yin and Han, 2003], CorClass [Zimmermann and De Raedt, 2004], ACRI [Rak et al, 2005], TFPC [Coenen and Leng, 2005], HARMONY [Wang and Karypis, 2005], MCAR [Thabtah et al, 2005], 2SARC1 and 2SARC2 [Antonie et al, 2006], CACA [Tang and Liao, 2007], ARUBAS [Depaire et al, 2008], и др.

2.1 Извличане на асоциативните правила

За извличането на асоциативните правила се прилагат различни техники, основно базирани на:

- Apriori [Agrawal and Srikant, 1994] (CBA, ARC-AC, ARC-BC, ACRI, ARUBAS): алгоритъмът прави множество пасове върху данните с цел извличане на патерните с дължини от 1 до k. Всеки пас се състои от 2 стъпки: (1) създаване на патерна-кандидат с дължина k; (2) запазване само на патерните, чиято подкрепа е над определена зададена граница;
- FP-tree [Han and Pei, 2000] (CMAR): FP-tree представлява разширено дърво на префикси, съхраняващо количествена информация за често срещаните патерни;
- FOIL [Quinlan and Cameron-Jones, 1993] (CPAR): това е един последователно покриващ алгоритъм, който строи правила на базата на логика от първи ред. След създаването на ново правило, всички удовлетворяващи се от него инстанции се премахват, след което започва строенето на следващото правило;
- Morishita & Sese Framework [Morishita and Sese, 2000] (CorClass): алгоритъмът изчислява значимостта на асоциативните правила на базата на общи статистически мерки като хи-квадрат или коефициент на корелация.

Генерирането на асоциативните правила може да бъде правено:

- използвайки всички транзакции (CBA, CMAR, ARC-AC);
- групирайки транзакциите по стойностите на класа им (ARC-BC).

2.2 Орязване

С цел намаляване на броя асоциативни правила се използват различни техники на орязване по време на създаването им (pre-pruning) или след това (post-pruning). Използват се различни евристики, основно базирани на минимална подкрепа, минимално доверие или различни видове изчистване на грешки [Kuncheva, 2004]. Когато орязването се прави като самостоятелна стъпка след изграждането на правилата се използват и критерии като покриваемост на данните от правилото (ACRI) или корелация между тялото и главата на правилото (CMAR).

По време на фазата на орязване, а също и в етапа на класификация, се използват също и различни критерии за подредба на правилата. Най-честите механизми се базират на подкрепата, доверието и мощността на правилата (ARC-AC, ARC-BC), но съществуват

и други техники, прилагащи косинусова мярка или мярка на покриваемостта (ACRI), а също и орязване по подредба (CBA, MCAR).

2.3 Разпознаване

По време на разпознаването също се използват различни подходи [Depaire et al, 2008]: използване на едно правило (CBA); използване на подмножество от правила (CPAR); или използване на всички правила (CMAR). За подреждане на правилата отново се използват различни мерки: избор на всички отговарящи правила, групиране по стойността на класа, подреждането им в класа по определен критерий, изчисляване на комбинирана мярка за първите Z правила, и пр.

3 Метод на достъп MDIM и ArM 32

MDIM (Multi-Dimensional Information Model) е контекстно-свободен метод на достъп, притежаващ следните предимства:

- замяната на имената с номера позволява да се ползват математически функции и адресни вектори за достъп до информацията;
- номерирането има предимството на използване на един и същ адресиращ механизъм за вътрешната памет и за външната памет;
- подходът позволява едновременното използване на различни размерности.

Основните конструкции в MDIM са:

- базови информационни елементи (произволни последователности от машинни кодове);
- номерирани информационни пространства (организация на базовите информационни елементи в структури, подобни на масиви, но с йерархична вътрешна организация). Всеки елемент е достъпен чрез координатен адрес $A=(n,p_1,...,p_n)$, където n е размерността (променлива), а $p_1,...,p_n$ са координати;
- индекси и мета-индекси (последователности от пространствени адреси). Специален вид пространствен индекс е проекцията (аналитично представен индекс). По-специално са дефинирани два вида проекции: йерархична проекция (фиксирани са началните координати) и произволна проекция (фиксирани са координати в произволни позиции).

Върху тези елементи могат да се извършват следните операции:

- с елемент – обновяване (счита се, че виртуално всички елементи съществуват), получаване на стойността, получаване на дължината, позициониране в елемента;
- с две пространства – копиране на първото пространство във второто или преместване на първото пространство във второто с изчистване или оставяне на съществуващата информация във второто пространство;
- с йерархична проекция – обхождане на дефинираната област и извличане на предишен/следващ празен/непразен елемент, получавайки целия индекс или неговата дължина за непразните елементи;
- с произволна проекция – същите операции, но само върху непразните елементи.

Програмната реализация на MDIM се нарича ArM 32.

4 Алгоритъм PGN

В тази глава се представя един асоциативен класификатор, наречен PGN. Една от най-важните му специфики е, че той е без параметри, за разлика от класическите асоциативни класификатори, при които потребителят трябва да зададе ниво на подкрепа и доверие.

Изграждането на асоциативните правила се извършва от най-дългите правила (инстанциите) към по-кратките докато престане да съществува възможност за пресичане между патерните в рамките на класа. Във фазата на орязването всички противоречиви и несъгласувани по-общи правила между класовете се изчистват, след което множеството от патерни се уплътнява чрез изхвърляне на всички по-конкретни правила, за които има по-общи в рамките на класа.

По-долу е дадено подробно описание на алгоритъма. Като пример е използвана една просто множество от данни, съдържащо следните инстанции:

```
R1:  (1| 1, 2, 4, 1)
R2:  (1| 1, 2, 3, 1)
R3:  (1| 3, 1, 3, 2)
R4:  (1| 3, 1, 4, 2)
R5:  (1| 1, 2, 4, 1)  Еднакво с R1
R6:  (1| 3, 1, 4, 2)  Еднакво с R4
R7:  (2| 3, 1, 1, 2)
R8:  (2| 2, 1, 1, 2)
R9:  (2| 3, 1, 2, 2)
```

4.1 Обучение

Процесът на обучение се състои от:

- обобщение – извличане на асоциативните правила;
- изрязване – изчистване на противоречията и изключенията между класовете и олекотяване на множествата от патерни в класовете.

Стъпка 1: Обобщение

Процесът на създаване на множеството от патерни се състои от 2 под-стъпки:

- добавяне на инстанциите към множеството от патерни;
- създаване на възможните пресичания между патерните в рамките на всеки клас поотделно.

Под-стъпка 1.1: Добавяне на инстанциите

Инстанциите от обучаващото множество $LS = \left\{ R^i \right\}$, $i = 1,...,t$ се добавят към множеството от патерни към всеки клас като начални патерни (Фигура 1).

Фигура 1. Добавяне на инстанциите в множеството от патерни

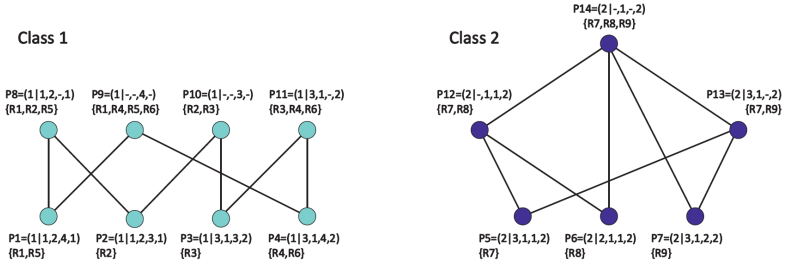
Под-стъпка 1.2: Добавяне на пресичанията

За всеки клас поотделно се правят пресичанията между всеки два негови патерна и ако това пресичане произвежда патерн, той се добавя в множеството от патерни на съответния клас. Ако патернът е вече записан в множеството от патерни само се увеличава множеството от инстанции/патерни, от които е създаден. Процесът е итеративен и продължава докато могат да се създават нови патерни.

В края на тази стъпка множеството от патерни се състои от:

$$PS = \left\{ P^l \right\},\ P^l : \begin{cases} R^l \in LS \\ P^l = P^i \cap P^j;\ \ P^i \in PS, P^j \in PS, c^i = c^j; \left| P^l \right| > 0 \end{cases}.$$

Фигура 2 показва този процес върху примерната база.



Фигура 2. Добавяне на пресичанията в множеството от патерни

Стъпка 2: Изрязване

В тази стъпка се премахват някои от патерните. Процесът се изпълнява на 2 етапа:

– изчистване на противоречията и изключенията между класовете;

– олекотяване на множеството от патерни в рамките на класовете.

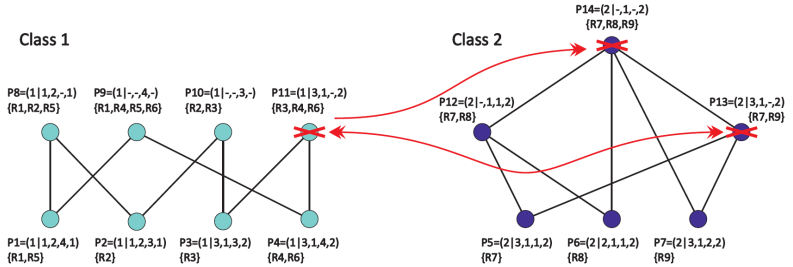
2.1: Изчистване на противоречията и изключенията между класовете

Тук се сравняват патерните, принадлежащи на различни класове. Ако тялото на единия е подмножество на тялото на другия, тогава първият (по-общия) се изтрива. Ако двете тела са еднакви, тогава и двата патерна се унищожават.

$$P^i, P^j \in PS,\ c^i \neq c^j :\ \begin{cases} \left|P^i \cap P^j\right| = \left|P^i\right| < \left|P^j\right| : \text{remove } P^i \\ \left|P^i \cap P^j\right| = \left|P^j\right| < \left|P^i\right| : \text{remove } P^j \\ \left|P^i \cap P^j\right| = \left|P^i\right| = \left|P^j\right| : \text{remove } P^i, P^j \end{cases}$$

Тази етап се опитва да осигури максимално доверие на оставащите правила. Операцията изтрива патерните, които не формулират комбинация, представителна за дадения клас защото в другия клас има патерн със същата или по-подробна комбинация от атрибути, който би претендирал да разпознае заявка с такива атрибути. Освен това, чрез изтриването на некоректните патерни (записи с еднакви тела от различни класове) операцията игнорира възможните несъответствия в обучаващото множество (инстанциите са част от патерните). Следва да бъде отбелязано, че идеята за осигуряване на максимално доверие от 100% може да доведе до несъздаването на множество от патерни в зашумени множества от данни.

Фигура 3 илюстрира етап на изрязването върху разглеждания пример. Процесът протича на два паса: първо маркиране, следвано от действително изтриване.



Фигура 3. Осигуряване на максимално доверие на правилата

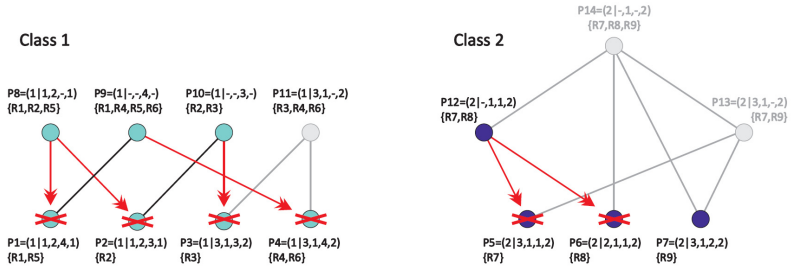
2.2: Оставяне на най-общите правила в класовете

Този етап се изпълнява в рамките на всеки клас. Патерните от един и същ клас се сравняват и, обратно на предишната операция, се заличават по-конкретните патерни.

$$P^i, P^j \in PS,\ c^i = c^j : \begin{cases} \left|P^i \cap P^j\right| = \left|P^i\right| < \left|P^j\right| : \text{remove } P^j \\ \left|P^i \cap P^j\right| = \left|P^j\right| < \left|P^i\right| : \text{remove } P^i \end{cases}$$

Логиката е, че след етапа на изрязване в множеството от патерни на класа са останали само такива, които нямат изключения в друг клас. Поради това можем да олекотим множеството от патерни чрез премахване на тези, за които съществуват патерни, които са техни подмножества.

Фигура 4 илюстрира действието на етапа върху примерното множество.



Фигура 4. Оставяне на по-общите правила

Като краен резултат в множеството от патерни остават само такива, които от една страна са общи за класа, а от друга страна техните тела не са подмножества на патерни от други класове.

За примерното множество от данни финалното множество от патерни има следния вид:

	Множество от патерни	Подкрепа	Подкрепящо множество
	Клас 1		
P8	(1| 1, 2, -, 1)	3	{R1,R2,R5}
P9	(1| -, -, 4, -)	4	{R1,R4,R5,R6}
P10	(1| -, -, 3, -)	2	{R2,R3}
	Клас 2		
P7	(2| 3, 1, 2, 2)	1	{R9}
P12	(2| -, 1, 1, 2)	2	{R7,R8}

4.2 Разпознаване

Записът за разпознаване се дава чрез стойностите на атрибутите си $Q = (? \mid a_1, a_2, ..., a_n)$ като някои от тях може да липсват.

Опитваме се да намерим най-доброто напасване между заявката и патерните от множеството от патерни на съответните класове.

Идеята е, че в множеството от патерни едновременно има патерни, които са много общи (само няколко конкретни атрибути) и други, които са много по-конкретни (с повече / много конкретни атрибути). Общите патерни са малки по размер, но много мощни за класа. Те съдържат само няколко атрибута, но с високо доверие за разпознаване на този клас. От друга страна, конкретните патерни са останали във финалния вариант, защото по-общи комбинации са били унищожени от други класове, което означава, че няма такива малко на брой специфични атрибути и само сложна комбинация от тях характеризира класа.

Нашето предложение да използваме процентът на пресичане се основава на идеята, че той позволява определено сближаване между късите и дългите патерни. Разбира се, процентът на пресичане страда от определен вид неравенство когато не е 100%; тогава късите патерни получават по-ниска стойност, отколкото по-дългите. В този случай, обаче, предпочитането на по-конкретните правила, когато няма пълно съвпадение със заявката, е по-добро, защото осигурява съвпадение на повече атрибути.

Тези предположения реализирахме в алгоритъма на разпознаване както следва: по време на разпознаването всички патерни, които имат максимален процент на пресичане със заявката изграждат списък на патерните, съдържащи потенциалните отговори. Списъкът се състои от триплети, съдържащи номера на класа, мощността и позицията на

патерна. Най-високият процент на пресичане се определя динамично по време на преглеждането на патерните. Като потенциални отговори се задържат само тези патерни, които имат такъв (максимален) процент на пресичане. Накрая, класът, който има максимална сума на подкрепа на патерните от този клас, принадлежащи към списъка на потенциалните отговори, се дава като отговор.

Да разгледаме един пример: имаме заявката за разпознаване $Q = (?|1,2,1,2)$:

	Множество от патерни	$P \cap Q$	$IntPerc(P,Q)$	**Подкрепа**	**Подкрепящо множество**
	Клас 1				
P8	(1| 1, 2, -, 1)	(?| 1, 2, -, -)	**66.7%**	3	{R1,R2,R5}
P9	(1| -, -, 4, -)	(?| -, -, -, -)	0	4	{R1,R4,R5,R6}
P10	(1| -, -, 3, -)	(?| -, -, -, -)	0	2	{R2,R3}
	Клас 2				
P7	(2| 3, 1, 2, 2)	(?| -, -, -, 2)	25.0%	1	{R9}
P12	(2| -, 1, 1, 2)	(?| -, -, 1, 2)	**66.7%**	2	{R7,R8}

Максималният процент на пресичане между патерните и заявката е 66.7%. Има два патерна от двата класа с такъв процент, т.е. има две множества от потенциални отговори: $class\ 1 : \{P8\}$ и $class\ 2 : \{P12\}$. Клас 1 има по-висока подкрепа 3, поради което клас 1 се дава като отговор.

5 Алгоритъм MPGN

Класификаторът PGN има редица предимства и експериментите с него показват много добри резултати [Mitov, 2011]. Едновременно с това той страда от един недостатък, свързан с експоненциалния растеж на операциите по време на процеса на създаване на множеството от патерни. За преодоляване на тази пречка се създаде алгоритъмът MPGN, чиято основна цел е използването на специален вид многослойни структури, наречени "пирамиди".

Процесът на обобщение представлява верига от създаване на патерни в горния слой, като резултат от пресичанията между патерните от по-нисък слой, дотогава докато се генерират нови патерни. За всеки клас процесът започва от слой 1, който съдържа инстанциите от обучаващото множество. По време на обобщението, за всеки клас се изгражда отделна пирамидална структура. Процесът на обобщаване създава "вертикални" взаимовръзки между патерните от съседните слоеве. Тези връзки за всеки патерн са представени от множество от предшественици и множество от наследници. Множеството от предшественици на конкретен патерн съдържа всички патерни от по-долния слой, използвани в процеса на неговото генериране.

Множеството от наследниците на даден патерн съдържа патерните от горния слой, които са създадени от него. Множествата от наследниците на върховете на пирамиди са празни.

С цел анализиране на различни типове поведение при асоциативните класификатори реализираме други алгоритми в следващите фази – изрязването и разпознаването.

Фазата на изрязване при MPGN се различава от тази при PGN доколкото тук се изчистват само върховете на пирамидите от различни класове, чиито тела напълно съвпадат (т.е. само противоречивите). Също така тук не се прилага олекотяване на множеството чрез последващо изчистване на по-подробните в рамките на класовете.

Фазата на разпознаване също протича в леко променен вариант – заявката за разпознаване се сравнява с патерните на съответните класове. Сравняването започва от върховете на класа и продължава надолу по предходниците им докато има патерни, които пълно разпознават заявката. Накрая се избира класът, в който са достигнати патерни с най-голяма мощност. При наличие на два или повече такива класове се прилагат различни стратегии за продължаване на процеса на разпознаване. В дисертационната работа са описани две от тях: S1 – избор от всеки клас на едно правило с максимално доверие в рамките на класа; и S2 – намиране на "доверието на множеството". Ако доверието е също равно се взима този от конкуриращите се класове, който е с най-голяма подкрепа. В случай, че процесът изобщо не стартира (нито един връх не разпознава напълно заявката) се прилагат други алгоритми за разпознаване. В работата е описан един възможен такъв алгоритъм.

6 Програмна реализация

PGN и MPGN са реализирани в рамките на средата за извличане на данни PaGaNe. Тя използва предимствата на многомерните номерирани информационни пространства, реализирани в метода за достъп ArM 32.

Важна характеристика на подходите, използвани в PaGaNe, е замяната на символните стойности на характеристиките на обектите с поредните номера на елементите от съответните подредени множества от допустими стойности. Всички инстанции или патерни могат да се представят от вектор с целочислени стойности, който може да бъде използван като координатен адрес за работа в съответното многомерно информационно пространство.

6.1 Предварителна обработка

Предварителната обработка има за цел да преобразува обучаващото множество в стандартна форма за последващите стъпки. Тя се състои от: (1) дискретизация на числени атрибути; (2) номериране на стойностите на атрибутите, и (3) избор на подмножество от атрибути.

6.1.1 Дискретизация

PGN, MPGN, както и останалите асоциативни класификатори обработват нечислови атрибути. По тази причина като предварителна обработка в PaGaNe са реализирани различни алгоритми за дискретизация. Подробно описание на реализираните алгоритми е направено в [Mitov et al, 2009b].

6.1.2 Преобразуване на първичните инстанции в числови вектори

В тази стъпка се генерират съответствия между първичните стойности на атрибутите и естествените числа (като поредни номера в подредените множества от допустими стойности за съответния атрибут). След това инстанциите се кодират в числени вектори чрез подмяна на стойностите на атрибутите със съответни им номера. Идеята е данните да се преобразуват като преки адреси в многомерните номерирани информационни пространства.

6.1.3 Избор на подмножеството от атрибути

Статистическите наблюдения върху действието на алгоритмите от PaGaNe върху различните множества от данни показват, че някои атрибути не дават важна информация. В PaGaNe такива атрибути могат да се маркират и да не участват в по-нататъшната обработка. Автоматичният избор на такова подмножество е неразработена част от реализацията на PaGaNe и това е една от задачите за решаване в близко бъдеще [Aslanyan and Sahakyan, 2010].

6.2 Процесът на обучение

Много важен аспект е, че за всеки клас съществуват отделни пространства, които са с многослойна структура, наречени "пирамиди". Всички слоеве имат еднаква структура и се състоят от "множество от патерни" и "пространство на връзките". За всеки клас се пази също "множество от върховете", които се използват при разпознаването.

6.3 Конструктивни елементи

Тук ще дефинираме структурата на "множество от патерни", "пространство на връзките", и "множество от върхове".

Множество от патерни

Всеки патерн принадлежи към определен клас c и слой l. Пълното означение на патерна трябва да бъде $P(c,l)$. В случай, че от контекста е ясно за кой клас става дума, патернът може да бъде означаван и просто с $P(l)$. Когато от контекста е ясен и слоят, тогава се обозначава просто с P.

Всички патерни от класа c, които принадлежат към слоя l, формират множество от патерни $PS(c,l) = \{P^i(c,l) \mid i = 1,...,n_{c,l}\}$. Всеки патерн $P(c,l)$ от $PS(c,l)$ имат идентификатори $pid(P,c,l)$, които са естествени числа. Идентификаторите се създават във възходящ ред на входящите патерни в множеството от патерни.

Процесът на обобщаване създава вертикални взаимовръзки между патерните от съседните слоеве. Тези връзки за всеки патерн са представени от две множества – на предшествениците и на наследниците.

Множеството от предшественици $PredS(P^i)$ съдържа идентификаторите на патерните от по-долния слой, които са били използвани в процеса на създаване на този патерн. Множествата от предшественици на инстанциите (слой 1) са празни.

Множеството от наследници $SuccS(P^i)$ съдържа идентификаторите на патерните от горния слой, които са създадени от този патерн. Множествата от наследниците на патерните, които са върхове на пирамидата, са празни.

Един патерн може да участва във формирането на повече от една пирамида, но върховете принадлежат само на една пирамида.

Пространство на връзките

Целта на пространството на връзките е да описва всички закономерности между атрибутите, които се съдържат в класовете. За стойностите на всеки атрибут се създават връзки към съдържащите го патерни, което позволява да се създаде йерархия от множества. Структурата на тази йерархия е както следва:

- множество на стойностите на атрибутите: множество от патерни за дадена стойност на даден атрибут;

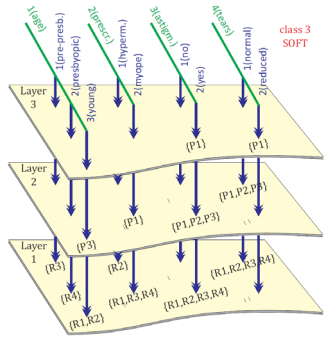
– множество на атрибутите: множество от множествата на стойностите на атрибутите за даден атрибут;
– пространство на връзките (едно): множество от всички възможни множества на атрибутите.

Създаването на пространството на връзките използва предимствата на многомерните номерирани информационни пространства чрез възможността за замяна на търсенето с директен достъп по координатни адреси.

По-долу ще съсредоточим вниманието си върху пространството на връзките, което се превръща в ключов елемент при генерирането на нови патерни, както и при търсенето на патерни, отговарящи на заявки.

Нека c е номерът на разглеждания клас и l е номерът на даден слой:

– множество на стойностите на атрибутите $VS(c,l,t,v)$, $v=1,...,n_t$ е множеството от всички идентификатори от инстанции/патерни за класа c, слой l, които имат стойност v за атрибут t: $VS(c,l,t,v)=\{pid(P^i,c,l),\ i=0,..,x \mid a_t^i=v\}$;
– множество на атрибутите $AS(c,l,t)$ за конкретен атрибут $t=1,...,n$ е множество на стойностите на атрибутите за клас c, слой l и атрибут t: $AS(c,l,t)=\{VS(c,l,t,1),...,VS(c,l,t,n_t)\}$, където n_t е броят стойности на атрибута t;
– пространство на връзките $LS(c,l)$ е множество от всички възможни множества на атрибутите за клас c и слой l: $LS(c,l)=\{AS(c,l,1),...,AS(c,l,n)\}$.



Фигура 5. Визуализация на пространството на връзките на клас 3 за "Lenses"

На Фигура 5 е представена визуализация пространството на връзките на клас 3 от множеството от данни "Lenses" [Frank and Asuncion, 2010].

Тази информация се съхранява в ARM-структури по много проста конвенция – множествата на атрибутните стойности $VS(c,l,t,v)$ се съхраняват в точки от ArM-архиви използвайки съответния адрес $(4,c,l,t,v)$, където 4 е размерността на ArM-пространството, където се съхраняват данните, c е номерът на класа, l е номерът на слоя, t е номерът на атрибута и v е номерът на съответната стойност на дадения атрибут. Разположението на пространствата на връзките в ArM-структурите позволява бързо извличане на съществуващите патерни от съответния слой и клас.

Множество от върхове

Множеството от върхове съдържа информация за патерните, които нямат наследници в пирамидите на съответния клас.

$$VrS(c) = \{pid(P^i,c,l) \mid i = 1,...,n_{c,l}; l = 1,...,l_{\max} : SuccS(P^i) = \varnothing\}.$$

6.4 Генериране на правилата

Процесът на генериране на правилата в рамките на всеки клас поотделно е верига от генериране на патерните от горния слой като сечение между патерните от по-долния слой. Процесът продължава дотогава докато е възможно създаването на нови патерни.

Всяка инстанция $R = (c \mid a_1,...,a_n)$ от обучаващото множество се включва в множеството от патерни на първо ниво на класа c: $R \in PS(c,1)$.

Започвайки от слой $l = 2$ се изпълняват следните стъпки:

(1) Създаване на пространството на връзките на долното ниво $l-1$ на клас c чрез добавяне на идентификаторите на патерните $P^i \in PS(c,l-1)$, $i = 1,...,n_{c,l-1}$ в множествата на стойностите на атрибутите: $pid(P^i) \in VS(c,l-1,k^i,a_k^i),\ k = 1,...,n$. Трябва да отбележим, че тези множества се създават подредени;

(2) Създават се множествата от патерни, които са пресечници на патерните от долния слой $P^i \cap P^j$, $P^i, P^j \in PS(c,l-1)$, $i,j = 1,...,k_{c,l-1}$,

$i \neq j$. Алгоритъмът на създаване на тези множества е даден в отделна точка по-долу;

(3) Ако няма създадени нови патерни (т.е. $PS(c,l) = \{\}$), процесът на генериране на правила за този клас спира;

(4) На базата на получените множества от патерни се създава множеството от патерни за слоя l $PS(c,l)$ по следния начин:

- всеки нов патерн се проверява за наличие в $PS(c,l)$;
- ако той не е съществувал в $PS(c,l)$, той получава поредния свободен номер като идентификатор; патернът се добавя в края на множеството от патерни; създава се неговото множество от предшественици $\{ (pid(P^i), l-1)$, $(pid(P^j), l-1) \}$;
- ако патернът вече е съществувал в $PS(c,l)$, неговото множество от предшественици се формира като обединение на вече съществуващото такова множество и $\{ (pid(P^i), l-1)$, $(pid(P^j), l-1) \}$.

(5) За всеки патерн от слой l се проверява съществуването на този патерн в по-долните слоеве (от $l-1$ до 2). Ако съществува такъв патерн, той се изтрива от множеството от патерни и съответното пространство на връзките от долния слой и множеството на предходниците на текущия патерн се обогатява с множеството от предходници на изтрития патерн (чрез обединение).

(6) Слоят l се увеличава с единица и процесът се повтаря.

Като резултат, за всеки клас е изградена отделна пирамидална структура. Всяка пирамида е описана от множества от предшественици и множество от наследници на патерните от съседните слоеве.

6.5 Генериране на множеството от патерни, пресечници на патерните от долния слой

За ограничаване на експоненциалния растеж при генерирането на патерните в реализацията на програмата са включени два параметъра:

- L1 (от 0 до 100) – процент на намаляване на патерните в слоя;
- L2 (от 0 до 100) –минимално съотношение в проценти между мощността на генерираните патерни към максималната мощност на патерните в долния слой.

Процесът на генериране на пресечници от патерни от определено множество от патерни $PS(c,l)$ пробягва всеки патерн $P^i \in PS(c,l)$.

За този патерн $P^i = (c \mid a_1^i, \ldots, a_n^i)$ генерирането на възможните пресечници става по следния начин:

(1) Създава се празно множество от резултатни патерни;

(2) За всички стойности на атрибути $a_1^i, \ldots, a_n^i$, различни от "-" на P^i вземаме съответните множества от стойности на атрибутите $VS(c, l, k^i, a_k^i),\ i = 1, \ldots, n$. Номерата на идентификаторите на патерните в тези множества са подредени.

(3) Активират се всички извлечени множества от стойности на атрибутите.

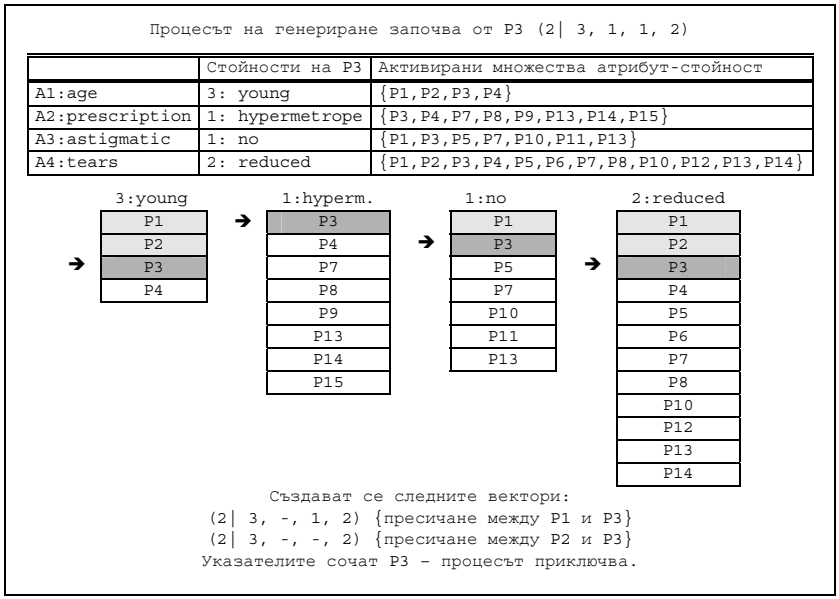
(4) От всяко от тях се взема първият идентификатор.

(5) Докато е активна поне една стойност на атрибута се извършват следните стъпки:

- присвояват се първоначални стойности на резултатния патерн: $V = (c \mid -, -, \ldots, -)$;
- намира се минималния идентификатор $pid(P^j)$ от всички активни множества от стойности на атрибутите;
- ако $pid(P^j) = pid(P^i)$, тогава това множество от стойности на атрибутите се деактивира;
- всички активни множества от стойности на атрибутите $VS(c, l, k^j, a_k^j),\ j = 1, \ldots, k_{c,l}$, за които $pid(P^j)$ е текущ идентификатор, предизвикват запълване на съответния атрибут с дадената стойност a_k^i от k^{th} атрибут в V. За тези множества се взема следващия идентификатор;
- ако $|V| > 0$ и $\min\left(\frac{|V|}{|P^i|}, \frac{|V|}{|P^j|}\right) > L2$ този патерн се включва в множеството от резултатни патерни с допълнителна информация, съдържаща $pid(P^i)$ и $pid(P^j)$.

Този процес е илюстриран на Фигура 6.

Накрая, патерните от създаденото множество от патерни се сортират по броя предшественици и L1% от тях с по-малък брой предшественици се отстраняват.



Фигура 6. Визуализация на процеса на генериране на множество от патерни

6.6 Процесът на разпознаване

Заявката за разпознаване се представя чрез стойностите на атрибутите си $Q = (?\,|\,b_1,...,b_n)$ като някои от характеристиките може да липсват. Процесът на разпознаване преминава през няколко стъпки:

(1) Системата взема съответните множества от стойности на атрибутите както и множествата за стойността "-" на всеки от атрибутите.

(2) Обединението на тези множества дава множеството от възможни класове $\{c_1,...,c_y\}$, на които записът може да принадлежи. Този подход значително намалява обема информация, който трябва да се обработи по време на разпознаването.

(3) Всички класове $\{c_1,...,c_y\}$ се сканират паралелно. За всеки клас c_x и за всеки слой от пространството на c_x се извършват следните стъпки:

– за всички стойности на атрибутите $b_1,...,b_n$ на Q, различни от "-", намираме съответните множества от стойности на атрибути от пространството на връзки на класа c_x от текущия слой;

– намира се сечението между тези множества. Като резултат се създава разпознаващото множество от патерни-кандидати. Ако множеството е празно, като отговор се дава класът с максимална подкрепа;

– за всеки патерн P, който принадлежи на разпознаващото множество, се изчислява $IntPerc(P,Q)$.

(4) От всички разпознаващи множества на класовете и слоевете се намират максималната мощност на патерните.

(5) Разпознаващите множества се олекотяват като се изхвърлят всички патерни, имащи по-ниска мощност. Новото множество от класове, които са потенциални отговори $\{c_1,...,c_{y'}\}$ съдържа само класовете, чиито (олекотени) разпознаващи множества не са празни.

(6) Ако само един клас е в множеството от класове потенциални отговори, той се дава като отговор и процесът спира.

(7) В противен случай, ако това множество е празно, вземаме отново първичното множество $\{c_1,...,c_{y'}\} = \{c_1,...,c_y\}$ и процесът продължава с разглеждане на това множество.

(8) Разглеждаме $\{c_1,...,c_{y'}\}$:

– за всеки клас, който принадлежи на множеството, се намира броят инстанции с максимален процент на пресичане със заявката и се изчислява съотношението между този брой и всички инстанции на класа;

– определя се максимумът на процентите на пресичане за всички класове и в множеството $\{c_1,...,c_{y'}\}$ се оставят само класовете с този максимален процент и максимално съотношение $\{c_1,...,c_{y''}\}$;

– ако $\{c_1,...,c_{y''}\}$ съдържа само един клас – този клас се дава за отговор. В противен случай, класът от $\{c_1,...,c_{y''}\}$ с максимален брой инстанции се дава като отговор. и процесът спира.

Възможността да се поддържат лесно създадените патерни позволява тази среда да се използва за тестване на различни видове модели на разпознаване.

Тук е описан алгоритъмът на разпознаване при стратегия S1. Алгоритъмът за стратегия S2 се различава само в точки 4 и 5, където критериите за избор на класове-кандидати са базирани на така нареченото "доверие на множеството".

7 Анализ на чувствителността

Бяха направени различни експерименти за изучаване на поведението на предложените алгоритми и за сравняване на резултатите на PGN и MPGN с резултатите от други класификатори.

Доколкото PGN и MPGN, както и повечето асоциативни класификатори, работят със символни атрибути, една част от експериментите беше насочена към избора на удобен дискретизатор, който да се прилага като предпроцесорна стъпка при класификационните алгоритми в PaGaNe.

Други експерименти проследяват как процесът на нарастване на обучаващото множество се отразява върху големината на класификационния модел и точността на разпознаване на PGN и MPGN.

Част от експериментите бяха адресирани към анализ на изходните точки на MPGN с цел да се проучи значението на различните клонове във фазата на разпознаването.

Друга част беше посветена на анализа на поведението на класификаторите при зашумяване на атрибутите в множеството от данни.

Беше направено и сравнение на PGN и MPGN с други класификатори от гледна точка на постиганата точност на разпознаване и анализ на разпознаването по отделните класове.

7.1 Обща рамка

7.1.1 Използвани множества от данни

Експериментите бяха провеждани върху множества от данни, достъпни в хранилището за данни UCI Machine Learning Repository [Frank and Asuncion, 2010]. От тях използвахме: Audiology, Balance scale, Blood transfusion, Breast cancer wo, Car, CMC, Credit, Ecoli, Forestfires, Glass, Haberman, Hayes-roth, Hepatitis, Iris, Lenses, Monks1, Monks2, Monks3, Post operative, Soybean, TAE, Tic tac toe, Wine, Winequality-red, и Zoo. Някои от множествата съдържат числови атрибути, поради което в предпроцесорната стъпка използвахме различни дискретизационни методи (Fayyad-Irani, Chi-square), реализирани в PaGaNe [Mitov et al, 2009b].

7.1.2 Провеждане на експериментите

С цел получаване на по-стабилни резултати експериментите бяха правени с прилагане на крос-валидиране. За постигане на равенство на данните при обучението и разпознаването за всички класификатори всички обучаващи и тестващи множества бяха изведени от PaGaNe (в arff-формат) и ползвани като входни файлове в останалите системи. Като алтернативни системи бяха използвани Weka [Witten and Frank, 2005] и LUCS-KDD Repository [Coenen, 2011].

Беше правено сравнение с CMAR (прагове: 1% подкрепа и 50% доверие) [Li et al, 2001] като представител на асоциативните класификатори (реализиран в LUCS-KDD Repository); OneR [Holte, 1993] и JRip (реализация на класификатора RIPPER [Cohen, 1995] във Weka) като представители на класификатори, базирани на правила на решение (decision rules); и J48 (реализация на C 4.5 [Quinlan, 1993] във Weka) и REPTree [Witten and Frank, 2005] като представители на класификатори, базирани на дървета на решение (decision trees). Изборът е продиктуван от факта, че трите типа класификатори имат подобен език на представяне на класификационния модел.

7.1.3 Анализирани конструкции

Най-популярният показател за сравняване на модели, създадени в резултат на процедурите на обучение в разглежданите типове класификатори, е броят на правилата.

Специално за MPGN има четири различни изходни точки на алгоритъма за разпознаване, свързани с различни негови части, които е интересно да бъдат изследвани.

Матрицата на объркване обикновено се прилага като основа за анализиране на резултатите от класификатори. Тя е $m \times m$ матрица, където m е броят на класовете. Редовете показват класа, към който тестващата заявка реално принадлежи. Колоните показват класа, където класификаторът е счел, че принадлежи заявката. Броят на правилно разпознатите случаи са представени по диагонала.

Най-често класификаторите се сравняват на база на получената точност (accuracy). Точността е броят на верните отговори спрямо общия брой на тестови случаи (заявки).

По-детайлният анализ се прави спрямо класовете поотделно, използвайки показатели като покриваемост (Recall), прецизност (Precision) и F-мярка (F-Measure). Покриваемостта за даден клас е броят на верните отговори спрямо действителния брой на тестовите случаи. Прецизността за даден клас е броят на верните отговори спрямо предсказания брой на тестовите случаи. F-мярката е параметър, който има за цел да извлече информация едновременно за покриваемостта и

прецизността. Тук F-мярката се изчислява като хармонична стойност на двата параметъра: $F = \frac{2 * precision * recall}{precision + recall}$.

За откриване на статистически значими разлики между класификаторите по отношение на средната точност използвахме теста на Фридман [Friedman, 1940]. Тестът на Фридман е непараметричен тест, базиран на даване на рангове на алгоритмите за всяко множество от данни. Като метод за даване на ранговете използвахме Average Ranks [Neave and Worthington, 1992]: за всяко множество алгоритмите се подреждат спрямо съответната мярка като най-добрият получава ранг 1, следващият 2 и т.н. Ако два или повече алгоритъма имат еднаква стойност, те получават еднакъв ранг, равен на средната стойност на ранговете им, които биха получили, ако бяха подредени последователно.

Нека n е броят от множествата от данни, k е броят на алгоритмите. Нека r_j^i е рангът на алгоритъм j за множеството от данни i. Средният ранг на всеки алгоритъм се изчислява като $R_j = \frac{1}{n}\sum_{i=1}^{k} r_j^i$. Спрямо нулевата хипотеза, която гласи, че всички алгоритми са еквивалентни и техните рангове R_j следва да са равни, статистиката на Фридман $\chi_F^2 = \frac{12n}{k(k+1)}\left[\sum_{j=1}^{k} R_j^2 - \frac{k(k+1)^2}{4}\right]$ е разпределена спрямо χ_F^2 с $k-1$ степени на свобода.

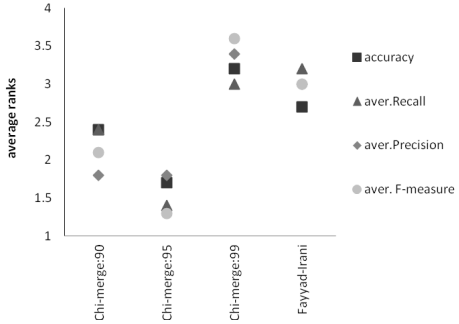
В случай на отхвърляне на нулевата хипотеза може да се продължи с прилагане на теста на Немени [Nemenyi, 1963], който се използва за сравняването на класификаторите един спрямо друг. Представянията на два класификатора значително се различават, ако разликата между съответните средни рангове е по-голяма от критичното разстояние: $CD = q_\alpha\sqrt{\frac{k(k+1)}{6n}}$ където критичните стойности q_α са базирани на статистиката на Стюдънт, разделена на $\sqrt{2}$. Резултатите от теста на Немени се представят под формата на диаграми на критичното разстояние [Demsar, 2006].

7.2 Избор на дискретизатор за предпроцесорна обработка

В това изследване използвахме следните множества от данни, които съдържат само реални атрибути: Blood transfusion, Ecoli, Forest fires, Glass, Iris. Експериментите бяха проведени чрез крос-валидиране на тройки групи. В случая бяха използвани първичните варианти на множествата (както са представени в UCI Repository).

Бяха изследвани дискретизаторите Chi-merge с ниво на значимост 90%, 95% и 99% и Fayyad-Irani, който е непараметричен метод.

Анализът на получените резултати показа, че Chi-merge с ниво на значимост 95% дава най-добри резултати. Близко до него е Chi-merge с 90% ниво на значимост. Chi-merge с 99% ниво на значимост дава доста лоши резултати заради прекаленото раздробяване на интервалите. Методът Fayyad-Irani в някои случаи дава много добри резултати, но се проваля при други множества от данни.



Фигура 7. Сравнение на различните дискретизационни методи

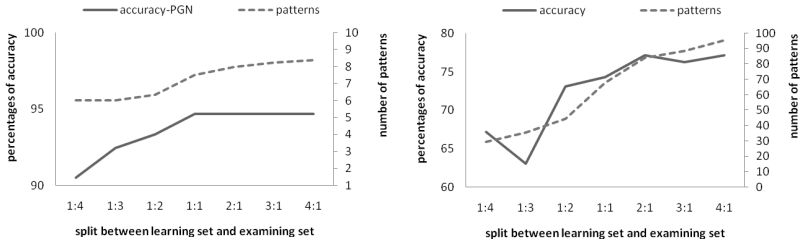
Независимо от факта, че тестът на Фридман в случая не показа статистическа значима различимост ($\chi_F^2 = 3.54$, $\alpha_{0.05} = 6.251$), като цяло може да направим извода, че за предпроцесорната стъпка като най-подходящ дискретизатор за следващите експерименти е най-добре да използваме дискретизатора Chi-merge с 95% ниво на значимост.

7.3 Изучаване на размера на обучаващото множество

Целта на тази част от изследването е да анализира зависимостта на точността на разпознаване от размера на обучаващото множество.

Например, един от експериментите беше направен върху множеството от данни "Iris" (Фигура 8а). Както може да видим,

половината от инстанциите са достатъчни, за да получим стабилно и добро разпознаване за това множество. Създаденият модел се състои от около 8 патерна.



а) "Iris" *б) "Glass"*

Фигура 8. Брой патерни и точност за класификатора PGN при различна големина на обучаващото множество

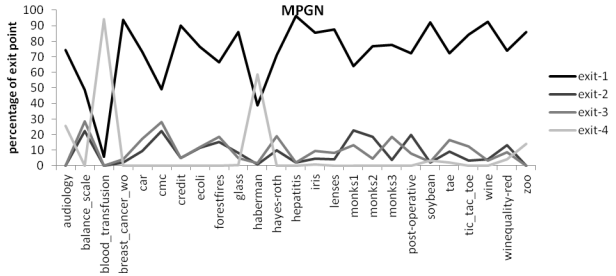
Друг експеримент беше проведен върху множеството от данни "Glass" (Фигура 8б). За това множество добро разпознаване с относително малък брой патерни се постига в случая на около 140 инстанции. Увеличаването на големината на обучаващото множество увеличава размера на множеството от патерни без да носи по-добро разпознаване.

Броят инстанции в обучаващото множество, което е достатъчно за постигането на добра точност и стегнато множество от патерни, силно зависи от конкретното множество от данни. PGN е непараметричен метод, но е препоръчително потребителят да експериментира с различна големина на обучаващото множество, за да намери компромис между размера на модела (простота) и постиганата точност. В бъдещи изследвания би могло да се разработи допълнителна процедура, която да решава този проблем чрез използване на принципа за описание чрез минимална дължина (Minimum Description Length Principle). Забелязахме също, че за някои множества от данни точността се стабилизира, но се увеличава броят на патерните, което е индикация, че в бъдещите изследвания би могло да се подобри частта по орязването в PGN.

7.4 Анализ на изходящите точки в MPGN

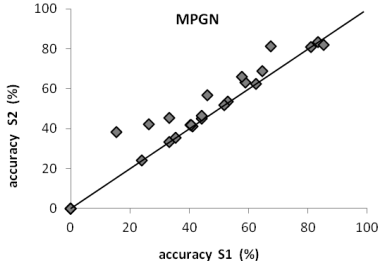
При проектирането на MPGN ние заложихме на идеята, че пирамидалната мрежа съдържа достатъчно информация за извършване на класифицирането. Очакването ни беше, че мнозинството от случаите ще бъдат класифицирани благодарение на използването на структурата. С други думи, очакваме, че само в редки случаи при

разпознаването системата ще попада на празно множество от потенциални отговори (Exit 4). Интерес представляваше и изследването в колко случаи разпознаването става само чрез един клас (Exit 1) и в колко случаи се явяват множество противоречиви класове, където се налага да се намесват други критерии за оценка като доверието (Exit 2) или подкрепата (Exit 3). Фигура 9 илюстрира процента от различни случаи на изходи. Както виждаме най-често разпознаването води към изход 1, което означава, че прилагането на пирамидалните структури на MPGN е оправдано.



Фигура 9. Изходящи точки за MPGN

Направихме и по-детайлно изследване на двете стратегии, предложени в MPGN в случая на конкуриращи се класове-кандидати (Exit 2 и Exit 3). Фигура 10 представя графика на постигнатите точности от двете стратегии една спрямо друга. Оценката на ранговете на двете стратегии дава следните резултати: S2 има средна точност 49%, а средната точност при S1 е 46%. Тестът на Фридман показва $\chi_F^2 = 2.56$, $\alpha_{0.05} = 1.960$, което означава, че MPGN-S2 статистически различимо превъзхожда MPGN-S1.



Фигура 10. Постигнати точности от MPGN-S1 и MPGN-S2 една спрямо друга

В заключение, анализът на различните части на разпознаващия модел на MPGN показва, че началната идея за използване на пирамидалните структури работи добре и в масовия случай разпознаването води еднозначно до един клас.

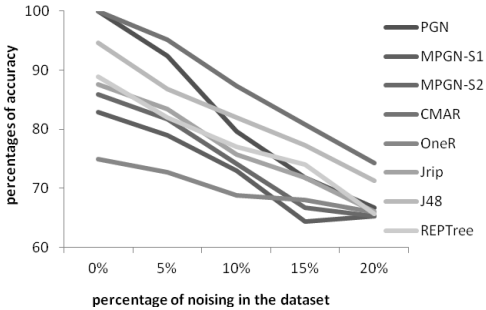
7.5 Шумът в множествата от данни

Във фазата на орязването се изтриват противоречивите патерни между класовете (а в случая на PGN и по-общите патерни, за които е налично изключение в друг клас) без да се гледа наличието на шум или честотата на срещане на противоречивия патерн. Това прави алгоритмите неустойчиви към наличието на шум в множествата от данни. Получената точност се определя от два важни фактора: индуктивното отклонение на обучаващия алгоритъм както и от качеството на обучаващите данни.

Като цяло има два вида на източници на шум [Wu, 1995]: шум в атрибутите (грешки, възникнали в стойностите на атрибутите в инстанциите) и шум в класовете (наличие на противоречиви примери или погрешно етикетирани инстанции).

Направихме експерименти с изкуствено зашумяване на атрибутите с цел проучване на надеждността на класификаторите PGN и MPGN.

Зашумяването на множествата от данни беше направено чрез избор на произволна инстанция и произволен атрибут и промяна на стойността му с произволно избрана друга възможна стойност. Системата пази информация кои инстанции и атрибути вече са били променяни и не прави нова смяна в същите позиции. Такива замени се правят докато се достигне желания процент зашумяване.



Фигура 11. Точност на различните класификатори в зашумените множества от данни, базирани на "Monks1"

За основа на провеждане на експеримента избрахме множеството от данни "Monks1" защото в това множество няма шум и разпределението на класовете е равномерно. Направихме зашумяване на атрибутите съответно с 5, 10, 15 и 20%.

Фигура 11 показва, че всички класификатори имат приблизително подобно поведение на намаляване на точността при поява на шум в множеството от данни. В този експеримент най-добре се представя CMAR. Много стабилен е също и J48. PGN и MPGN са чувствителни към появата на шум, което потвърждава нашата хипотеза, че подходът на даване на приоритет на доверието спрямо подкрепата има своите недостатъци при зашумени множества от данни.

7.6 Сравнение с други класификатори

Направихме сравнение на предложените класификатори PGN и MPGN (с двете стратегии на разпознаване S1 и S2) с CMAR, OneR, JRip, J48 и REPTree в две направления: обща точност и F-мярка.

7.6.1 Сравнение на общата точност

В Таблица 1 са представени процентите на общата точност, постигната от разглежданите класификатори. С цел подготовка за прилагане на теста на Фридман Таблица 2 представя назначаването на рангове на класификаторите за изследваните множества от данни.

Таблица 1. Проценти на общата точност, постигната от разглежданите класификатори

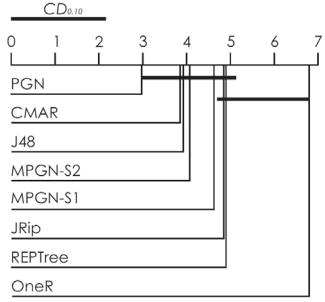
Множество от данни	PGN	MPGN-S1	MPGN-S2	CMAR	OneR	JRip	J48	REPTree
audiology	75.50	69.00	69.00	59.18	47.00	69.50	72.00	62.50
balance_scale	77.89	81.41	83.49	86.70	60.10	71.95	66.18	67.15
breast_cancer_wo	96.43	92.85	93.56	93.85	91.85	93.28	94.28	93.99
car	92.59	82.87	85.71	81.77	70.03	86.75	90.80	88.20
cmc	49.90	46.03	46.64	53.16	47.25	50.38	51.60	50.17
credit	87.54	85.65	86.09	87.10	85.51	85.07	85.36	85.07
haberman	55.27	73.21	73.21	71.90	72.88	73.21	73.21	74.20
hayes-roth	81.94	65.22	67.49	83.42	50.77	78.12	68.23	73.53
hepatitis	80.65	81.94	81.94	84.52	81.94	77.42	79.36	79.36
lenses	74.00	83.00	83.00	88.00	62.00	83.00	83.00	80.00
monks1	100.00	82.9	85.92	100.00	74.98	87.53	94.68	88.91
monks2	73.06	80.52	81.02	59.74	65.73	58.73	59.90	63.90
monks3	98.56	90.43	93.50	98.92	79.97	98.92	98.92	98.92
post-operative	66.67	52.22	52.22	51.11	68.89	70.00	71.11	71.11
soybean	93.15	84.00	84.00	78.48	37.44	85.35	87.64	78.18
tae	52.94	52.88	52.88	35.74	45.76	34.43	46.97	40.43
tic_tac_toe	88.93	96.13	95.62	98.75	69.93	98.02	84.23	80.37
wine	96.09	92.19	93.87	91.70	78.63	90.45	87.03	88.16
winequality-red	64.98	59.35	59.60	56.29	55.54	53.72	58.22	57.03
zoo	98.10	89.24	89.24	94.19	73.29	88.19	94.14	82.19

Таблица 2. Рангове спрямо постигнатата точност на разглежданите класификатори

Множество от данни	PGN	MPGN-S1	MPGN-S2	CMAR	OneR	JRip	J48	REPTree
audiology	1	4.5	4.5	7	8	3	2	6
balance_scale	4	3	2	1	8	5	7	6
breast_cancer_wo	1	7	5	4	8	6	2	3
car	1	6	5	7	8	4	2	3
cmc	5	8	7	1	6	3	2	4
credit	1	4	3	2	5	7.5	6	7.5
haberman	8	3.5	3.5	7	6	3.5	3.5	1
hayes-roth	2	7	6	1	8	3	5	4
hepatitis	5	3	3	1	3	8	6.5	6.5
lenses	7	3.5	3.5	1	8	3.5	3.5	6
monks1	1.5	7	6	1.5	8	5	3	4
monks2	3	2	1	7	4	8	6	5
monks3	5	7	6	2.5	8	2.5	2.5	2.5
post-operative	5	6.5	6.5	8	4	3	1.5	1.5
soybean	1	4.5	4.5	6	8	3	2	7
tae	1	2.5	2.5	7	5	8	4	6
tic_tac_toe	5	3	4	1	8	2	6	7
wine	1	3	2	4	8	5	7	6
winequality-red	1	3	2	6	7	8	4	5
zoo	1	4.5	4.5	2	8	6	3	7
средно	**2.975**	**4.625**	**4.075**	**3.85**	**6.8**	**4.85**	**3.925**	**4.9**

Критичната стойност за нулева хипотеза за теста на Фридман е $\alpha_{0.05} = 14.067$; в нашия случай $\chi_F^2 = 29.492$, което означава, че нулевата хипотеза се отхвърля, т.е. класификаторите са статистически различни.

Отхвърлянето на нулевата хипотеза на теста на Фридман ни дава основание да приложим теста на Немени. В този случай $CD_{0.10} = 2.153$.



Фигура 12. Визуализация на резултатите от теста на Немени

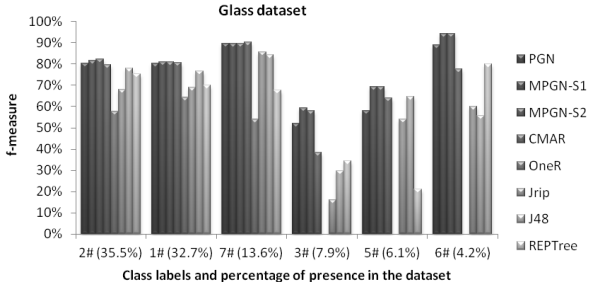
Фигура 12 показва резултатите от теста на Немени. Всички групи от класификатори, които са статистически неразличими, са свързани с линия. Резултатите показват, че PGN има най-добро общо представяне измежду разглежданите класификатори. MPGN е сравним с J48, JRip и REPTree. Първите четири класификатора (PGN, CMAR, J48, и MPGN-S2) значително превъзхождат OneR.

7.6.2 Анализ на F-мерките на множества от данни с много класове

Общата точност на разпознаване е добър критерий за оценка на класификаторите. В случаите на множества от данни с много класове и неравномерно разпределение на инстанциите е необходимо да се направи и по-точен анализ като се използва F-мярката, за да се оцени колко прецизно и пълно класификаторът разпознава конкретните класове, които са слабо представени. Като примери са използвани множествата от данни "Glass" и "Soybean".

Детайлен анализ върху "Glass"

В множеството от данни "Glass" има 6 класа с много неравномерно разпределение между тях: 2#: 35.51%; 1#: 32.71%; 7#: 13.55%; 3#: 7.94%; 5#: 6.07%; 6#: 4.21%.

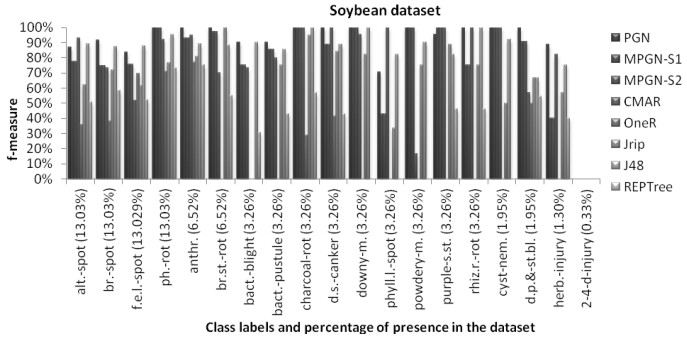


Фигура 13. F-мярката на разглежданите класификатори за класовете на "Glass"

Фигура 13 представя F-мерките за всеки клас, получени от различните класификатори. Както може да видим PGN и MPGN имат добро представяне за всеки клас. Например, слабо представеният клас "3#" не се разпознава добре от CMAR, OneR, JRip, J48 и REPTree; "5#" – от OneR и REPTree; "6#" – от OneR (F-мерките им са по-малки от 50%).

Детайлен анализ върху "Soybean"

Множеството от данни "Soybean" има 19 класа с различно разпределение – от 13.029% до 0.326%. Фигура 14 показва F-мерките за всеки клас при различните класификатори. Тук класът с най-малка подкрепа "2-4-d-injury" не се разпознава защото има само 1 инстанция – тя попада или в обучаващото множество, или в тестовото множество. Както може да видим PGN разпознава успешно всички останали класове независимо от разликите в тяхната подкрепа. J48 също разпознава добре, но се проваля в слабо представените класове. MPGN има относително добро представяне.



Фигура 14. F-мярката на разглежданите класификатори за класовете на "Soybean"

Разглеждайки резултатите от общата точност на разпознаване, както и от по-финия анализ на поведението на класификаторите върху множества от данни с много класове и неравномерно разпределение можем да заключим, че се оправдават нашите очаквания, че стратегията на PGN да фокусира върху доверието вместо върху подкрепата има потенциал да създаде удачни асоциативни алгоритми.

8 Заключение

Основните цели на дисертационната работа бяха:

- представяне на един асоциативен класификатор без параметри, който се фокусира основно върху доверието при изграждането на асоциативните правила и едва в по-късен етап на подкрепата на правилата;
- показване на предимствата от използването на многомерните номерирани информационни пространства като структури на паметта в процесите по извличане на знания на примера на създадения асоциативен класификатор.

Основните приноси в дисертационната работа може да се обобщят по следния начин:

- предложен е нов асоциативен класификатор PGN, който поставя под въпрос общия подход на даване на приоритет на подкрепата спрямо доверието и се фокусира първо върху доверието на асоциативните правила, а само в по-късен етап на подкрепата на правилата;
- разработен е метод за ефективно изграждане и съхраняване на множеството от правила в многослойна структура MPGN, използвайки възможностите на многомерните номерирани информационни пространства.
- предложените алгоритми и структури са реализирани в рамките на средата за извличане на данни PaGaNe;
- проведените експерименти доказват жизнеността на предложените подходи, показвайки добро представяне на PGN и MPGN в сравнение с други класификатори от групите, базирани на правила, особено в случаите на множества от данни с много класове и с неравномерно разпределение.

Насоки за бъдещо развитие

Тази работа определи някои възможни посоки за по-нататъшни изследвания като:

- реализация на PGN изрязването и разпознаването върху пирамидалните структури на MPGN;
- подобряване на изрязването в PGN с цел, отчитане на явлението, че от определен момент увеличаването на размера на обучаващото множество увеличава големината на множеството от патерни, но без повишаване на точността;
- обсъждане на различни техники за мярка за качество на правилата, с цел преодоляване на процеса на отхвърляне на едно правило за сметка на друго, което се среща рядко;
- разширяване на функционалността на PaGaNe в посока на автоматично избиране на подмножеството от атрибути;
- прилагането на PGN в различни практически области.

9 Публикации

Обзорите, идеите, реализацията и експериментите, включени в дисертацията са представени на следните научни форуми и публикувани в съответните томове на конференции или списания:

1. Mitov, I., Ivanova, K., Markov, K., Velychko, V., Vanhoof, K., Stanchev, P.: PaGaNe – A Classification Machine Learning System Based on the Multidimensional Numbered Information Spaces.
   Presented at Int.Conf. "Intelligent Systems and Knowledge Engineering" (ISKE2009), 27-28.11.2009, Hasselt, Belgium.
   Published in World Scientific Proceedings Series on Computer Engineering and Information Science, No:2, pp.279-286,
   ISBN: 978-981-4295-05-5.
2. Mitov I., Ivanova, K., Markov, K., Velychko, V., Stanchev, P., Vanhoof. K.: Comparison of Discretization Methods for Preprocessing Data for Pyramidal Growing Network Classification Method.
   Presented at Int.Conf. i.Tech-2009, Madrid, Spain, 02-05.09.2009.
   Published in Int. Book Series "Information Science and Computing" – Book No: 14. New Trends in Intelligent Technologies, Sofia, 2009, pp.31-39, ISSN: 1313-0455.
3. Mitov, I.: MPGN – An Approach for Discovering Class Association Rules. Serdica Journal of Computing, Vol.5, No.4, 2011, pp.385-414, ISSN: 1312-6555.
4. Ivanova, K., Mitov, I., Stanchev, P., Dobreva, M., Vanhoof, K., Depaire, B.: Applying Associative Classifier PGN for Digitised Cultural Heritage Resource Discovery.
   In Proceedings of the First International Conference "Digital Preservation and Presentation of Cultural Heritage", Veliko Tarnovo, Bulgaria, 11-14.09.2011, IMI-BAS, Sofia, 2011, pp.117-126, ISSN: 1314-4006.
5. Depaire, B., Ivanova, K., Markov, K., Mitov, I., Vanhoof, K., Velychko, V.: Multi-dimensional Information Spaces as Memory Structures for Intelligent Data Processing in GMES.
   Chapter 15 in the book "Intelligent Data Processing in Global Monitoring for Environment and Security", Sofia, 2011, pp.347-370, ISBN: 978-954-16-0045-0.

10 Литература

[Agrawal and Srikant, 1994] Agrawal, R., Srikant, R.: Fast algorithms for mining association rules. In Proc. 20th Int. Conf. Very Large Data Bases, 1994, pp.487-499.

[Agrawal et al, 1993] Agrawal, R., Imieliński, T., Swami, A.: Mining association rules between sets of items in large databases. Proc. of the ACM SIGMOD Int. Conf. on Management of Data, Washington, DC, 1993, pp. 207-216.

[Antonie et al, 2006] Antonie, M.-L., Zaiane, O., Holte, R.: Learning to use a learned model: a two-stage approach to classification. In 6th IEEE Int. Conf. on Data Mining (ICDM'06), Hong Kong, China, 2006, pp.33-42.

[Aslanyan and Sahakyan, 2010] Aslanyan, L., Sahakyan, H.:On structural recognition with logic and discrete analysis. Int. J. Information Theories and Applications, 17/1, 2010, pp.3-9.

[Bayardo, 1998] Bayardo, R.: Efficiently mining long patterns from databases. In Proc. of the ACM SIGMOD Int. Conf. on Management of Data, Seattle, Washington, United States, 1998, pp.85-93.

[Coenen and Leng, 2005] Coenen, F., Goulbourne, G., Leng, P.: Tree structures for mining association rules. Data Mining and Knowledge Discovery, Kluwer Academic Publishers, 8, 2004, pp.25-51.

[Coenen, 2011] Coenen, F.: Liverpool University of Computer Science – Knowledge Discovery in Data. http://www.csc.liv.ac.uk/~frans/KDD/ last visited: 22.11.2011

[Cohen, 1995] Cohen, W.: Fast effective rule induction. In Proc. of the 12th Int. Conf. on Machine Learning, Lake Taho, California, Morgan Kauffman, 1995.

[Demsar, 2006] Demsar, J.: Statistical comparisons of classifiers over multiple data sets. J. Mach. Learn. Res., 7, 2006, pp.1-30.

[Depaire et al, 2008] Depaire, B., Vanhoof, K., Wets, G.: ARUBAS: an association rule based similarity framework for associative classifiers. IEEE Int. Conf. on Data Mining Workshops, 2008, pp.692-699.

[Frank and Asuncion, 2010] Frank, A. Asuncion, A.: UCI Machine Learning Repository http://archive.ics.uci.edu/ml. Irvine, CA: University of California, School of Information and Computer Science, 2010.

[Friedman, 1940] Friedman, M.: A comparison of alternative tests of significance for the problem of m rankings. Annals of Mathematical Statistics, Vol. 11, 1940, pp.86-92.

[Han and Pei, 2000] Han, J., Pei, J.: Mining frequent patterns by pattern-growth: methodology and implications. ACM SIGKDD Explorations Newsletter 2/2, 2000, pp.14-20.

[Holte, 1993] Holte, R.: Very simple classification rules perform well on most commonly used datasets. Machine Learning, Vol. 11, 1993, pp.63-91.

[Kuncheva, 2004] Kuncheva, L.: Combining Pattern Classifiers: Methods and Algorithms. Willey, 2004.

[Li et al, 2001] Li, W., Han, J., Pei, J.: CMAR: Accurate and efficient classification based on multiple class-association rules. In: Proc. of the IEEE Int. Conf. on Data Mining ICDM, 2001, pp.369-376.

[Liu et al, 1998] Liu, B., Hsu, W., Ma, Y.: Integrating classification and association rule mining. In Knowledge Discovery and Data Mining, 1998, pp.80-86.

[Liu et al, 2003] Liu, G., Lu, H., Lou, W., Yu, J.: On computing, storing and querying frequent patterns. Proc. of the 2003 ACM SIGKDD international conference on knowledge discovery and data mining (KDD'03), Washington, DC, 2003, pp.607-612

[Markov, 1984] Markov K.: A multi-domain access method. Proc. of the Int. Conf. on Computer Based Scientific Research. Plovdiv, 1984, pp.558-563.

[Markov, 2004] Markov, K.: Multi-domain information model. Int. J. Information Theories and Applications, 11/4, 2004, pp.303-308.

[Markov, 2005] Markov, K.: Building data warehouses using numbered multidimensional information spaces. Int. J. Information Theories and Applications, 2005, 12/2, pp.193-199.

[Mitov et al, 2009a] Mitov, I., Ivanova, K., Markov, K., Velychko, V., Vanhoof. K., Stanchev, P.: "PaGaNe" – A classification machine learning system based on the multidimensional numbered information spaces. In World Scientific Proc. Series on Computer Engineering and Information Science, No.2, pp.279-286.

[Mitov et al, 2009b] Mitov, I., Ivanova, K., Markov, K., Velychko, V., Stanchev, P., Vanhoof, K.: Comparison of discretization methods for preprocessing data for pyramidal growing network classification method. In Int. Book Series Information Science & Computing – Book No: 14. New Trends in Intelligent Technologies, 2009, pp. 31-39.

[Mitov, 2011] Mitov, I.: Class Association Rule Mining Using Multi-Dimensional Numbered Information Spaces. PhD Thesis, Hasselt University, Belgium (2011)

[Morishita and Sese, 2000] Morishita, S., Sese, J.: Transversing itemset lattices with statistical metric pruning. In Proc. of the 19th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, 2000, pp.226-236.

[Neave and Worthington, 1992] Neave, H., Worthington, P.: Distribution Free Tests. Routledge, 1992.

[Nemenyi, 1963] Nemenyi, P.: Distribution-free multiple comparisons. PhD thesis, Princeton University, 1963

[Quinlan and Cameron-Jones, 1993] Quinlan, J., Cameron-Jones, R.: FOIL: A midterm report. In Proc. of European Conf. On Machine Learning, Vienna, Austria, 1993, pp.3-20.

[Quinlan, 1993] Quinlan, R.: C4.5: Programs for Machine Learning. Morgan Kaufmann Publishers, 1993.

[Rak et al, 2005] Rak, R., Stach, W., Zaiane, O., Antonie M.-L.: Considering re-occurring features in associative classifiers. In Advances in Knowledge Discovery and Data Mining, LNCS, Vol. 3518, 2005, pp.240-248.

[Tang and Liao, 2007] Tang, Z., Liao, Q.: A new class based associative classification algorithm. IAENG International Journal of Applied Mathematics, 2007, 36/2, pp.15-19.

[Thabtah et al, 2005] Thabtah, F., Cowling, P., Peng, Y.: MCAR: multi-class classification based on association rule. In Proc. of the ACS/IEEE 2005 Int. Conf. on Computer Systems and Applications, Washington, DC, 2005, p.33.

[Wang and Karypis, 2005] Wang, J., Karypis, G.: HARMONY: Eficiently Mining the Best Rules for Classification. In Proc. of SDM, 2005, pp.205-216.

[Witten and Frank, 2005] Witten, I., Frank, E.: Data Mining: Practical Machine Learning Tools and Techniques. 2nd Edition, Morgan Kaufmann, San Francisco, 2005.

[Wu, 1995] Wu, X.: Knowledge Acquisition from Database. Ablex Pulishing Corp., USA, 1995.

[Yin and Han, 2003] Yin, X., Han, J.: CPAR: Classification based on predictive association rules. In SIAM Int. Conf. on Data Mining (SDM'03), 2003, pp.331-335.

[Zaïane and Antonie, 2002] Zaiane, O., Antonie, M.-L.: Classifying text documents by associating terms with text categories. J. Australian Computer Science Communications, 24/2, 2002, pp.215-222.

[Zimmermann and De Raedt, 2004] Zimmermann, A., De Raedt, L.: CorClass: Correlated association rule mining for classification. In Discovery Science, LNCS, Vol. 3245, 2004, pp.60-72.

Съдържание