УДК 61:004.651(075.8)
Наконечний О.Г.[1)], Марценюк В.П.[2)], Андрущак І.Є.[3)]
[1)]Київський національний університет імені Тараса Шевченка,
[2)]Тернопільський державний медичний університет імені І.Я.Горбачевського,
[3)]Луцький національний технічний університет

АЛГОРИТМ ПОБУДОВИ ІНДУКЦІЇ ДЕРЕВА РІШЕНЬ З ВИКОРИСТАННЯМ ПОКАЗНИКА ПРИРОСТУ ІНФОРМАЦІЇ

Наконечний О.Г., Марценюк В.П., Андрущак І.Є. Алгоритм побудови індукції дерева рішень з використанням показників приросту інформації. У роботі розглядаються питання програмної реалізації методу індукції дерева рішень на основі показника відношення приростів інформації. Даний підхід порівнюється з традиційним показником приросту інформації при розробці систему підтримки клінічних рішень. Проект реалізовано в середовищі Netbeans на основі Java-класів.
Ключові слова: прийняття рішень, дерево рішень, Java, SQL.

Наконечный А.Г., Марценюк В.П., Андрущак И.Е. Алгоритм построения индукции дерева решений с использованием показателей прироста информации. В работе рассматриваются вопросы программной реализации метода индукции дерева решений на основе показателя отношения приростов информации. Данный подход сравнивается с традиционным показателем прироста информации при разработке системы поддержки клинических решений. Проект реализован в среде Netbeans на основе Java-классов.
Ключевые слова: принятие решений, дерево решений, Java, SQL.

Nakonechnyy O., Martsenyuk V., Andruschak I. Induction algorith for constructing decision trees using growty rates information. The paper deals with the software implementation of the method of decision tree induction is based on the ratio of growth media. This approach is compared with the traditional indicator of growth in developing information system to support clinical decisions. The project was implemented in an environment based on Netbeans Java-classes.
Keywords: decision making, decision tree, Java, SQL.

Вступ. Одним з підходів, що відображає природній процес мислення при прийнятті рішень є метод індукції дерева рішень, який знайшов реальні практичні застосування, зокрема в диференціальній діагностиці в медицині. Алгоритм грунтується на виборі визначальних атрибутів, за допомогою яких будуються класифікаційні правила в задачах класифікації або прогнозування [Quinlan, 1986], [Breiman, 1984].

Традиційний метод індукції дерева рішень грунтується на обчисленні показника приросту інформації. Як вже було попередньо зазначено [Han, 2001], показник приросту інформації $Gain(A_i)$ має ухил в сторону тестів з багатьма виходами. Тобто він надає перевагу атрибутам, які мають дуже велике число значень.

Тому метою даної роботи є реалізація алгоритму індукції дерева рішень, який би був позбавлений такого недоліку через використання показника відношення приростів інформації.

Метод на основі відношення приростів інформації. Для прикладу розглянемо як атрибут ідентифікатор id з таблиці навчальних даних categorised_data. Поділ набору навчальних даних по даному атрибуту призведе до великої кількості розбиттів (дорівнює кількості пацієнтів), кожне з яких включатиме лише один запис. Оскільки кожне розбиття не потребує подальшого поділу, то інформація, яка необхідна для класифікації даних в D_j на основі цього розбиття становить $Info_{id}(D_j) = 0$ (це можна показати і з формул для обчислення $Info_{A_i}(D_j)$ та $Info(D_j)$ також). Отже, приріст інформації внаслідок розбиття по цьому атрибуту є максимальним. З іншої сторони, таке розбиття є безглуздим з точки зору класифікації.

З метою подолання такого недоліку в методі C4.5 [Han, 2001] використовується удосконалений показник на основі приросту інформації, який називається відношення приростів (gain ratio). Він застосовує до приросту інформації свого роду нормалізацію, використовуючи значення «розділяючої інформації», яке визначається за аналогією до $Info(D)$:

$$SplitInfo_{A_i}(D_j) = -\sum_{l=1}^{K_i} \frac{\#(D_j^l)}{\#(D_j)} \log_2 \left(\frac{\#(D_j^l)}{\#(D_j)} \right). \quad (1)$$

Таке значення представляє потенційну інформацію, яка генерується при поділі набору навчальних даних D_j на K_i розбиттів, що відповідають K_i виходам тесту щодо атрибуту A_i. Зауважимо, що для кожного виходу розглядається кількість записів, що відповідають умові даного виходу відносно загального числа записів в D_j. Це відрізняє його від показника приросту інформації, в якому вимірюється інформація щодо класифікації, яка отримується на основі того ж розбиття. Згідно означення відношення приростів становить:

$$GainRatio(A_i) = \frac{Gain(A_i)}{SplitInfo(A_i)}. \quad (2)$$

Атрибут з максимальним значенням відношення приростів обирається як атрибут поділу. Хоча, як зазначають в [Han, 2001] у випадку, коли інформація поділу наближається до нуля, то значення відношення приростів стає нестійким. Тому слід накладати відповідні обмеження на значення інформації поділу.

В цілому отримуємо таку рекурсивну процедуру.

Генерація дерева рішень

Вхідні дані: D – множина навчальних наборів даних $(A_1^i, A_2^i, ..., A_p^i, C^i)$.

Вихідні дані: дерево рішень

Метод:

1. Створити вузол N.

2. Якщо усі набори в D належать до спільного класу C, тоді повернути вузол N як листок із назвою класу C.

3. Якщо список атрибутів (а отже і D) є порожнім, тоді повернути вузол N як листок із назвою найпоширенішого класу в D.

4. Застосувати *Алгоритм відбору атрибуту* із списку атрибутів і для множини D (на основі показника відношення приростів інформації) з метою відшукання «найкращого» атрибуту поділу.

5. Вилучити атрибут поділу із списку атрибутів.

6. Для кожної умови поділу j для атрибуту поділу розглянемо D_j – множину наборів з D, що задовольняють умову поділу j.

7. Якщо D_j – порожня, тоді приєднати до вузла N листок під заголовком наявних в D_j класів з вказуванням їх розподілів, інакше – приєднати до N вузол, що повертається рекурсивним викликом методу *Генерація дерева рішень* з вхідними даними D_j та список атрибутів.

8. Кінець циклу кроку 6.

9. Повернути вузол N.

SQL-реалізація розрахунку інформаційних показників

При реалізації методу Attribute_selection_method слід розрахувати інформаційні показники $Info(D_j)$, $Info_{A_i}(D_j)$ та $SplitInfo_{A_i}(D_j)$ на j-му кроці рекурсії для атрибута A_i.

Зауважимо, що множина наборів D_j тут описується хеш-таблицею htAttribute_list, з якої отримуємо перелік включених атрибутів sAttribute_list та умов поділу sConditions.

Використавши вкладені запити, псевдоніми та агрегативні функції можна розрахувати показник інформації поділу $SplitInfo_{A_i}(D_j)$ в результаті виконання такого запиту:

```
sql = "select -SUM((Alias1.Dj/Alias2.D)*(LOG(Alias1.Dj/Alias2.D)/LOG(2))) from (select SUM(1) as
Dj from (select * from categorised_data " + (sConditions.matches("")?"": " where " + sConditions) +
")Alias3 group by Alias3." +
((Attribute_for_list)htAttribute_list.get(A)).attribute.getAttributeFieldName() + ")Alias1, (select SUM(1)
```

as D from (select " + sAttribute_list + " from categorised_data " + (sConditions.matches(""))?"": " where " + sConditions) + ")Alias4)Alias2".

Зауважимо, що тут дещо змінено крок 7 методу генерації дерева рішень [Han, 2001], а саме, якщо D_j – порожня, тоді пропонується приєднати до вузла N листок під заголовком наявних в D_j класів з вказуванням їх розподілів. З цією метою в класі AttributeListPeer розроблено метод:

public String Classes_ratio_in_D (DataManager dataManager, Hashtable htAttribute_list).

В результаті виконання запиту:

String sql = "select Alias1.class_name, (Alias1.count_class / Alias2.count_tuples)*100 from (select DISTINCT(class) as class_name,SUM(1) as count_class from (select * from categorised_data " + (sConditions.matches(""))?"": " where " + sConditions) + ")Alias3 group by class) Alias1, (select SUM(1) as count_tuples from (select * from categorised_data " + (sConditions.matches(""))?"": " where " + sConditions) + ")Alias4) Alias2";

метод повертає перелік наявних в D_j класів з вказуванням їх розподілів.

Приклад. Метою є побудувати дерево рішень щодо діагностування серцевої недостатності на основі даних 63-х пацієнтів навчально-практичного центру первинної медико-санітарної допомоги Тернопільського державного медичного університету імені І.Я.Горбачевського. Використано таку таблицю атрибутів:

INSERT INTO mysql.attribute (id, attribute_name, attribute_field_name) VALUES (1, 'What is age?', 'A1'), (2, 'What is sex?', 'A2'), (3, 'What is pulse?', 'A3'), (4, 'What is SAP?', 'A4'), (5, 'What is DAP?', 'A5');

Набори включають лише категоріальні дані (попередньо оброблені), наприклад:

INSERT INTO mysql.categorised_data (id, A1, A2, A3, A4, A5, class) VALUES (1,'senior','female','normal','high','high','healthy').

На рис.2 представлене побудоване дерево рішень. Час, затрачений на індукування дерева – 1233 мілісекунд, що більше порівняно з показником приросту інформації. Це пов'язано з тим, що здійснюється розрахунок ще одного показника, а саме – розділяючої інформації $SplitInfo_{A_i}(D_j)$.

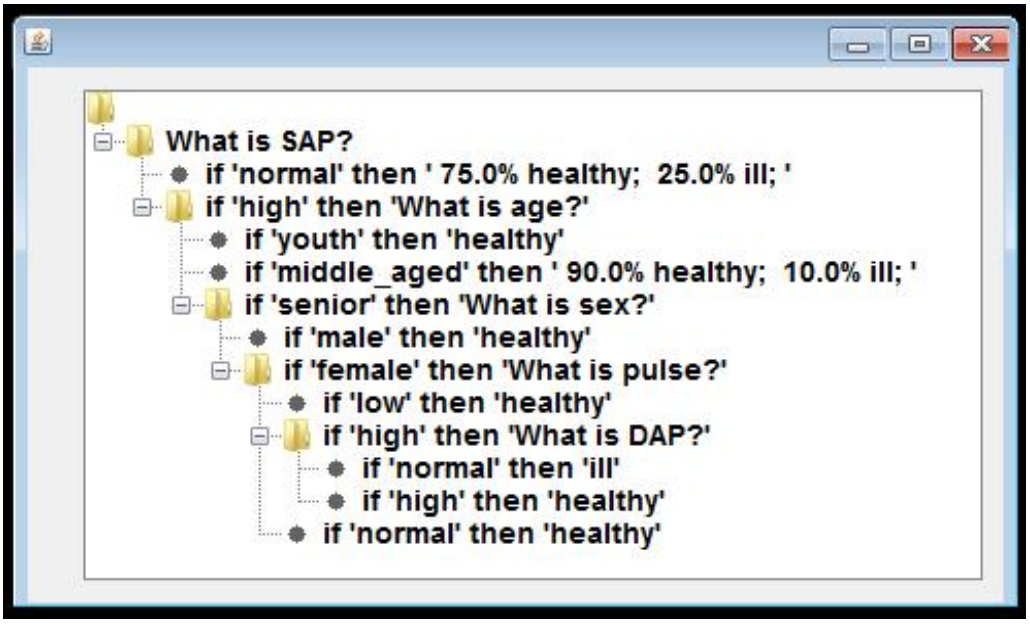

Рис.2. Дерево рішень для діагностування серцевої недостатності на основі показника відношення приростів інформації

Висновки. У роботі розглянуто питання програмної реалізації методу індукції дерева рішень на основі показника відношення приростів інформації.

Аналізуючи отримані дерева рішень на основі показника відношення приростів інформації, бачимо, що змінився порядок ієрархічності атрибутів. Так вдалося позбавитися ухилу в сторону атрибутів з більшим числом категоріальних значень, що спостерігається при використанні показника інформаційного приросту.

У перспективі бачиться необхідним вивчення питання обчислювальної складності запропонованого методу.

1. J.Han and M.Kamber, Data Mining: Concepts and Techniques, Morgan Kaufmann, San Francisco, 1st edition, 2001.
2. T.Hastie, R.Tibshirani and J.H.Friedman, The Elements of Statistical Learning, Springer, New York, 1st edition, 2001.
3. C.Ordonez, Comparing association rules and decision trees for disease prediction, In Proc. ACM HIKM Workshop, 2006, pp. 17-24.
4. C.Ordonez, Integrating K-means clustering with a relational DBMS using SQL, IEEE Transactions on Knowledge and Data Engineering (TKDE) 18(2) (2006), 188-201.
5. J.R.Quinlan. Induction of decision trees. Machine Learning, 1: 81-106, 1986.
6. J.R.Quinlan. C4.5: Programs for Machine Learning. Morgan Kaufmann, 1993.
7. L.Breiman, J.Friedman, R.Olshen, and C.Stone. Classification and Regression Trees. Wadsworth International Group, 1984.
8. Марценюк В.П., Кравец Н.О. О программной среде проектирования интеллектуальных баз данных // Клиническая информатика и телемедицина – 2004. – №1. – с.47-53.
9. Марценюк В.П. Математичні моделі в системі підтримки прийняття рішень страхового забезпечення лікування онкологічних захворювань: підхід на основі динаміки Гомперца / В.П. Марценюк, І.Є. Андрущак, І.С. Гвоздецька, Н.Я. Климук // Доповіді Національної академії наук України. –2012. – №10. – С. 34-39.
10. 1 Марценюк В.П. Підхід на основі актуарних математичних моделей до задач страхової медицини / В. П. Марценюк, І.Є. Андрущак, Н.Я Климук // Медична інформатика та інженерія. Науково-практичний журнал. – 2010. – №4. – С. 85-87.
11. Марценюк В.П. О модели онкологического заболевания со временем пребывания на стадии в соответствии с распределением Гомперца / В.П. Марценюк, Н.Я. Климук // Проблемы управления и информатики. Международный научно-технический журнал . – 2012. – № 6. – С. 137-143.
12. Марценюк В.П., Семенець А.В. Медична інформатика. Інструментальні та експертні системи. – Тернопіль: Укрмедкнига, 2004. – 222 с.