Proceedings of Joint Conference of The Korean Data Analysis Society and Korean Data And Information Science Society, October 17-18, 2003, pp. 239-247

CHAID Algorithm by Cube-based Sampling

Hee-Chang Park[1], *Kwang-Hyun Cho*[2]

Abstract

Decision tree algorithms are used extensively for data mining in many domains such as retail target marketing, fraud dection, data reduction and variable screening, etc. CHAID(Chi-square Automatic Interaction Detector), is an exploratory method used to study the relationship between a dependent variable and a series of predictor variables. In this paper we propose and CHAID algorithm by cube-based sampling and explore CHAID algorithm in view of accuracy and speed by the number of variables.

Keywords : data mining, decision trees, CHAID, random samplimg

1. 서론

의사결정나무는 의사결정 규칙(decision rule)을 도표화하여 관심대상이 되는 집단을 몇 개의 소집단으로 분류(classification)하거나 예측(prediction)을 수행하는 데이터 마이닝(data mining) 기법 중의 하나로, 분류와 예측을 목적으로 하는 신경망 분석(neural network analysis), 판별분석(discriminant analysis), 회귀분석(regression analysis)등의 다른 분석들에 비해 연구자가 분석과정을 쉽게 이해하고 설명할 수 있다는 장점이 있다. 이러한 의사결정나무 알고리즘은 시장세분화, 고객세분화 등의 세분화(segmentation), 고객을 신용도에 따라 우량/불량으로 분류하는 분류 문제, 고객속성에 따라 대출한도액을 예측하는 예측, 차원축소 및 변수선택(data reduction and variable screening), 범주의 병합(category merging) 또는 연속형 변수의 이산화(discretizing continuous variable) 등의 분야에서 유용하게 활용되고 있다.

지금까지의 연구를 살펴보면 의사결정나무분석을 수행하기 의한 다양한 분리기준, 정지규칙, 가지치기 방법들이 제안되어 있으며, 이들을 어떻게 결합하느냐에 따라서 서로 다른 의사결정나무 형성방법이 만들어진다. 또한 정확하고 빠르게 의사결정나무를 형성하기 위해서 다양한 알고리즘이 제안되고 있다. 의사결정나무 알고리즘에는 Hartigan(1975)이 제안한 CHAID(Chi-squared

[1]Professor, Department of Statistics, Changwon National University, Changwon, Kyungnam, 641-773, Korea. E-mail : hcpark@sarim.changwon.ac.kr

[2]Graduate Student, Department of Statistics, Changwon National University, Changwon, Kyungnam, 641-773, Korea

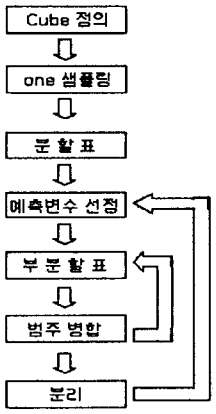
Automatic Interaction detecion), Breiman(1984)이 제안한 CART(Classification and Regression Trees), Quinlan(1993)에 의해 제안된 C4.5, 그리고 Lon와 Shin(1997)이 제안한 QUEST(Quick, Unbiased, Efficient, Statistical Tree) 알고리즘 등이 있다.

이들 중에서 Hartigan에 의하여 처음 소개된 CHAID는 카이제곱-검정(이산형 목표변수) 또는 F-검정(연속형 목표변수)을 이용하여 다지분리(multiway split)를 수행함으로 데이터를 빠르고 효율적으로 탐색하는 의사결정나무의 기본적인 알고리즘이라고 할 수 있다. CHAID는 목표변수가 이산형일 경우에는 카이제곱검정을 이용하는데 독립변수가 목표변수와 유의한 차이가 있는지를 조사하여 목표변수에 가장 유의한 독립변수를 찾아 그 변수를 기준으로 나무를 형성해 나가며 각 마디에서 정지규칙중의 하나가 만족될 때까지 이 과정을 반복한다. 연속형 목표변수에 대한 분리기준으로는 ANOVA의 F값을 사용한다.

거대한 양의 데이터에 대해 CHAID 알고리즘을 이용하여 모형을 구축하기 위해서는 시간과 노력이 많이 소비된다. 본 논문에서는 방대한 데이터 베이스에 대하여 큐브(cube) 기반 표본추출기법을 CHAID 알고리즘에 적용시켜 기존의 CHAID 알고리즘의 나무 모형과 동일하면서 모형구축 시간을 단축시키는 알고리즘을 제시하여 그 결과를 탐색하고자 한다. 2절에서는 큐브 기반 표본에 의한 CHAID 알고리즘을 제시하고, 3절에서는 2절에서 제시한 알고리즘을 바탕으로 예제 및 모의실험을 실시하여 수행결과를 탐색하며, 4절에서 결론을 맺고자 한다.

2. 큐브 기반 샘플링에 의한 CHAID 알고리즘

전형적인 방법의 CHAID 알고리즘은 아주 방대한 양의 데이터를 대상으로, 의사 결정나무를 형성해나가기 까지 매우 많은 계산과정을 거치게 되므로 많은 시간이 요구된다. 이러한 문제점을 보완하기 위해 큐브 기반 샘플링에 의한 CHAID 알고리즘을 제안하고자 한다. 큐브를 기반으로 추출된 표본을 사용한 CHAID 알고리즘은 다음과 같다.



<그림 4> 큐브기반 한점 샘플링에 의한 CHAID 수행단계

각 단계에 대한 설명을 요약하면 다음과 같다.

1) 큐브의 정의

큐브 기반 한점 샘플링을 수행하기 위하여 각 데이터 별로 Cube를 정의한다.

```
<cfquery name="serch" DataSource="#DB#">
    Select #Prediction_var#
    From decision
</cfquery>
<cfloop query="serch">
  <cfset num = #Cube_lavel#*#Prediction_var#>
  <cfquery name="insert" DataSource="#DB#">
     <cfinclude template = "Cube_create.cfm">
  </cfquery>
</cfloop>
```

2) 샘플링

각 예측변수의 범주 별로 하나의 데이터만 랜덤 샘플링한다. 여기서 각 예측변수의 범주에 데이터가 하나도 없으면 샘플링을 하지 않는다. 샘플링 된 데이터로 의사결정 나무 형성에 필요한 DB를 재구성한다. 이 때 불량 데이터는 제거된다. 알고리즘은 다음과 같다.

```
<cfquery name="distinct" DataSource="#DB#">
  Select distinct(lavel) as number
  From #Data_Base#
</cfquery>
<cfloop query="distinct">
  <cfquery name="serch" DataSource="#DB#">
    Select #Prediction_var#
    From #Data_Base#
    Where lavel = #number#
    order by RAND() LIMIT 1
  </cfquery>
  <cfif #Prediction_var# neq null or 0 or #Prediction_num#>
    <cfinclude template = "reject_data.cfm">
  </cfif>
</cfloop>
```

[단계 2] 분할표

새로 구성된 DB에서 각 예측변수에 대해 목표변수의 범주를 수준으로 하는 분할표 분할표를 만든다. 각 예측변수별로 카이제곱 통계량을 구하고 카이제곱 통계량 DB에 저장한다. 이에 대한 알고리즘은 다음과 같다.

```
<cfinclude template = "delete_partition.cfm">
<cfloop index = i from = 1 to = #ArrayLen(prediction_var)#>
  <cfset partition = ArrayNew(3)>
  <cfloop query=#size#>
    <cfloop index = j from = 1 to = #target_size#>
      <cfinclude template = "create_partition.cfm">
    </cfloop>
  </cfloop>
  <cfset chi_value = Σ(n_ij − υ_ij)^2 / υ_ij >
   <cfset df = 2 * (#size.RecordCount#-1)>
  <cfif #chi_value# gt #Evaluate("chi_statistic[#df#]")#>
    <cfinclude template = "insert_Qvalue.cfm">
  </cfif>
</cfloop>
```

[단계 3] 예측변수 선정

카이제곱 통계량 DB에서 통계량이 가장 큰 예측변수부터 범주의 병합을 시행한다. 예측변수 범주의 각 쌍과 목표변수의 범주를 사용한 부분할표를 만들고, 유의한 정도가 가장 약한 쌍의 유의확률이 주어진 임계값보다 크면, 이들 두 쌍을 하나의 범주로 병합하는 과정을 반복한다. 이에 대한 알고리즘은 다음과 같다.

```
<cfquery name = "tree" DataSource="#DB#">
  select node
  from #partition#
  order by s_value desc
</cfquery>
<cfloop query = "tree">
  <cfinclude template = "sub_partition.cfm">
</cfloop>
```

[단계 4] 부분할표

선택된 예측변수에 대하여 예측변수의 각 쌍과 목표변수의 범주를 사용한 부분할표를 만들고 부분할표에서 구해진 통계량이 카이제곱 통계량보다 크면, 이들 두 쌍을 하나의 범주로 병합하는 과정을 반복한다. 알고리즘은 다음과 같다.

```
<cfloop index = "n" from = 1 to = #sub_size#>
  <cfset partition_sub = ArrayNew(2)>
  <cfset category = ArrayNew(1)>
  <cfinclude template = "create_node.cfm">
  <cfinclude template = "sub_partition.cfm">
  <cfloop query=#size#>
    <cfloop index = j from = 1 to = #target_size#>
      <cfinclude template = "create_sub_partition.cfm">
    </cfloop>
  </cfloop>
  <cfset chi_value = Σ(n_ij − υ_ij)^2 / υ_ij>
  <cfset df = 2 * (#size.RecordCount#-1)>
 <cfinclude template = "Annexation.cfm">
</cfloop>
```

[단계 5] 범주 병합 및 분리

각 예측변수의 부분할표에서 얻어진 범주들의 병합여부를 판정한다. 여기서 부분할표에서 새로 얻어진 예측변수를 사용하여 구해진 통계량이 카이제곱 통계량보다 크면 범주를 병합하고 카이제곱 통계량보다 작으면 예측변수를 범주에 따라 분할한다. 예측변수를 분할한 후 아직 분리되지 않은 예측변수에 대하여 위의 과정들을 반복한다. 이에 대한 알고리즘은 다음과 같다.

```
<cfif #chi_value# gt #Evaluate("chi_statistic[#df#]")#>
  <cfset index_2 = #index_1#-1>
  <cfloop index = "in" from = 1 to = #index_2#>
    <cfif #q_value_sub[1][in]# eq #ArrayMax(q_value_sub[1])#>
      <cfset array_list = "#q_value_sub[2][in]#">
    </cfif>
  </cfloop>
  <cfset category = ArrayNew(1)>
  <cfset new_category = ArrayNew(1)>
    <cfloop index = "id" from = 1 to = #listLen(array_list,'/')#>
      <cfset category[id] = #listGetAt(array_list,id,'/')#>
      <cfset new_category[id] = #listGetAt(array_list,id,'/')#>
    </cfloop>
<cfelse>
   <cfbreak>
</cfif>
```

3. 예제 및 모의실험

본 절에서는 2절에서 구현한 알고리즘을 바탕으로 표본추출비율에 따른 수행시간 및 정확도를 탐색하기 위하여 모의실험을 실시하였다. 본 실험의 구현환경은 다음과 같다.

CPU : Intel Pentium4-1.8GHz Northwood
RAM : 256MB
O/S : Linux 7.1
Language : coldfusion 5.0
Database : mysql 3.23.51

이 절에서 사용된 데이터는 모 대학교 신입생 1383명을 대상으로 가정환경에 대하여 조사한 실제 데이터로, 이들 중 본 연구에 이용된 변수는 다음과 같다.

* 목표변수
1. 가정환경
(1) 화목함 (2) 화목하지 않음
* 예측변수
1. 성별
(1) 남자 (2) 여자
2. 아버지학력
(1) 대학이상 (2) 중, 고졸 (3) 초등학교 이하
3. 가정교육방식
(1) 이해심 많음 (2) 보통 (3) 권위적 및 자녀무시 (4) 무관심
4. 생활부담
(1) 부모, 형제, 친척부담 (2) 자신부담(아르바이트) (3) 기타
5. 종교
(1) 기독교 (2) 불교 (3)천주교 (4) 기타 (5) 무교
6. 통학시간
(1) 30분이내 (2) 30분~1시간 미만 (3) 1시간~2시간 이내 (4) 2시간 이상
7. 자신의 외모
(1) 남들 이상 (2) 보통 (3) 남들 이하
8. 자신의 학과 만족
(1) 만족스럽다 (2) 보통 (3) 불만족스럽다

실제 데이터의 수가 표본추출에 의한 CHAID 알고리즘의 효율성 문제를 논의하기에는 부적절하므로, 먼저 1383건의 데이터에 대하여 random으로 500개의 데이터를 20번 반복으로 추출하여 얻어진 100,000건의 데이터 이용하여 모의실험을 하였다. 이 모의실험에서 사용된 100,000건의 데이터에 대한 속성은 다음과 같다.

1. 성별

	화목함	화목하지않음
남	34602	17471
여	34706	13221

2.아버지학력

	화목함	화목하지않음
대학이상	46932	23232
중,고졸	19384	5609
초등학교졸 이하	2992	1851

3. 가정교육방식

	화목함	화목하지않음
이해심 많음	13621	14199
보통	45504	7984
권위적	403	442
무관심	9780	8067

4. 생활부담

	화목함	화목하지않음
부모	60338	24980
자신부담	7397	4614
기타	1573	1098

5. 종교

	화목함	화목하지않음
기독교	17512	17488
불교	16080	15920
천주교	5092	4908
기타	1490	1510
무교	11121	8879

6. 통학시간

	화목함	화목하지않음
30분이내	15121	14879
30분~1시간	20011	19989
1시간~2시간	9702	10298
2시간이상	5133	4867

7. 자신의 외모

	화목함	화목하지않음
남들 이상	15015	14985
보통	35420	34580
남들 이하	15255	14745

8. 자신의 학과만족

	화목함	화목하지않음
만족	20044	19956
보통	19981	20019
불만족	11009	8991

<표3> 예측변수 1, 2, 3, 4를 사용한 알고리즘 수행표(단위 : ms)

구분	전체시간	샘플링제외	표본의 크기	트리
기존 알고리즘	95365	27052	100000	가정교육방식-아버지학력-생활부담-성별
한점샘플링	3320	957	52	가정교육방식

<표4> 예측변수 1, 2, 3, 4, 5를 사용한 알고리즘 수행표(단위 : ms)

구분	전체시간	샘플링제외	표본의 크기	트리
기존 알고리즘	96028	27554	100000	가정교육방식-아버지학력-생활부담-성별
한점샘플링	4325	1590	260	가정교육방식-어버지학력

<표5> 예측변수 1, 2, 3, 4, 5, 6를 사용한 알고리즘 수행표(단위 : ms)

구분	전체시간	샘플링제외	표본의 크기	트리
기존 알고리즘	100438	27146	100000	가정교육방식-아버지학력-생활부담-성별
한점샘플링	20173	5911	992	가정교육방식-아버지학력-생활부담

<표6> 예측변수 1, 2, 3, 4, 5, 6, 7를 사용한 알고리즘 수행표(단위 : ms)

구분	전체시간	샘플링제외	표본의 크기	트리
기존 알고리즘	101224	28080	100000	가정교육방식-아버지학력-생활부담-성별
한점샘플링	26557	11580	3008	기존 알고리즘과 동일 (중간노드 약간 다름:4번째 노드)

<표7> 예측변수 1, 2, 3, 4, 5, 6, 7, 8를 사용한 알고리즘 수행표(단위 : 밀리초)

구분	전체시간	샘플링제외	표본의 크기	트리
기존 알고리즘	101924	28167	100000	가정교육방식-아버지학력-생활부담-성별
한점샘플링	31133	12260	9142	기존 알고리즘과 동일

<표3>에서부터 <표7>의 수행표는 예측변수를 하나씩 추가하면서 실험을 하였다. <표3>에서보는 바와 같이 알고리즘 수행시간에서는 기존의 CHAID 알고리즘의 수행시간보다 많이 단축되었으나 트리의 정확도 면에서 문제가 있다. <표4>에서는 기존의 데이터에서 종교란 예측변수를 추가하여 트리의 정확도를 높였다. <표5>,<표6>에서는 통학시간, 자신의 외모 두 가지 예측변수를 추가하여 모의실험을 한 결과 트리의 정확도 면에서 상당히 높아졌다. <표7>에서 보는 봐와 같이 알고리즘 수행시간에서 기존의 CHAID 알고리즘의 수행시간보다 많이 단축되었고 트리의 구조도 동일한 결과를 얻었다. 200,000 Data에 대해서도 100,000 Data와 똑 같은 결과가 나왔다. 그 이유는 큐브 기반 한점 샘플링 알고리즘이 Data의 개수에 의하여 sample size가 결정되는 것이 아니고 예측변수의 개수와 그 범주의 수에 따라 sample size가 결정되기 때문이다. 이 이유로 예측변수의 개수가 적으면 sample size가 적게 된다. sample size가 적으면 의사결정나무 수행을 제대로 할 수 없다. 따라서 큐브 기반 한점 샘플링 알고리즘은 예측변수의 수가 적은 데이터에는 적합하지 못한 것으로 확인되었다.

4. 결론

본 논문에서는 의사결정나무 알고리즘 중에서 카이제곱 통계량 또는 F 통계량을 이용하여 다지

분리를 수행함으로써 데이터를 빠르고 효율적으로 탐색하는 의사결정나무의 기본적인 알고리즘인 CHAID 기법에 그리드 기반 표본을 이용하는 알고리즘을 제시하고, 이에 대한 정확도와 수행속도를 탐색하였다. 그 결과 그리드 기반 표본 추출에 의한 CHAID 알고리즘이 기존의 알고리즘보다 수행속도 면에서 효과적인 것을 알 수 있었다. 반면에 예측변수의 개수가 적으면 sample size가 적게 된다. sample size가 적으면 의사결정나무 수행을 제대로 할 수 없다. 그래서 큐브 기반 한점 샘플링 알고리즘은 예측변수의 수가 적은 데이터에는 적합하지 못한 것으로 나타났다. 향후에는 이러한 단점을 보완하기 위한 각종 연구가 진행되기를 기대한다.

참고문헌

1. 최종후, 한상태, 강현철, 김은석 (1998). *AnswerTree를 이용한 데이터마이닝 의사결정나무분석*, 서울, 고려정보산업(주).

2. Breiman, L., Friedman, J.H., Olshen, R.A. and Stone, C.J. (1984). *Classification and regression trees,* Wadsworth, Belmont.

3. Catlett, J. Megainduction : *Machine Learning on Very Large Databases.* PhD thesis, University of Sydney, 1991

4. Han, j., Kamber, M.(2001) *Data Mining : Concepts and Techniques.* Morgan Daufmann Publishers.

5. Hartigan, J.A. (1975), *Clustering Algorithms,* New York: John Wiley & Sons, Inc.

6. Ji Hyun You(2002). *Clustering algorithm using a Center Of Gravgity for grid-based sample.*

7. Kass, G.(1980), An exploratory technique for investigating large quantities7of categorical data, *Applied Statistics.* 29, 2, 119-127

8. Kass, G.V.(1980), *An Exploratory Technique for Investigating Large Quantities of Categorical Data*

9. Margaret H.Dunham(2003). *Data Mining Introductory and Advanced Topics.*

10. Chan, P.K. and Stolfo, S.J. (1993). Experiments on multistrategy learning by meta-learning. In *Proc. CIKM,,*314-323.

11. Quinlan, J.R. (1986), Induction of decision trees. *Machine Learning,* 1, 81-106

12. Quinlan, J.R. (1993), *C4.5 Programs for Machine Learning.* San Mateo, Morgan Kaufmann.

13. Wang, W., Yang, I. and Muntz, R. (1997). STING: A statistical information grid approach to spatial data mining. *Very Large Data Bases(VLDB'97).* 186-195