The Journal of The Institute of Internet, Broadcasting and Communication (IIBC)
Vol. 16, No. 4, pp.147-152, Aug. 31, 2016. pISSN 2289-0238, eISSN 2289-0246

http://dx.doi.org/10.7236/JIIBC.2016.16.4.147
JIIBC 2016-4-21

기계학습 모델의 간략화 방법에 대한 연구

A Study on Simplification of Machine Learning Model

이계성*, 김인국**

Gye-Sung Lee*, In-Kook Kim**

요 약 데이터에 내포되어 있는 주요 정보나 지식을 추출해 내는 기계학습 방법에서 주요 이슈의 하나는 지식 표현 방식이다. 여러 가지 구조로 표현될 수 있는 지식을 모델이라고 부른다. 모델에는 그 내부 구조에 따라 트리구조, 네트워크 구조, 리스트 구조, 규칙 등 다양한 구조로 나눈다. 구조의 차이는 단지 표현의 차이뿐만 아니라 그것이 갖는 문제해결 능력에도 차이가 있다. 본 논문에서는 모델을 간략화 시켜 오버피팅 문제를 해결하고 분류 능력을 향상시키는 방법을 제안한다. 모델을 단순화 시키는데 사용되는 파티션 유틸리티 기준함수 제시하고 휴리스틱을 이용하여 균형 잡힌 계층 구조를 생성하는 방법을 제안한다.

Abstract One of major issues in machine learning that extracts and acquires knowledge implicit in data is to find an appropriate way of representing it. Knowledge can be represented by a number of structures such as networks, trees, lists, and rules. The differences among these exist not only in their structures but also in effectiveness of the models for their problem solving capability. In this paper, we propose partition utility as a criterion function for clustering that can lead to simplification of the model and thus avoid overfitting problem. In addition, a heuristic is proposed as a way to construct balanced hierarchical models.

Key Words : Model Simplification, Partition Utility, Overfitting Problem, Pruning

Ⅰ. 서 론

기계학습은 수집된 데이터 또는 관측 값으로부터 유용한 정보를 추출하는 것이다. 결과는 트리, 네트워크 등 다양한 형태로 표현되며 이는 데이터들에 대한 요약으로 볼 수 있고 모델이라 부르기도 한다. 모델을 유도하는 다양한 알고리즘이 개발되어 왔고 다양한 알고리즘처럼 생성되는 모델도 다양하다. 다양한 모델의 생성은 다양한 설명과 해석을 가능하게 하는 이점이 있는 반면 어느 것이 적절한지 선택의 문제가 있게 된다. 모델 생성에는 모델 형태와 구조를 결정하는 편향기준(biased criterion) 함수가 중요한 역할을 한다. 통상 기준함수를 는 그 기준에 적합한 것을 선호하는 방향과 정도를 결정하므로 모델 구축에 핵심이 된다. 생성된 모델은 자세할수록 숨겨진 지식에 대한 이해도나 설명 능력을 제고할 수 있으나 문제해결 관점에서 보면 오버피팅이라는 역효과가 나는 경우가 자주 발생한다. 추출된 지식을 활용하는 목적이라면 생성된 모델은 세세한 지식 요소들을 제거하여 일반화(generalization)하는 과정이 필요하다. 이를 모델의 단순화, 간략화라는 과정이라 부르는데 많은 기계학습 알고리즘은 단순성 원리를 채용하여 일반화를 구현한다.

*정회원, 단국대학교 컴퓨터과학과
**정회원, 단국대학교 컴퓨터학과(교신저자)
접수일자 : 2016년 6월 24일, 수정완료 : 2016년 7월 24일
게재확정일자 : 2016년 8월 5일

Received: 24 June, 2016 / Revised: 24 July, 2016 / Accepted: 5 August, 2016
**Corresponding Author: igkim@dankook.ac.kr
Dept. of Computer Science, Dankook University, Korea

Ⅱ. 단순성 원리

데이터를 통해 가설이나 모델을 학습하는 학습 방법에는 오컴의 면도날이라고 불리는 원리가 폭넓게 적용되어 오고 있다. 이 원리의 핵심은 '엔티티는 필요이상으로 확장되어서는 안 된다'는 뜻으로 해석된다[1]. 다른 해석으로 '다수 개의 원리, 정의, 해석은 필요 이상으로 사용되어서는 안 된다'가 있다. 기존의 원리가 충분한 설명을 제공한다면 새로운 원리를 만들지 말아야 한다는 것이다. 단, 충분한 설명을 제시하지 못하여 세부적인 설명이 불가피하다면 추가적인 원리가 필요하다고 이해 될 수 있을 것이다. 이 원리는 어떤 현상을 이해하는데 있어 필요 이상의 가정은 불필요하므로 가설이나 이론에 대한 관측 가능한 예측을 하는데 있어 불필요한 가정은 제거되어야 한다는 것이기도 하다. 이 원리가 기계학습 알고리즘의 핵심 규칙으로 널리 사용되어 왔다.

단순성을 활용한 대표적 알고리즘인 ID3[2] 계열의 알고리즘은 속성별 분류 능력을 조사하여 단순한 확장이 그렇지 않은 확장보다 선호하게 되는 편향된 기준에 의해 동작되는데 이를 계량화한 것이 엔트로피이다. 속성별 엔트로피를 구해 가장 작은 엔트로피를 선택하여 재귀적으로 의사결정 트리를 확장해 나가는데 확장된 노드 중 노드를 구성하는 구성 사례들이 같은 부류로 모여질 때 그 노드는 확장이 중단된다. 그림 1은 갑상선(thyroid) 질환을 진단하는데 사용되는 의사결정 트리이다. 매 단계에서 속성이 결정되는데 그 결정에서 사용되는 기준함수인 정보 엔트로피는 오컴의 면도날의 원리를 그대로 적용되어 사용된다.

엔트로피 H는 식(1)로 구해진다.

$$H(X) = E(I(X)) \tag{1}$$

여기서 X는 이산적 랜덤 변수이고 $x_1 \cdots x_n$ 값을 갖는다. E는 기댓값 함수이고 $I(X)$는 정보량을 나타내는 함수이다. 이산 확률사건 X에 대한 엔트로피 값은 식(2)와 같이 표현된다.

$$H(X) = \sum_{i=1}^{n} p(x_i)\log(\frac{1}{p(x_i)}) = -\sum_{i=1}^{n} p(x_i)\log p(x_i) \tag{2}$$

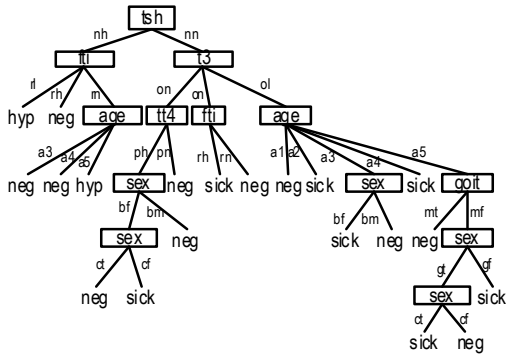

그림 1. 갑상선 질병의 의사결정 트리
Fig. 1. Decision tree of Thyroid disease

두 번째로 카테고리 유틸리티를 사용하여 계층적 구조를 생성하는 무감독 알고리즘인 COBWEB[3]을 살펴보자. 카테고리 유틸리티(CU)는 주어진 데이터를 클러스터링하여 분류하는데 사용되는 기준함수이다. ID3과 근본적으로 다른 점은 COBWEB에서는 사례의 클래스 정보가 없다는 점이다. 클래스 정보에 의한 학습은 클래스별로 값의 분포 값을 알 수 있는 확률분포함수를 가질 수 있으나 클래스 정보가 없는 사례 집합에서는 값들의 확률 분포 자체에 의해 분류가 이뤄진다.

카테고리 유틸리티의 기댓값은 $P(A_i = V_{ij}|C_k)^2$ 로 정의되고 이것은 주어진 자료가 클래스 C_k 에 속한다고 가정할 때 A_i가 V_{ij} 값을 갖게 될 확률을 나타내고 그 의미하는 바는 자료의 해당 클래스와의 확률적 결합도를 나타낸다. 클래스 C_k 에 대해 CU는 다음과 같이 정의된다[3].

$$CU_k = P(C_k)\sum_i\sum_j P(A_i = V_{ij}|C_k)^2 - P(A_i = V_{ij})^2 \tag{3}$$

이것은 속성값 쌍이 클래스 k에 대해 올바르게 추측될 수 있는 속성 값들의 기댓값(첫 번째 항)과 클래스에 대한 정보가 없을 때의 기댓값(두 번째 항)과의 차이에 해당하는 증분을 의미한다. 이 증분이 커질수록 카테고리 유틸리티 값은 커지게 되고 같은 위치에 있는 다른 클러스터와 구분이 좀 더 명확하게 된다는 의미를 내포하게 된다. 카테고리 유틸리티의 역할은 클러스터 내부의 유사성과 클러스터 간 차이점을 극대화하는 타협점을 찾는 것이라고 볼 수 있다. 평가함수를 바탕으로 분류 트리를 구축하는 과정에서 클래스에 해당되는 클러스터는 동적으로 생성될 수 있다. 위의 식에서 C_k는 초기에는 1개

의 클래스로 시작하여 새로운 사례 개체가 추가될 때마다 C_k에 속할 때와 새로운 C_l에 배당이 될지 결정되는데 이는 CU_k에 의해 결정된다.

생성된 트리 모델의 최 하단에 위치한 노드는 같은 클래스 사례들로 구성되어 있다. 만일 학습을 위한 입력으로 사용되는 사례 집단이 대규모일 때는 그 결과로 생성되는 트리 구조가 대형일 것이므로 생성자체도 매우 어려운 과정이 될 뿐만 아니라 문제해결 관점에서 단말 노드의 유효성이 매우 낮을 것이다.

이를 개선하기 위한 방법으로 절단(pruning)이라는 것이 있다. 이는 생성된 트리 구조에서 불필요하거나 잉여적인 것을 제거하는데 사용된다. 특정 기준에 의해 사전 또는 사후에 절단하면 모델이 단순화되고 문제해결 능력도 개선된 모델로 바뀌게 된다. 기계학습 알고리즘이 점진적으로 모델을 구성해 나갈 때 한쪽으로 편향되는 모습으로 모델이 구성될 수 있다. 예로 트리 구조는 한번 자식 노드로 확장되면 그 이하로만 확장이 되기 때문에 사향(skewed) 트리구조로 귀결될 수 있다. 이런 단점을 해결하기 위한 시도로 승진, 병합과 같은 동적 재구성 방법을 사용하거나 특정 시점에서 전체적인 구조를 재구성하는 방법이 연구되기도 한다[4].

Ⅲ. 모델의 간략화

트리구조를 생성해내는 알고리즘은 항상 오버피팅 문제를 안고 있다. 트리구조의 하부 노드에서 형성되는 클러스터 들은 너무 상세한 분류이거나 불필요하게 구체적인 정보를 갖게 된다. 이들을 판별하여 제거하는 방안으로 사전, 사후 절단 방법이 있다. 절단은 설정된 평가 함수에 의해 결정된다. 사후 절단인 리샘플링을 이용하여 모델을 간략화 시키는 방법을 고려해 보자. 이 방법은 2개의 과정으로 나눠서 진행된다. 먼저 트리를 하부구조까지 완전하게 생성한다. 그 후 리샘플링을 이용하여 노드의 유효성을 판단해 프론티어 라인을 찾아낸다. 프론티어 라인은 사례 전체를 배타적으로 커버하는 노드들의 모임으로 구성되는 라인을 일컫는다. 그림 2는 3개의 레벨로 이뤄진 트리 구조에서 리샘플링 방법을 사용하여 속성 별로 유효 노드를 찾는 그림을 보여주고 있다. 점선은 프론티어 라인을 나타내고 이는 우도를 통해 유효노드를 찾아 결정된다[10].

유효노드를 찾기 위해 테스트 세트에 속한 사례를 하나씩 선택한 후 각 속성별 값을 가지고 우도를 찾는다. 각 사례는 트리를 루트로부터 시작하여 확률적으로 가장 유사한 클러스터에 속하도록 분류해 나간다. 이 과정 중 각 속성 A_i에 대한 값과 해당 노드의 A_i 값을 비교하여 확률적으로 가장 큰 값으로 일치한다면 그 노드에 대한 카운트를 1씩 증가시킨다. 이런 방식으로 카운트를 증가시켜가면서 트리를 방문한 후 카운트 값이 가장 큰 값을 갖는 노드들을 찾아 프론티어 라인을 형성해 나간다. 프론티어 라인 아래에 있는 노드는 테스트세트의 사례들에 대하여 각 속성별 값의 비교할 경우 올바른 예측에 필요한 확률적 근거를 충분히 제공해 주지 못하는 것으로 판단되어 절단 대상이 된다.

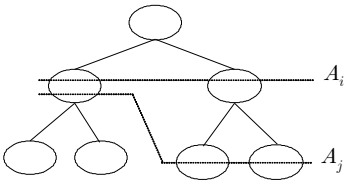

그림 2. 리샘플링에 의한 유효 노드 선택
Fig. 2. Valid node selection by resampling

위의 방법과 달리 사전 절단 방법으로 기준 함수를 활용하는 방법이 있다. COBWEB[3]에서 사용한 카테고리 유틸리티를 종합한 파티션 유틸리티(PU)[5][9]를 정의한다.

$$PU = \frac{1}{K}\sum_{j=1}^{K} p(c_j)\,CU_j \tag{4}$$

이 수치는 현재의 노드가 K개의 노드로 확장될 때 현재 노드의 파티션 유틸리티가 된다. 하위 노드의 카테고리 유틸리티의 기대값을 K로 나눠 산출한 값이 된다. 여기서 K는 하위 레벨의 노드 수이다. K로 나누는 이유는 파티션을 이루는 클러스터의 수가 작을수록, 다시 말해, 모델의 규모가 작아지도록 유도하는 경향이 있다고 말할 수 있어 좀 더 작고 간략화된 모델을 형성해 나가는데 도움을 준다. 트리 구조 모델에서 이 값은 하위로 내려갈수록 증가하는 경향이 있다. 그 이유는 하위 노드로 갈수록 CU 값이 상승하기 때문이다. 그러나 어느 레벨 이후에는 값이 줄어들 가능성이 높다. 이런 현상이 일어나는 근본 이유는 어느 시점에서는 하위 클러스터로 구분하는 것이 결코 클러스터링에 도움이 되지 못하기 때문이다. 초기

에 상승한 PU 값이 어느 시점에서는 정지하거나 하락할 경우 이는 모델의 오버피팅을 암시하는 시점이라고 볼 수 있고 이 지점이 절단의 적절한 지점이 된다고 판단할 수 있다.

Ⅳ. 실험 및 결과

데이터 세트는 풍선[6]으로 구성된 데이터베이스이다. 원래는 4개의 데이터세트로 구성되어 있는데 그중 하나인 small-yellow 데이터 세트가 실험을 위해 사용된다. 원래의 데이터세트에는 16개의 사례가 포함되어 있고 각각은 4개의 속성을 갖는다(color, size, act, age). 그리고 각 사례는 클래스 레이블 정보를 갖고 있다 (inflated). 이 데이터 세트로부터 얻어진 결론은 만일 color가 yellow이고 size가 small이면 그 풍선은 inflated (바람이 찬 상태)가 된다. 그렇지 않은 경우는 deflated(바람이 빠진 상태)가 된다. 12개의 데이터(d_1~d_{12})를 훈련세트로 선발되고 클래스 레이블을 속성으로 편입시켜 클러스터링을 시행하였다. 12 데이터 사례 중 5개(d_1~d_5)는 inflated 이고 나머지(d_6~d_{12})는 deflated이다.

1. PU에 의한 모델 구축

이 데이터세트에 COBWEB 알고리즘을 적용하여 그림 3 a)와 같은 트리구조가 생성되었다. 각 노드의 좌측 상단에 노드 번호가 부여되어 있다. 박스 내에는 사례 번호와 파티션 유틸리티 점수가 표기되어 있다. 총 12개의 사례에 d_1에서 d_{12}까지 식별자를 부여하였다. 우선 첫 번째 레벨에서는 3개의 그룹으로 분류되었음을 확인할 수 있다. d_1에서 d_5까지는 같은 그룹인 inflated에 속한 것이기 때문에 첫 번째 레벨에서 중간 노드에 이들이 몰려 있음을 볼 수 있다. d_1에서 d_5에 이르는 데이터로 모여진 레벨 1의 중간 노드의 파티션 유틸리티 값이 0.590으로 그 좌우에 있는 노드의 파티션 유틸리티 값보다 상대적으로 높음을 볼 수 있는데 이는 deflated에 속한 7개의 데이터가 두 개의 그룹으로 분리된 것으로 원인을 돌릴 수 있을 것이다. 이 둘을 합친 것 보다는 파티션 유틸리티가 낫기 때문에 이들을 분리한 결과로 귀결된 것이다. 이는 모델의 단순성 원리가 적용된 단면을 보여 주고 있다고 할 수 있다.

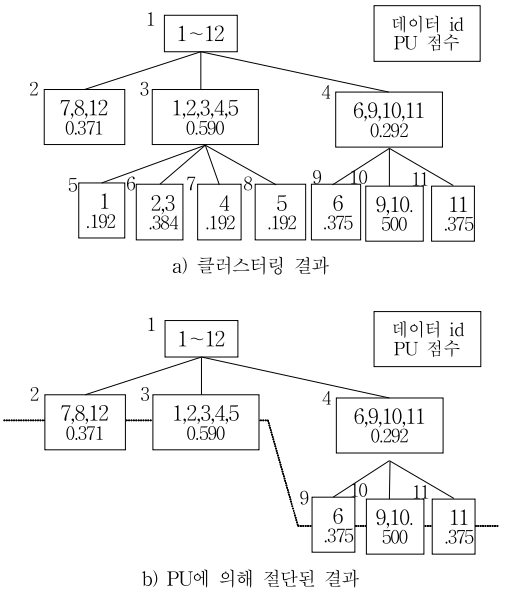

그림 3. 풍선데이터에 대한 클러스터링
Fig. 3. Clustering for balloon data

레벨 1의 각 노드는 하위 노드로 확장이 지속되는데 2, 6, 10번 노드는 1개의 사례로 구성된 노드로 확장된다. 1개의 사례로 구성된 노드들의 PU 값은 급격히 줄어든다. PU 값을 기준으로 볼 때 절단된 후의 결과 트리 구조는 그림 3 b)와 같게 된다. 절단된 노드는 그리지 않은 노드를 포함하여 총 11개가 되어 전체 노드의 61%인 절단율을 보여준다.

4번 노드를 제외하면 나머지 2개의 노드는 분리에 의해 유틸리티 값이 개선되지 못함을 알 수 있다. 레벨 1에서 좌측 2개의 노드의 파티션 유틸리티 값을 살펴보면 0.371과 0.590으로 되어 있다. 이 값은 그 아래 레벨로 확장될 때 하위 노드의 파티션 유틸리티 값은 모두 그 부모 노드의 값에 비해 낮음을 확인할 수 있다. 이 수치의 의미는 파티션으로 분리하는 것 보다는 그들이 모여서 구성된 노드가 보다 더 유용한 역할을 할 수 있을 것으로 보는 것이다. 반대로 4번 노드는 두 번째 레벨로 확장할 때 그 파티션 유틸리티 값이 상승하는 것을 볼 수 있어 이 노드는 확장 대상이 된다. 확장된 노드들에서 프론티어 라인(점선)을 구성하게 된다.

2. 휴리스틱에 의한 모델의 간략화

리샘플링에 의한 사후 절단을 하든가 파티션 유틸리티와 같은 기준함수를 통해서 사전 절단이든지 간략화된

모델을 구하기 위해 가능한 모델을 모두 생산한 후 이중에서 가장 좋은 모델을 선택하는 모델선택 방법도 연구되고 있다[7][9][10]. 또는 여러 개의 모델을 결합하여 활용하는 방법도 제시되기도 하였다[8]. 이와 같은 접근 방법은 많은 계산 비용 때문에 제한적일 수밖에 없다. 보다 효과적인 알고리즘으로 데이터 특성을 미리 파악하여 보다 안정적인 모델 구조를 생성해 내는 방법을 제안한다. 풍선 예제를 통해 살펴보면 우선 데이터를 전처리하여 데이터 구조를 파악한다. 그 구조란 데이터 속에 존재하고 있는 확률적 분포를 미리 파악하는 것으로 일단 이 분포를 대표하는 데이터 일부로 프로토타입 모델을 신속하게 구축한다. 이 모델을 기준으로 데이터를 배분하여 데이터 처리 순서를 정한 후 모델을 구성해 나가면 좀 더 균형 잡힌 구조를 유도해 낼 수 있다(그림 4). 초기 그룹을 안정적으로 형성할 수 있는 seed를 찾아 이를 통해 트리를 구성하면 균형 잡힌 트리 구조를 생성해 낼 수 있게 된다.

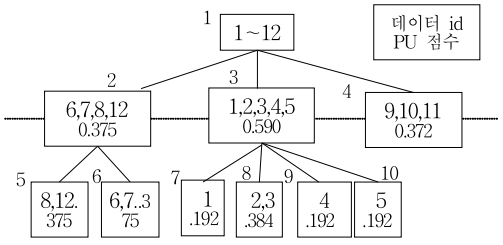

그림 4. 휴리스틱에 의한 트리 구조
Fig. 4. Tree structure by heuristic

그림 4의 결과를 보면 레벨 1이 프론티어 라인을 형성한다. 레벨 1의 3개의 노드 이후에 확장되는 모든 하위 노드는 각각의 부모노드의 파티션 유틸리티 값에 비해 모두 작은 것을 알 수 있다. 휴리스틱을 추가하여 생성된 트리구조는 앞서서 구해진 트리구조와 약간의 차이를 갖게 된다. 그 이유는 데이터를 사전 분석하여 분포 정보를 확인한 후 그 분포에 의거하여 데이터를 처리한 결과로 인한 것이다. 첫 번째 레벨에서는 3개의 노드로 구성되어 있으나 각 노드의 구성은 중간 노드만 같고 나머지 두 개의 노드에서는 다른 구성을 갖는다. 두 번째 레벨에서는 노드의 수에서나 노드의 구성 내용에 있어 서로 다른 구성을 갖는다. 여기서 중요한 차이를 보여주고 있는데 파티션 유틸리티에 의해 구성된 프론티어 라인이 앞에서는 5개의 클러스터를 갖는 분류가 형성되었으나 여기서는 오로지 3개로 구성된 클러스터링이 형성되었다. 휴리스틱과 파티션 유틸리티를 이용하여 보다 간략하고 균형 잡힌 모델이 구축된 것을 보여준 것이다. 트리 전체의 파티션 유틸리티 값은 최상위 노드인 루트에서 구해 그 값을 비교할 수 있다. 파티션 유틸리티로 사후 절단을 한 트리의 전체 파티션 유틸리티는 0.418인 반면 휴리스틱을 적용한 트리구조의 전체 파티션 유틸리티는 0.445로 좀 더 나은 모델임을 수치적으로 증명하였다. 두 트리 모델에 대한 비교는 표 1에 요약되었다.

표 1. 두 트리 모델의 비교
Table 1. Comparisons of tree models

구분	PU	휴리스틱
트리 구조	그림 3 b	그림 4
총 클러스터 수	5	3
트리 규모 (노드 수)	18	19
절단 후 트리 규모 (노드 수)	7	4
절단율	61%	79%
절단 후 트리 레벨	2	1
전체 파티션 유틸리티	0.418	0.445

Ⅴ. 결 론

본 논문은 기계학습 분야에서 적용되는 기본 원리인 단순화 원리의 의미와 활용, 구현 방법에 대해 조사하고 분석하였다. 모델을 단순화하는 전략으로 파티션 유틸리티 함수를 이용하여 사전절단 과정을 거치면서 노드의 확장을 제한함으로 트리 모델의 무분별한 확장을 막았다. 이와 더불어 데이터 속에 포함되어 있는 확률분포를 활용하여 초기 모델이 균형 잡히게 구성될 수 있도록 유도하는 방안을 제안하여 균형 잡힌 최종 모델이 완성되도록 시도되었다. 파티션 유틸리티를 통한 모델의 단순화는 기계학습의 문제인 오퍼피팅 문제를 해결하는 하나의 방안으로 활용될 수 있음을 보였다. 개선 효과를 판별하기 위해 파티션 유틸리티 점수로 모델의 개선 효과를 보였다.

References

[1] Wikipedia, https://en.wikipedia.org/wiki/
[2] Quinlan, J.R. “Induction of Decision Trees,” Machine Learning, 1, pp.81-106 1986.

[3] Fisher, D., "Iterative Optimization and Simplification of Hierarchical Clustering,", pp.147-179, J. of AI Research, 1996
[4] Jeon, Jinho, Lee, Gyesung., Wu, X, "Rearranging datga objects for efficient and stable clustering," Applied Computing(ACM), March, 2005
[5] Biswas, G, Weinberg, J.B. and Fisher, H.D., "ITERATE: A conceptual clustering algorithm for data mining" IEEE Tr. on systems, man and cybernetics, vol.28, part C No.2, May 1998.
[6] UCI machine learning repository, https://archive.ics.uci.edu/ml/datasets/
[7] Myung, Jay., Navarro, D.J., and Pitt, M.A., "Model Selection by Normalized Maximum Likelihood," Journal of Mathematical Psychology, pp. 167-179, Elsevier, 2006.
[8] Zhang, X.S., Shrestha, B., Yoon, S, et.al., "An Ensemble Architecture for Learning Complex Problem-solving Techniques from Demonstration," ACM Trans. on Intelligent Systems and Technology, pp.3-38, 2012
[9] Cho Younghee, Lee Gyesung, A study on improving prediction accuracy by modelling multiple similar time series, pp. 137-143, JIIBC, 2010.
[10] Cho Younghee, Lee Gyesung, Prediction on Clusters by using Information Criterion and multiple seeds, pp. 145-152. , JIIBC, 2010.

저자 소개

이 계 성(정회원)

- 1982년 : 한국과학기술원 전산학과 이학석사.
- 1994년 : Vanderbilt 대학 컴퓨터과학과 공학박사.
- 1994년 ~ 현재 : 단국대학교 컴퓨터과학과 교수

<주관심분야 : 데이터마이닝, 지능형시스템, 생명정보학>

김 인 국(정회원)

- 1982년 : 단국대학교 수학교육학 학사
- 1985년 : 에모리대학교 컴퓨터과학과 석사
- 1995년 : 아주대학교 컴퓨터공학과 공학박사
- 1986년 ~ 현재 : 단국대학교 컴퓨터과학과 교수

<주관심분야 : 실시간 운영체제, 빅데이터분석>