Journal of the Korea Convergence Society
Vol. 12. No. 8, pp. 39-45, 2021

ISSN 2233-4890 / eISSN 2713-6353
https://doi.org/10.15207/JKCS.2021.12.8.039

결정트리를 이용하는 불완전한 데이터 처리기법

이종찬
청운대학교 컴퓨터공학과 교수

Incomplete data handling technique using decision trees

Jong Chan Lee
Professor, Dept. of Computer Engineering, Chungwoon University

요 약 본 논문은 손실값을 포함하는 불완전한 데이터를 처리하는 방법에 대해 논한다. 손실값을 최적으로 처리한다는 것은 학습 데이터가 가지고 있는 정보들에서 본래값과 가장 근사한 추정치를 구하고, 이 값으로 손실값을 대치하는 것이다. 이것을 실현하기 위한 방안으로 분류기가 정보를 분류하는 과정에서 완성되어가는 결정트리를 이용한다. 다시말해 이 결정트리는 전체 학습 데이터 중에서 손실값을 포함하지 않는 완전한 정보만을 C4.5 분류기에 입력하여 학습하는 과정에서 얻어진다. 이 결정트리의 노드들은 분류 변수의 정보를 가지는데, 루트에 가까운 상위 노드일수록 많은 정보를 포함하게 되고 말단 노드에서는 루트로부터의 경로를 통해 분류 영역을 형성하게 된다. 또한 각 영역에는 분류된 데이터 사건들의 평균이 기록된다. 손실값을 포함하는 사건들은 이러한 결정트리에 입력되어 각 노드의 정보에 따라 순회과정을 통해 사건과 가장 근접한 영역을 찾아가게 된다. 이 영역에 기록된 평균값을 손실값의 추정치로 간주하고, 보상 과정은 완성된다.

주제어 : 손실값, FLDF, 불완전한 데이터, 보상 평균, 결정트리, 분류기

Abstract This paper discusses how to handle incomplete data including missing values. Optimally processing the missing value means obtaining an estimate that is the closest to the original value from the information contained in the training data, and replacing the missing value with this value. The way to achieve this is to use a decision tree that is completed in the process of classifying information by the classifier. In other words, this decision tree is obtained in the process of learning by inputting only complete information that does not include loss values among all training data into the C4.5 classifier. The nodes of this decision tree have classification variable information, and the higher node closer to the root contains more information, and the leaf node forms a classification region through a path from the root. In addition, the average of classified data events is recorded in each region. Events including the missing value are input to this decision tree, and the region closest to the event is searched through a traversal process according to the information of each node. The average value recorded in this area is regarded as an estimate of the missing value, and the compensation process is completed.

Key Words : Missing value, FLDF, Incomplete data, Imputation mean, Decision tree, Classifier.

*This paper is sponsored by the academic research project of Chungwoon University in 2021. (No. 2021-49).
*Corresponding Author : Jong Chan Lee(jclee@chungwoon.ac.kr)
Received May 16, 2021 Revised June 21, 2021
Accepted August 20, 2021 Published August 28, 2021

1. 서론

원격지로부터 수집되거나 서로 상이한 운영체제 환경에서 모아지는 데이터들은 손실값을 가지는 불완전한 데이터인 경우가 많다[1,2]. 이러한 데이터를 처리하는 방안으로 1) 손실값을 포함하는 사건들을 무시하는 방법, 2) 손실변수의 평균이나 확률 등의 예측값을 구해 손실값에 할당하는 방법. 3) 전체 데이터로부터 손실값을 추정할 수 있는 유용한 정보를 추출하여 손실값의 추정값을 구하는 방법[3-7] 등이 있다. 이들 중 1)은 수집된 상당수의 정보를 포기하는 것으로 정보의 손실을 피할 수 없다. 2)는 학습 데이터를 이루는 모집단에 대한 사전 지식을 가진 경우에 적합하며 모수(parametric) 통계의 기법들을 이용하는 경우 좋은 결과를 얻는 것으로 알려졌다. 3)은 딥러닝 기법을 사용하는 대부분의 경우로, 모집단의 분포가 불규칙하여 정규분포 등의 분포는 물론 표준편차나 평균 등의 사전 정보가 부족한 경우에 적용될 수 있으며 비모수(nonparametric) 통계의 기법들이 사용될 수 있다. 본 논문의 손실값을 처리하는 관점은 3)의 경우에 대한 해법을 위한 것이다.

본 논문에서 손실값을 최적으로 보상할 수 있는 값을 구하는 것이 목적이며, 이 값을 분류 알고리즘에서 분류된 영역에서 찾아낼 수 있을 것이라는 아이디어를 기본으로 출발하였다. 이를 구현하기 위해 불완전한 데이터로부터 완전한 정보만을 골라 결정트리(decision tree)를 구성하고 이 결정트리에 의해 구분된 영역의 정보로 손실된 값의 추정치를 구해간다. 결정트리가 구분한 영역에는 다차원의 공간에서 서로 거리가 유사한 이웃값들끼리 모아지게 되기 때문에 손실값에 가장 근접한 영역을 찾는다면 이 영역에서 최적의 보상 정보를 구할 수 있을 것이다. 이를 구현의 면에서 보면, 학습 데이터를 손실된 사건과 손실되지 않은 사건으로 양분 한 다음, 손실되지 않은 데이터를 가지고 C4.5 분류 알고리즘[8]을 이용해 분류하며 결정트리를 구성하면서 각각의 정보 영역이 구분되어 간다. 그리고 손실 데이터들을 차례로 이 분류 영역에 순회하도록 하여 손실값에 가장 근접한 영역을 찾아가도록 한다. 이 영역에는 구분된 사건들의 평균값을 이용해 손실 변수에 해당되는 값을 손실값으로 대치하여 보상이 이루어지도록 한다.

손실값의 추정치와 실제값의 차이를 알아보는 성능 평가를 위해 Fisher의 식을 사용하는 SVM 알고리즘[9]으로 학습한 후 그 성능을 비교해 본다. 성능 평가를 위해 SVM을 사용하는 것은 첫째 C4.5 분류기는 학습 데이터가 정수인 것이 일반적이고 실수 속성을 가지는 경우 별도의 처리 방법이 필요하다. 그러나 FLDF의 분류기는 학습 데이터의 속성값이 정수든 실수든 상관이 없다. 둘째 딥러닝과 같이 반복적으로 학습하는 알고리즘을 이용한다면 좀 더 좋은 결과를 산출할 수 있겠지만, 이들 알고리즘은 기본적으로 손실된 정보를 보상하는 기능이 있어 손상된 정보의 성능을 비교하는 실험에는 적절하지 않기 때문이다. 각 변수들에 대해 손실의 정도를 달리한 성능 평가 결과에서 정보의 손실을 최소화하는 보상 방법이 의미가 있음을 확인한다.

2. 관련 연구

2.1 C4.5 분류기를 이용한 결정트리의 생성

C4.5는 엔트로피 함수를 기반으로 귀납적 추론(Inductive Inference) 방식을 사용하는 분류기로 학습 데이터의 속성값들에 대해 학습 과정에서 하향식 분할과 정복(top-down devide-and-conquer)으로 결정 트리를 구성해 나간다[9]. 이 결정 트리는 학습 데이터를 최적으로 분류할 수 있는 속성(attribute, A)을 반복적으로 선택하여 트리를 점진적으로 완성해 나간다. 속성의 선택 기준은 얼마나 비슷한 항목들끼리 서로 모이도록 분류하는지 이며, 이를 측정하는 방법으로 지니계수, 카이제곱 통계량, 엔트로피 등이 있는데 C4.5는 엔트로피 함수를 사용한다.

C4.5 알고리즘에서는 각 속성들에 대해 (1) 식과 같은 Gain_ratio 함수를 계산한다. 이 함수는 각각의 속성값에 따라 분류할 때 같은 부류를 가지는 사건들이 얼마나 잘 모아지는지를 측정하는 값으로, 각 속성을 선택했을 때 얻을 수 있는 정보획득량을 예측할 수 있게 한다. 따라서 (1) 식을 각 속성들에 적용하여 가장 큰 정보 이득을 가진 속성을 선택하고, 이 속성값에 따라 같은 값을 갖는 사건들이 하부 노드로 전달되며 분기된다. 이와 같은 과정을 같은 속성들을 가지는 사건들이 임의의 영역으로 모두 나누어질 때까지 실행하면 학습 데이터에 따르는 결정 트리가 완성되어 진다.

$$Gain\_ratio(A) = Gain(A) / Split\_info(A) \quad (1)$$

- S : 각 노드에서의 데이터 집합
- $|S|$: S의 데이터 개수
- S_{A_j} : S 집합 중에서 속성 A가 j 값을 가지는 S의 부분 집합

- $|S_{A_j}|$: S_{A_j}의 데이터 개수
- $freq(C_i, S)$: S에서 부류가 C_i인 사건의 개수, i=1,...,k

(1) 식에서 Gain(A)는 정보획득량을 나타내는 함수로 (2) 식과 같이 나타날 수 있다.

$$Gain(A) = info(S) - info_A(S) \quad (2)$$

여기서 $info(S)$는 예측 속성에 대한 엔트로피로 계산되어 분할 전 엔트로피라 하며, $info_A(S)$는 예측 속성에 대한 독립 변수의 엔트로피로서 분할 후 엔트로피라 한다. 따라서 $info(S) - info_A(S)$ 값이 가장 큰 속성이 정보가 가장 많다고 판단하여, 각각의 속성들 중 가장 큰 값을 가진 속성값을 기준으로 분할을 진행한다. 이러한 과정은 Gain(A) 값이 0에 근접할 때까지 반복된다.

$$info(S) = -\sum_{i=1}^{k}(freq(C_i,S)/|S|) \cdot \log_2(freq(C_i,S)/|S|)$$

$$info_A(S) = \sum_{j=1}^{n}(|S_{A_j}|/|S|) \cdot info(S_{A_j})$$

$$info(S_{A_j}) = -\sum_{i=1}^{k}(freq(C_i,S_{A_j})/|S_{A_j}|) \cdot \log_2(freq(C_i,S_{A_j})/|S_{A_j}|)$$

(1)의 Split_info(A)는 결정 트리가 균등하게 배분되도록 하는 역할을 담당하며 (3) 식에 나타나 있다.

$$Split\_info(A) = -\sum_{j=1}^{n}(|S_{A_j}|/|S|) \cdot \log_2(|S_{A_j}|/|S|) \quad (3)$$

2.2 결정트리에서 구한 확률값으로 손실값을 대치하는 알고리즘

손실값을 처리하는 가장 간단한 방법은 본래의 학습 데이터에서 손실 사건이 가지고 있는 정보를 모두 무시하고 손실값에 균등한 확률값을 할당하는 것이다[10]. 이 방법은 엔트로피가 균등한 값에 가장 크기 때문에 엔트로피를 척도로 하는 알고리즘에는 적합하다[11]. 그러나 딥러닝과 같이 엔트로피와 연관 없는 학습에서는 적합하지 않아[12] 학습 데이터 자체가 가지고 있는 정보로 확률을 구함으로써 손실값을 보상하는 알고리즘이 고안되었다[13,14]. 이 알고리즘에서는 각 속성값의 확률을 표현하기 쉽게 데이터 구조를 변경한 데이터 확장 기법을 사용하였는데 처리 과정은 다음과 같다.

1. 학습 데이터 집합을 손실 데이터와 비손실 데이터로 분리한다.
2. 비손실 데이터를 C4.5에 입력하여 학습 과정을 통해 분류하고, 이때 각 부류당 영역을 의미하는 결정트리가 구성되어 진다. 다시 말해 손실이 없는 완전한 사건들은 C4.5 분류기로 보내 점진적으로 결정트리를 완성하고, 데이터 확장 기법을 위해 원-핫-인코딩으로 변환한다.
3. 손실 데이터의 각 사건들을 차례로 2번의 결정트리에 입력하여 가장 근접한 영역을 찾고 이 영역의 정보를 확률로 나타내어 손실값을 대체한다. 다시 말해 손실 사건들에서 손실되지 않은 변수를 결정트리에 입력해 순회하며 확률값을 구하고 손실 변수에 채워 데이터 확장 기법의 변환을 완성한다.

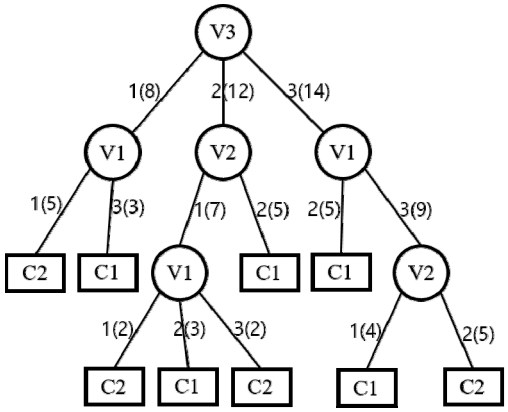


Fig. 1. Example of decision tree obtained through C4.5

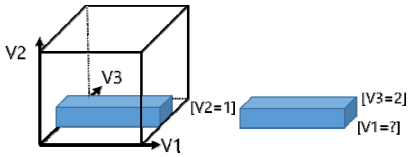
Fig. 1에서 V1은 3, V2는 2, 그리고 V3는 3개의 카디너리티를 가지는 임의의 학습 데이터를 이용하여 학습한 결과 얻어진 결정트리라고 가정한다. 여기서 동그라미는 선택된 변수를 의미하는 노드이며, 노드와 노드 사이의 수들(ex: 2(5))은 변수값(분류된 영역에서의 사건수)을 의미한다. 또한 네모는 말단(leaf) 노드이며 각 부류의 영역을 의미한다. 이 결정트리에 손실값을 포함하는 사건들을 입력한다. 손실값을 포함하는 사건의 예로 {-, 1, 2} 사건의 경우 VL 표현식으로 나타내면 [V1=?][V2=1] [V3=2]가 된다. 이 사건을 Fig. 1의 결정트리에 입력하여 트리를 순회하면 [V3=2]이고 [V2=1]인데 V1 값이 손실되었다. 그러므로 말단 노드의 각 부류 영역으로 도달하지 못하기 때문에 확률로 손실값을 나타내야 한다. 다시 말해 V1 노드와 말단 노드 사이의 정보를 이용해 P(V1==1)=2/7, 즉 V1이 1일 확률은 2/7이다. 그리고 P(V1==2)=3/7, P(V1==3)=2/7로 나타내진다. 즉, P(V1) = {2/7, 3/7, 2,7}의 확률값으로 표현된다. 다른 사건의 예들을 보면 {3, -, 3}의 경우 P(V2) = {4/9, 5/9}이고, {-, 2, 1} 사건은 P(V1) = {5/8, 0, 3/8}

가 된다. 마지막으로 {3, 2, -} 사건의 경우 P(V3) = {8/34, 12/34, 14/34}의 확률을 구할 수 있다. 그러나 이 예의 경우 순회 중 첫 번째로 만나는 노드가 손실되어 정확한 확률값을 구할 수 없다. 이런 경우를 위해 손실이 많이 된 변수가 결정트리의 상단에 선택되지 않도록 하는 별도의 알고리즘을 적용하여 이러한 경우가 생기지 않도록 회피할 수 있을 것이다.

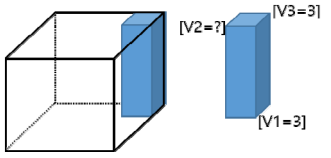
3. 결정트리에서 영역별 평균값을 산출하는 알고리즘

손실값을 위한 보간은 기존의 정보들을 이용해 추정값을 구하는 것이다. 이러한 방법들 중의 하나로 2.2 절에서 결정트리를 이용해 확률값을 구하여 손실값의 추정값으로 대입하는 방법을 소개했다. 결정트리를 이용하는 또 다른 방법으로 결정트리의 각각의 경로로부터 결정된 영역에서 사건들의 평균값을 구하고 손실값에 대해 결정트리를 순회하는 과정을 통해 이 평균값으로 손실값을 대체하는 추정값을 구하는 것이다. 따라서 2.2절의 알고리즘이 결정트리로부터 확률값을 이용했다면 새롭게 제안하는 알고리즘은 같은 결정트리로부터 평균값을 이용한다는 것이다.

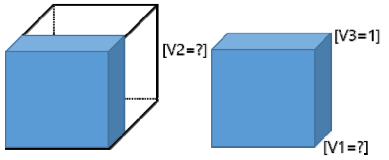
이러한 학습 과정은 2.2절의 알고리즘에서 학습 데이터를 손실 데이터와 비손실 데이터로 분류하고 비손실 데이터를 C4.5에 입력한 후 결정트리를 산출한다는 과정(1, 2번)까지는 같다. 그러나 제안 알고리즘은 확장된 데이터 표현을 사용하지 않고, 결정트리로부터 확률값 대신 평균값을 구한다는 점이 다르다. Fig. 2는 Fig. 1의 결정트리를 사용해 손실 데이터에 대해 평균값을 산출하는 방법을 설명하고 있다. 우선 Fig. 2 (a)의 {-, 1, 2}사건은 [V3=2][V2=1]이나 [V1=?]로 손실되었다. 따라서 [V3=2][V2=1]이며 V1 축에 해당하는 모든 학습 데이터의 평균을 구한 후 이를 V1 변수에 할당하게 된다. (b)의 {3, -, 3}사건 또한 [V3=3][V1=3]인 V2 축의 모든 학습 데이터의 평균값으로 V2의 손실값을 추정한다. Fig. 1의 결정트리 면에서 본다면 (a)사건은 [V3=2]와 [V2=1]를 차례로 방문한 후 7개 사건의 평균값을 구하고, (b)사건은 [V3=3][V1=3]의 방문 후에 9개 데이터의 평균값을 계산한다. (c)사건은 [V3=1]로 순회한 후 V1이 손실되었으므로 이 노드의 자식 노드에 있는 모든 데이터의 평균값을 구한다. 마지막으로 (d) 사건의 예는 결정트리의 루트 노드인 V3가 손실되었으므로 더 이상 순회할 수 없어 모든 학습 데이터의 평균값으로 손실값의 추정치를 구한다.



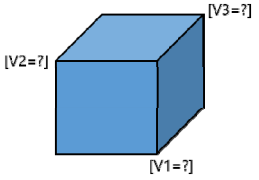
(a) {-, 1, 2} event



(b) {3, -, 3} event



(c) {-, 2, 1} event



(d) {3, 2, -} event

Fig. 2. the average value for the missing value obtained from the decision tree in Fig. 1

이와 같이 손실값에 결정트리를 이용하는 방법의 장점은 손실 데이터들에 남아있는 정보들을 이용하여 손상된 부분에 가장 근접한 추정치를 찾아간다는 것이다. 대부분의 불완전한 데이터들이 일부 정보만이 손상된 것들이기 때문에 손상된 사건들에 남아있는 많은 정보를 이용하여 유용한 정보로 복원하는 작업은 의미가 있을 것이다.

4. 실험

본 논문이 제안하고 있는 알고리즘의 성능 평가를 위해 UCI 기계학습 저장소(Machine Learning Repository)의 데이터들[15] 중에 “Balance & Scale”(625 사건, 4

변수, 3 부류)과 “Car Evolution”(1728 사건, 6 변수, 4 부류)를 사용하였다. 그리고 인간의 수면 뇌파를 기록한 “Sleep Stage Scoring”(799 사건, 11 변수, 6 부류)데이터를 사용하였다. 성능 평가를 위해 10겹 교차 검증(10 fold cross validation)을 사용하였다. 즉, 전체의 10%는 테스트 데이터로 사용하여 성능 평가하는 과정을 10번 반복하고 각 결과의 평균을 산출하였다. 그 결과를 변수와 손실률 별로 Table 1에 나타내었다. 여기서 10%의 테스트 데이터를 제외한 90%의 학습 데이터들 중에 5%, 15%, 30%, 45%의 손실비율 만큼의 사건들을 임의(random)로 선택하고, 이 사건의 각 변수에 교대로 변수값을 손실시키며 결과를 알아보았다. 손실율이 30%라는 것은 테스트 데이터를 제외한 학습 데이터 중에 30%는 손실 데이터로 나머지 70%를 비손실 데이터로 임의 분류한 후 실험이 이루어진다는 것이다.

실험 과정은 비손실 데이터로 C4.5 분류 알고리즘을 이용해 결정트리를 구성한 후, 손실데이터의 각각의 사건들을 차례로 이 결정트리를 순회하는 방식으로 구분된 영역을 찾은 다음 이 영역의 평균값으로 손실값을 대치해 나간다. 모든 손실 데이터에 대해 이러한 과정을 반복하여 평가 데이터를 완성한다. 그리고 이 평가 데이터를 SVM 알고리즘에 입력하여 분류한 후 본래의 테스트 데이터의 부류와 비교해 손실된 데이터가 얼마나 복구되는지를 평가한다.

Table 1의 결과는 결정트리를 사용한 두 알고리즘의 성능을 비교했는데, 회색은 제안된 알고리즘의 결과이고 흰색은 2.2에서 설명한 결정트리로부터 확률값을 추출하여 손실값을 추정한 결과이다. 두 가지 결과는 변수와 손실률에 따라 약간의 변화는 있으나 거의 비슷한 결과를 산출하고 있다. 그러나 이들 결과에서 보이는 값들은 앞선 논문[13,14]에 비교하였듯이 균등한 확률값을 손실값에 대치한 결과에 비해 우수한 결과임을 확인하였다. 이는 손상되지 않은 학습 데이터의 정보를 가지고, 손상된 정보에 일반화 함으로서 불완전한 데이터에 남아있는 정보를 복원하려는 의도에서 얻어진 결과라고 평가한다.

결정트리의 상부 노드에 손실값이 포함될 경우 처리 성능에 안 좋은 결과를 보였는데 이 점은 간단히 처리할 수 있다. 즉, (1) 식의 최대의 Gain_ratio를 가지는 변수가 손실된 변수라면 두 번째로 큰 값을 가지는 변수에 따라 노드를 분기하면 이러한 점을 회피할 수 있고 지금의 실험보다 상당히 나은 결과를 보일 수 있었다. 그러나 본 논문은 이렇게 인위적인 방법을 사용하지 않았고 아이디어 자체의 결과만 가지고 비교하였다.

Table 1. Performance evaluation result of two algorithms handling missing values using decision tree

(White: Probability method, Gray: Proposed method)

(a) Sleep Stage Scoring data

	5%	15%	30%	45%
V1	87.489	85.029	84.104	83.432
	87.352	86.829	87.388	87.015
V2	87.741	86.712	83.011	82.924
	87.586	86.819	86.771	86.379
V3	86.534	85.280	84.890	79.427
	87.681	88.427	85.593	85.859
V4	87.820	83.607	85.369	81.433
	88.134	87.645	88.811	88.546
V5	87.790	87.373	86.436	81.014
	88.872	88.180	87.296	87.113
V6	86.884	87.231	85.953	83.126
	87.162	87.477	87.237	85.653
V7	86.632	85.727	85.265	81.920
	88.066	88.758	87.237	85.549
V8	87.848	85.602	83.052	81.788
	87.794	87.792	85.966	87.309
V9	88.689	87.910	86.205	84.834
	87.279	87.795	87.611	87.376
V10	87.941	85.839	84.923	83.686
	88.951	86.999	85.477	86.999
V11	88.249	85.795	86.116	83.340
	89.979	89.072	87.908	88.104
Average	87.602	86.010	85.029	82.448
	88.078	87.800	87.027	86.900

(b) Car evolution data

	5%	15%	30%	45%
V1	88.603	85.893	85.499	83.779
	89.893	87.366	87.543	84.731
V2	88.104	86.919	85.064	83.537
	88.201	87.603	87.368	85.541
V3	89.005	85.083	85.083	83.252
	89.127	87.965	88.763	86.139
V4	88.273	88.137	86.336	86.454
	89.406	88.698	88.296	88.345
V5	88.119	86.648	86.171	85.137
	87.513	87.324	88.088	87.235
V6	87.868	88.838	87.848	85.726
	88.199	85.643	83.229	84.241
Average	88.329	86.920	86.000	84.648
	88.723	87.433	87.215	86.039

(c) Balance & Scale data

	5%	15%	30%	45%
V1	91.611	89.825	88.741	88.045
	90.684	91.701	89.143	84.037
V2	90.975	90.492	88.659	87.456
	90.067	90.316	89.307	87.922
V3	90.741	90.742	89.740	85.210
	91.556	91.695	87.420	82.799
V4	89.621	88.550	87.427	87.204
	91.662	90.123	86.371	85.193
Average	90.737	89.902	88.642	86.979
	90.992	90.959	88.060	84.988

5. 결론

본 논문은 손실값을 포함하는 불완전한 데이터를 처리하기 위해 손실되지 않은 정보들을 이용해 결정트리를 구성한 후, 이 트리에 손상된 사건들을 차례로 입력하여 손실값의 추정값을 찾아가는 방법을 살펴보았으며 실험을 통해 잃어버렸던 정보의 일부를 회복할 수 있는지 알아보았다. 결정트리를 구하기 위해 엔트로피를 기반으로 하는 C4.5 분류기를 이용하였으며 분류된 각 영역에서 사건들의 평균값을 계산하여 손실된 정보를 대체하도록 하였다. 또한 손실값을 평균값으로 보상 처리한 데이터의 평가를 위해 SVM 알고리즘에 입력하여 분류를 진행하여 손상되기 전의 정보와의 비교로 손상된 정보의 처리 성능을 알아보았다.

손실값을 처리하는 방법은 많이 있으나 본 논문에서 제안한 방법은 귀납적 추론 방식을 이용하는 분류기를 이용해 결정트리를 구성하고 여기서 손상된 정보를 찾아가는 방법을 사용한 점에서 기존 방법들과 차이가 있다. 실험 과정에서 결정트리의 상부 노드에서 손실값이 발견될 경우 처리 성능에 악영향을 미친다는 것을 알았고 간단한 회피 방법도 알아보았다.

다음 연구에서 이러한 회피 방법으로 얼마만큼의 성능 개선이 되는지도 알아볼 수 있을 것으로 본다. 또한 속성값의 손실뿐만 아니라 부류 또는 테스트 데이터에 손실이 있는 경우에도 적용할 수 있도록 알고리즘을 확장 발전시킬 수 있을 것이다. 마지막으로 불완전한 데이터를 자주 만나게 되는 유비쿼터스 분야에서 또는 서로 상이한 여러 센서들이나 환경에서 자료를 수집해야 하는 분야에서 제안 방법이 유용하게 쓰일 수 있기를 바란다.

REFERENCES

[1] J. Han, J. Pei & M. Kamber. (2011). Data Mining: Concepts and Techniques. *Waltham : Elsevier.*

[2] R. Kohavi & J. R. Quinlan. (2002). Data mining tasks and methods: Classification: Decision-tree discovery, Handbook of data mining and knowledge discovery. *New York : Oxford University Press, 267-276.*

[3] T. Delavallade & T. H. Dang. (2007). Using Entropy to Impute Missing Data in a Classification Task. *IEEE International Fuzzy Systems Conference.* (pp. 1-6). DOI : 10.1109/FUZZY.2007.4295430

[4] A. Sportisse, C. Boyer, A. Dieuleveut & J. Josse. (2020). Debiasing Averaged Stochastic Gradient Descent to handle missing values. *34th Conference on Neural Information Processing Systems.* (pp. 1-11). Vancouver.

[5] T. F. Johnson, N. J. B. Isaac, A. Paviolo & M. González-Suárez. (2020). Handling missing values in trait data. *Global Ecology & Biogeography, 1-12.* DOI : 10.1111/geb.13185

[6] S. Huang & C. Cheng. (2020). A Safe-Region Imputation Method for Handling Medical Data with Missing Values. *Symmetry, 12(11),* 1792. DOI : 10.3390/sym12111792

[7] J. You, X. Ma, D. Y. Ding, M. Kochenderfer & J. Leskovec. (2020). Handling Missing Data with Graph Representation Learning. *arXiv preprint arXiv:2010.16418.*

[8] J. R. Quinlan. (1993). *C4.5 : Program for Machine Learning.* San Mateo : Morgan Kaufmann.

[9] J. C. Lee, D. H. Seo, C. H. Song & W. D. Lee. (2007). FLDF based Decision Tree using Extended Data Expression. *The 6th Conference on Machine Learning & Cybernetics.* (pp.3478-3483).

[10] D. Kim, D. Lee & W. D. Lee. (2006). Classifier using Extended Data Expression. *In 2006 IEEE Mountain Workshop on Adaptive and Learning Systems.* (pp. 154-159). DOI : 10.1109/SMCALS.2006.250708

[11] J. C. Lee. (2018). Application Examples Applying Extended Data Expression Technique to Classification Problems. *Journal of the Korea convergence society, 9(12),* 9-15. DOI : 10.15207 /JKCS.2018.9.12.009

[12] J. C. Lee. (2019). Deep Learning Model for Incomplete Data, *Journal of the Korea Convergence Society, 10(2),* 1-6. DOI : 10.15207 /JKCS.2019.10.2.001

[13] J. C. Lee & W. D. Lee. (2010). Classifier handling incomplete data. *Journal of the Korea Institute of Information and Communication Engineering, 14(1),* 53-62.

[14] J. C. Lee. (2021). A data extension technique to

handle incomplete data. *Journal of the Korea Convergence Society, 12(2)*, 7-13.
DOI: 10.15207 /JKCS.2021.12.2.007

[15] Center for Machine Learning and Intelligent Systems, University of California, Irvine. (2020). *UCI Machine Learning Repository.*
https:// archive.ics.uci.edu/ml/datasets.php

이 종 찬(Jong Chan Lee) [종신회원]

· 1988년 2월 : 충남대학교 계산통계학과 (학사)
· 1990년 2월 : 충남대학교 대학원 전산학과(석사)
· 1996년 2월 : 충남대학교 대학원 전산학과(박사)
· 1996년 3월 ~ 현재 : 청운대학교 컴퓨터공학과 교수
· 관심분야 : 딥러닝, 패턴분류, 정보보호, 데이터압축
· E-Mail : jclee@chungwoon.ac.kr