Journal of the Korean Institute of Industrial Engineers
Vol. 41, No. 5, pp. 439-446, October 2015.
ISSN 1225-0988 | EISSN 2234-6457
http://dx.doi.org/10.7232/JKIIE.2015.41.5.439

<Original Research Paper>

Hyper-Rectangles를 이용한 단일 분류기 설계

정인교 · 최진영†

아주대학교 산업공학과

Design of One-Class Classifier Using Hyper-Rectangles

In Kyo Jeong · Jin Young Choi

Department of Industrial Engineering, Ajou University

Recently, the importance of one-class classification problem is more increasing. However, most of existing algorithms have the limitation on providing the information that effects on the prediction of the target value. Motivated by this remark, in this paper, we suggest an efficient one-class classifier using hyper-rectangles (H-RTGLs) that can be produced from intervals including observations. Specifically, we generate intervals for each feature and integrate them. For generating intervals, we consider two approaches : (i) interval merging and (ii) clustering. We evaluate the performance of the suggested methods by computing classification accuracy using area under the roc curve and compare them with other one-class classification algorithms using four datasets from UCI repository. Since H-RTGLs constructed for a given data set enable classification factors to be visible, we can discern which features effect on the classification result and extract patterns that a data set originally has.

Keywords: Hyper-Rectangles, One-Class Classification, Interval Merging, Interval Conjunction, Clustering

1. 서 론

분류 문제는 데이터 마이닝, 기계 학습, 인공 지능 등의 분야에서 다루고 있는 가장 기본적인 이슈들 중 하나로서, 인스턴스가 가지고 있는 속성 값들을 이용해 그 인스턴스가 속하는 클래스를 결정하는 것이다. 이러한 분류 문제는 고려되는 클래스의 개수에 따라 다중 분류와 단일 분류로 나눌 수 있다(Tax, 2001). 다중 분류는 여러 가지 클래스의 데이터를 이용해 분류기를 만들고, 새로운 인스턴스가 그 클래스들 중 어디에 속하는지를 예측한다. 반면 단일 분류는 오직 하나의 타겟 클래스만을 이용해 분류기를 만들며, 분류 결과가 해당 클래스에 속하는지 아닌지에 따라 양(positive) 또는 음(negative)으로 판별한다. 일반적으로 단일 분류 문제는 비교할 수 있는 다른 클래스가 없어서 타겟 클래스의 고유 특성을 알기가 모호하다는 점 때문에 다중 분류 문제보다 어렵다.

최근 들어 단일 분류 문제는 클래스 정보의 제한성 및 빅데이터로 인한 다중 분류기 구축의 계산 복잡성 등의 이유로 그 중요성이 점점 커지고 있다. Tax(2001)의 문헌에서는 이에 대한 몇 가지 예를 제시한다. 사과와 복숭아 데이터만 있는 경우, 이를 이용해 다중 분류 모델을 만들었다고 하자. 그런데 새로운 인스턴스로 고양이를 주면 이 모델은 고양이를 사과 또는 복숭아로 분류할 것이다. 만일 고양이에 대한 정보가 없다면 단순히 사과 또는 복숭아인지 아닌지를 판별하는 게 옳을 것이다. 이처럼 알 수 없는 이기종의 클래스들이 다수 존재하는 상황에서 모든 클래스에 대한 정보가 제한적이라면 다중 분류가 적합하지 않을 수 있다. 비슷한 예로서, 공장에서 기계를 작동시킬 때 정상적인 안정 상태는 비슷하지만, 이상 상태의 경우는 매우 다양하다. 전기적인 문제일 수도 있고 부품 결함일 수도 있다. 따라서 기계의 상태를 파악하기 위해서는 이상 상태보다는 정상 상태의 데이터를 이용하는 게 정확하고 계산도 쉬울 것이다. 한편, 어떤 경우에는 전적으로 계산 복잡도 때문에 단일 분류를 수행하기도 한다. 데이터 집합의 사이즈가 아주

† 연락저자 : 최진영 교수, 16499 수원시 영통구 원천동 월드컵로 206 아주대학교 산업공학과, Tel : 031-219-2422, Fax : 031-219-1610,
E-mail : choijy@ajou.ac.kr
2015년 7월 27일 접수; 2015년 9월 3일 수정본 접수; 2015년 9월 3일 게재 확정.

크다면 다중 분류 모델을 형성하는데 높은 계산 복잡도가 요구된다. 최근에는 데이터의 크기가 수천 테라바이트 또는 페타바이트 등으로 점점 더 커지고 있기 때문에 다중 분류보다 단일 분류가 계산량을 큰 폭으로 줄일 수도 있다.

기본적으로 단일 분류를 수행하는 방법은 주어진 데이터를 이용해 폐곡선을 그리는 것이다. Tax and Duin(1999a, 1999b)는 이러한 개념을 이용하여 Support Vector Data Description (SVDD) 방법을 제안하였다. 이 방법은 새로운 인스턴스가 폐곡선 안에 있으면 양으로, 밖에 있으면 음으로 분류한다. SVDD는 소수의 데이터만으로도 데이터에 정확한 경계를 그릴 수 있다는 장점을 갖는다. 또한 확률 분포를 이용해 단일 분류를 수행할 수도 있다(Tax, 2001). 이 방법에서는 주어진 데이터를 이용하여 타겟 클래스의 분포를 추정하고, 새로운 인스턴스가 타겟 클래스 데이터들의 일정 범위 내에 존재할 확률을 계산한다. 이 확률이 클수록 양일 가능성이 커진다는 사실에 근거하여 설정된 기준값에 따라 새로운 인스턴스를 양 또는 음으로 분류한다. 확률 분포를 이용한 분류는 결측치나 에러에 강건한 편이다. 그러나 이러한 기존 단일 분류 알고리즘들은 분류 요인을 파악할 수 없는 블랙박스 형태라는 결정적인 한계점을 지닌다(Baehrens *et al*., 2010). 즉, 인스턴스가 왜 그렇게 분류됐는지를 알 수 없다. 예를 들어 단일 분류를 통해 기계가 이상 상태로 접어들었다고 해도, 어떤 요인 때문에 그렇게 분류되었는지를 알 수 없다는 것이다.

본 논문에서는 기존의 단일 분류기와 비교할 때, 블랙박스 형태의 한계점을 극복할 수 있으면서 동시에 경쟁력 있는 분류 정확도를 갖는 효율적인 단일 분류 방법으로서 Hyper-rectangles (H-RTGLs)를 이용한 단일 분류기를 제안하고자 한다. 이 방법은 타겟 클래스의 데이터를 이용하여 적절한 개수와 볼륨을 갖는 H-RTGL들을 생성하여 분류 모델을 구축한다. 제안된 단일 분류기는 H-RTGLs를 통해 분류 요인에 대한 직관적인 해석이 가능하다는 장점을 갖는다. 본 논문에서는 H-RTGLs를 이용하는 두 가지 단일 분류기를 제안한 후, 다른 단일 분류기와 비교하여 분류 알고리즘의 성능을 검증하고, 분류 요인을 이용하여 분류 결과를 해석하는 방법을 설명한다.

본 논문의 구성은 다음과 같다. 먼저 제 2장에서는 단일 분류 방법에 대한 연구동향을 소개한다. 제 3장에서는 본 연구에서 제안하는 두 가지 H-RTGLs를 이용한 단일 분류기를 설명한다. 제 4장에서는 UCI machine-learning repository에 있는 대표적인 4가지 데이터를 이용하여 수행한 수치 실험을 통해 제안된 단일 분류 알고리즘의 성능을 검증하며, H-RTGL이 제공하는 해석력을 기술한다. 마지막으로 제 5장에서 결론을 내리고 추후 연구를 제안한다.

2. 관련 연구 동향

최근에 Khan and Madden(2014)는 단일 분류 방법을 체계적으로 정리하여 크게 Support Vector Machines(SVM) 기반과 비-SVM 기반으로 나누었다. SVM은 다중 분류를 위해 Cortes and Vapnik(1995)가 고안한 기법으로, 서로 다른 클래스 사이에 support vector를 그린 후 새로운 인스턴스가 그것의 어느 편에 있는지를 기준으로 분류를 수행한다. 이러한 SVM 기반 단일 분류에는 두 가지 접근법이 존재한다. 하나는 제 1장에서 언급한 SVDD이고, 다른 하나는 커널 기법을 이용해 hyper-plane을 그리는 one-class SVM이다(Schölkopf *et al*., 2000). 이 두 기법은 Gaussian kernel을 이용할 때 동일해지며 가장 좋은 성능을 보인다(Tax, 2001). SVM 기반의 단일 분류 방법은 높은 분류 정확도를 가지면서 사용하기에도 쉬운 편이기 때문에 가장 많이 이용되는 기법이다.

비-SVM 기반의 단일 분류는 다중 분류 기법들이 응용된 경우가 대부분이며, 대표적으로 Decision tree, Artificial Neural Network(ANN)을 들 수 있다. Decision tree는 룰 기반의 알고리즘으로 룰을 통해 데이터를 점진적으로 분류하는 기법이다. 알고리즘이 간결하고 모델을 쉽게 이해할 수 있다는 장점 때문에 분류 요인을 해석하는 것이 중요한 분야에서 많이 사용된다. 일반적인 Decision tree 방법으로는 CART나 C4.5(Breiman *et al*., 1983; Quinlan, 1993) 등이 있으며 이를 기반으로 하는 다양한 단일 분류 기법이 제안되었다. Letouzey *et al*.(2000)은 한 가지 클래스의 데이터 만으로 C4.5를 구현하는 POSC4.5를 개발했다. 또한, Li *et al*.(2009)은 one-class data stream에 사용하기 위해 속도가 향상된 버전의 Decision tree(Domingos and Hulten, 2000)를 응용한 OcVFDT을 제안했다. Breiman(2001)은 기존의 Decision tree가 과적합(overfitting)되는 점을 극복하기 위해 다수의 tree를 형성하는 Random forests를 제안했다. Désir *et al*. (2012)은 이를 이용한 단일 분류 기법으로 one-class random forests를 제안했다.

ANN은 생물의 신경망이 뉴런을 통해 정보를 주고받는 것처럼 다수의 perceptron으로 신경망을 형성해 일련의 계산 과정으로 분류를 수행하는 기법이다(Khan *et al*., 2001). 이 기법은 주어진 데이터를 가장 정확하게 묘사할 수 있는 알고리즘으로 평가받고 있다. 최근에는 이것의 응용 버전이라고 할 수 있는 Deep learning이 크게 주목 받고 있어 그 사용이 두드러지고 있는 추세다(Schmidhuber, 2015). ANN을 이용한 단일 분류 방법으로 Manevitz and Yousef(2000)은 한 가지 클래스 밖에 사용할 수 없다는 제한을 해결하기 위해 병목층(bottleneck layer)을 이용한 Autoencoder neural network를 제안했다. 또한 이를 one-class document를 분류하는 데에 사용하였다(Manevitz and Yousef, 2007). 그리고 Skabar(2003)은 ANN으로 데이터 집합의 사후확률분포(posterior probability distribution)를 구할 수 있다는 점에 착안해, 한 가지 클래스 데이터와 클래스를 모르는(unlabeled) 데이터로 ANN을 구축하는 기법을 제안하였다. 이외에도 다양한 다중 분류 알고리즘들이 단일 분류를 위해 응용되었다. 그러한 연구들에는 Nearest neighbors(Tax, 2001), Gaussian process (Kemmler *et al*., 2013), Minimum spanning tree(Juszczak *et al*., 2009), Fuzzy system(Bosco and Pinello, 2009; Utkin, 2012) 등이 있다.

그러나 지금까지 제안된 단일 분류 방법들 중에서 분류 결과에 대한 요인을 해석할 수 있는 기법은 Decision tree와 Fuzzy system 기반의 방법들뿐이다. 두 가지 모두 룰 기반의 알고리즘으로서 Decision tree는 역치(threshold)를 기준으로 명확히 분류하는 hard boundary를 가지며, Fuzzy system은 역치로부터 떨어진 거리를 계산하여 특정 확률 분포를 갖는 soft boundary를 갖는다(Hüllermeier, 2011). 이 외의 다른 단일 분류 알고리즘들은 분류 요인에 대해 해석할 수 없는 블랙박스 형태라는 한계를 갖는다.

3. Hyper-Rectangles를 이용한 단일분류기

이 장에서는 Jeong and Choi(2015)에 기반한 H-RTGLs를 이용한 두 가지 단일 분류 알고리즘을 제안한다. 이를 위해 먼저 n개의 인스턴스 $x_i(i=1, 2, \cdots, n)$으로 구성된 데이터 집합 $\Omega=\{x_1, x_2, \cdots, x_n\}$를 고려한다. 이 때, 각 인스턴스 x_i는 m개의 속성(feature) $y_{ir}(r=1, 2, \cdots, m)$을 가진다($x_i=(y_{i1}, y_{i2}, \cdots y_{im}), i=1, 2, \cdots, n$).

3.1 병합기반 H-RTGLs를 이용한 단일 분류기

병합기반 H-RTGLs(merging-based hyper-rectangles : *MbH*)를 이용한 단일 분류기는 다음과 같은 5단계를 거쳐 생성된다.

(1) 인터벌 생성

주어진 데이터 집합 Ω에 포함된 인스턴스들을 각 속성마다 사영(projection)시킨다. 인스턴스 x_i의 속성 r에 대한 사영은 다음과 같이 정의된다.

$$proj_r(x_i)=y_{ir},\ \forall i, r$$

n개의 인스턴스를 속성 n에 대해 사영했을 때 $n_0^r(\le n)$개의 점(point)들을 얻게 되며, 각각의 사영점(projection point)마다 y_{ir}을 포함하는 하나의 인터벌을 다음과 같이 생성한다.

먼저 속성 r에 대한 사영점들 간 최소거리(minimum distance)를 다음과 같이 정의한다.

$$\delta_r = \min_{i\neq j,\, dist>0} dist(proj_r(x_i), proj_r(x_j)),\ \forall r,$$

여기서, $dist(proj_r(x_i), proj_r(x_j))=|y_{ir}-y_{jr}|$이다. 또한, 집합 X에서 집합 Y로의 함수 f에 대해 원상(preimage) 함수를

$$f^{\leftarrow}(y)=\{x\in X|f(x)=y\}\ \forall\quad y\in Y$$

로 정의하면, 사영점 y_{ir}을 갖는 인스턴스들의 개수를 $|proj_r^{\leftarrow}(y_{ir})|$로 표현할 수 있다. 만일 임의의 y_{ir}에 대해 $|proj_r^{\leftarrow}(y_{ir})|>1$이라면, 이에 해당하는 인스턴스들은 속성 r에 대해서 동일한 인터벌을 갖는다. 이러한 경우, $|proj_r^{\leftarrow}(y_{ir})|$에 따라 인터벌의 길이를 $(1+\alpha), (\alpha\ge 0)$배 만큼 늘려줄 수 있도록

$$\sigma_r(x_i)=(1+\alpha)^{|proj_r^{\leftarrow}(y_{ir})|-1}\quad \forall i, r$$

를 정의하여 해당하는 인스턴스들의 밀도를 생성되는 인터벌 길이에 반영한다. 만일

$$\sigma_r^*=\max_i \sigma_r(x_i),\quad \forall r$$

라고 하면,

$$\frac{\delta_r\cdot\sigma_r(x_i)}{\sigma_r^*}\le\delta_r,\ \forall i, r$$

인 관계가 성립한다.

이를 이용하여 인스턴스 x_i의 속성 r에 대한 인터벌 길이를

$$\theta_r(x_i)=w\cdot\delta_r\cdot\frac{\sigma_r(x_i)}{\sigma_r^*},\quad \forall\ w\ge 1, i, r$$

로 정의한다. 여기서 w는 인터벌 길이를 조절하는 파라메터이다. 만일 $w=1$이면, $\theta_r(x_i)\le\delta_r,\ \forall i$이 되어 겹치는 인터벌이 없게 되며, $w>1$이면 겹치는 인터벌들이 존재할 수 있고 이들은 병합될 수 있다. 따라서 인스턴스 x_i의 속성 r에서의 인터벌은 사영점 y_{ir}을 중심으로

$$itvl_r(x_i)=\left[proj_r(x_i)-\frac{\theta_r(x_i)}{2}, proj_r(x_i)+\frac{\theta_r(x_i)}{2}\right],\ \forall i, r$$

로 생성된다.

(2) 인터벌 병합

데이터 집합 Ω의 속성 r에 대한 인터벌 생성시 겹치는 인터벌이 존재하기 위해서는 w가 다음의 조건을 만족해야 한다.

(Property 1)

만일 $w>\min_{x_i, x_j\in\Omega}\frac{|y_{ir}-y_{jr}|}{\delta_r}\cdot\frac{2\sigma_r^*}{\sigma_r(x_i)+\sigma_r(x_j)}$이면, 데이터 집합 Ω의 속성 r에 대한 인터벌들 중에는 적어도 하나의 겹치는 인터벌이 반드시 존재한다.

(증명) 만일 인스턴스 (x_i, x_j)의 속성 r에 대한 인터벌이 겹친다면, $\frac{\theta_r(x_i)+\theta_r(x_j)}{2} > |y_{ir}-y_{jr}|$이고, $\frac{\theta_r(x_i)+\theta_r(x_j)}{2} = w \cdot \delta_r \cdot \frac{\sigma_r(x_i)+\sigma_r(x_j)}{2\sigma_r^*}$이므로, w에 대해 정리하면 $w > \frac{|y_{ir}-y_{jr}|}{\delta_r} \cdot \frac{2\sigma_r^*}{\sigma_r(x_i)+\sigma_r(x_j)}$라는 조건을 얻을 수 있다. 이 때, 부등식의 우변 값은 인스턴스 (x_i, x_j) 조합에 따라 다양한 값을 가지며, w가 이 값들 중 최소값보다 크다면 적어도 하나의 겹치는 인터벌이 반드시 존재하게 된다.

또한 겹치는 인터벌들은 다음과 같이 greedy한 방법으로 병합될 수 있다 : $itvl_r(x_i)$에 대한 인터벌 병합은 먼저 $ITVL_r^t \leftarrow itvl_r(x_i)$로 하고 남아있는 모든 인스턴스들에 대해서

$$ITVL_r^t \leftarrow ITVL_r^t \cup itvl_r(x_j) \; if \; ITVL_r^t \cap itvl_r(x_j) \neq \phi$$

를 반복해서 적용한다. 그 결과, 속성 r에 대해 서로 겹치지 않는 인터벌 집합

$$I_r = \{ITVL_r^1, ITVL_r^2, \cdots, ITVL_r^{qr}\}, \; \forall r$$

을 얻을 수 있다. 여기서, q_r은 속성 r에 대해 병합 과정으로 얻게 된 전체 인터벌 개수이다.

(3) 인터벌 결합

모든 속성에 대해서 인터벌 병합을 수행하면 m개의 겹치지 않는 인터벌 집합 $I_r(r=1, 2, \cdots, m)$을 얻을 수 있다. 이를 이용하여 m-폴드 인터벌 프로덕트(m-fold interval product)

$$I = I_1 \times I_2 \times \cdots \times I_m$$

를 만들고, 이로부터 데이터 집합 Ω에 대해 인터벌 결합(conjunction)을 추출한다. 인터벌은 데이터의 사영점을 중심으로 생성됐다는 점과 인터벌 병합이 인터벌들의 합집합(union)을 통해 이뤄졌다는 점으로 인해, 인스턴스 x_i의 속성 r에 대한 사영점 y_{ir}을 커버하는 인터벌은 I_r에 항상 존재한다. 따라서 인터벌 결합은 인스턴스를 이용해 인터벌 프로덕트 I에서 속성마다 해당하는 인터벌들을 하나씩 찾아서 만들 수 있다. 즉, I_r에 포함된 인터벌 중에서 인스턴스 x_i의 속성 r을 포함하는 인터벌을 $ITVL_r(x_i)$라 하면, 인스턴스 x_i에 대해 결합된 인터벌은 다음과 같다.

$$\pi(x_i) = ITVL_1(x_i) \wedge ITVL_2(x_i) \wedge \cdots \wedge ITVL_m(x_i), \; \forall i$$

이렇게 생성된 인터벌들은 모든 속성에서 겹치지 않기 때문에 하나의 인스턴스는 오직 한 개의 인터벌 결합에 속한다. 반면 한 개의 인터벌 결합은 여러 개의 인스턴스들을 포함할 수 있다. 즉, 동일한 인터벌 결합을 갖는 인스턴스들이 존재할 수 있다.

(4) *MbH* 생성

결합된 인터벌 $\pi(x_i)$는 H-RTGL의 형태를 가지고 있지만 모든 속성을 동시에 고려하지 않고 각 속성마다 독립적으로 생성된 것이기 때문에 커버되는 데이터에 비해 불필요하게 클 수도 있다. 따라서 이번 단계에서는 다음과 같이 $\pi(x_i)$들에 대한 fitting을 통해 조밀한 최종 *MbH*를 만든다 : 먼저, 인터벌 결합 $\pi(x_i)$에 포함되는 인스턴스들을 속성 r에 사영한다. 그 다음 사영한 점들에서 최소점(minimum point)와 최대점(maximum point)을 찾는다. 마지막으로 그것을 각각 인터벌 $ITVL_r(x_i)$의 새로운 왼쪽 끝점(left end point)와 오른쪽 끝점(right end point)으로 정의한다 (만일, 인스턴스가 한 개이면 $ITVL_r(x_i)$은 변하지 않음). 이 때, 파라메터 v를 도입해 인터벌 길이를 조절할 수 있으며, 최종적으로 $\pi(x_i)$를 속성 r에 fitting 시킨 결과는 다음과 같다.

$$Fit_r(\pi(x_i)) = \Big[\min_{x_j \in \pi^{\leftarrow}(\pi(x_i))} proj_r(x_j) - v, \; \min_{x_j \in \pi^{\leftarrow}(\pi(x_i))} proj_r(x_j) + v \Big] \; \forall v \geq 0, r$$

모든 속성에 대한 fitting 과정을 통해 $\pi(x_i)$는 조밀한 *MbH*가 되며 다음과 같이 표현된다.

$$MbH(x_i) = Fit_1(\pi(x_i)) \wedge Fit_2(\pi(x_i)) \wedge \cdots \wedge Fit_m(\pi(x_i)), \; \forall i$$

*MbH*는 $\pi(x_i)$를 fitting시켜 만들기 때문에 생성되는 *MbH*의 개수는 $\pi(x_i)$의 개수와 같다. 그리고 $\pi(x_i)$들 간은 겹치지 않지만 *MbH*는 v를 통해 크기를 조절하기 때문에 서로 겹치는 부분이 생길 수 있다.

(5) 단일 분류기 구축

데이터 집합 Ω에 대해서 생성된 *MbH* 중에서 동일한 것들을 제외하고 최종적으로 정리된 n_1개의 *MbH*들의 집합을

$$S_{MbH} = \{MbH^1, MbH^2, \cdots, MbH^{n_1}\}$$

라고 하면, 이를 이용한 단일 분류기는 다음과 같이 정의될 수 있다.

$$f_{MbH}(x) = \sum_{\tau=1}^{n_1} I_{\{x \in MbH^\tau\}}.$$

이 때, I_A는 A가 참일 때 1의 값을 갖는 정의 함수 (indicator function)이다. 즉, 새로운 인스턴스 x는 생성된 *MbH* 집합 S_{MbH}에 의해 커버되는지의 여부로 1(양) 또는 0(음)로 분류된다.

3.2 군집기반 H-RTGLs를 이용한 단일 분류기

군집기반 H-RTGLs(clustering-based hyper-rectangles : *CbH*)를 이용한 단일 분류기는 다음과 같은 5단계를 거쳐 생성된다.

(1) *K*-Means를 이용한 군집 생성

K-Means 알고리즘(MacQueen, 1967)은 가장 널리 알려진 군집 생성 방법 중 하나이다. 데이터 집합에 적합한 *K*를 찾는 명확한 방법이 없다는 단점이 있지만, 비교적 빠르고 강력하기 때문에 여전히 많이 이용되고 있다. 이 알고리즘은 군집의 평균(mean)을 이용해 전체 데이터 집합을 *K*개로 나누는 것이다. 이때 각 군집의 평균을 centroid라고 하면, 알고리즘은 *K*개의 centroid가 더 이상 변하지 않을 때까지 반복된다.

(2) 인터벌 생성

인스턴스 개수가 p_k인 k번째 군집 C_k를

$$C_k = \left\{ x_1^{C_k}, x_2^{C_k}, \cdots, x_{p_k}^{C_k} \right\}, \ \forall k$$

라고 정의하면, 모든 군집들의 합집합은 전체 데이터 집합 Ω가 되고, 군집 간은 서로 겹치지 않는다.

$$\bigcup_{k=1}^{K} C_k = \Omega \text{ and } \bigcap_{k=1}^{K} C_k = \phi$$

이 때, H-RTGL을 만들기 위해 필요한 인터벌은 군집에 속한 인스턴스들을 각 속성마다 사영 시켜 만들 수 있다. 즉, 군집 C_k의 인스턴스들을 속성 r에 사영시키고 그 점들의 최소점과 최대점을 각각 왼쪽 끝점과 오른쪽 끝점으로 정의하여 인터벌 $ITVL_r^{C_k}$를 다음과 같이 생성한다.

$$ITVL_r^{C_k} = \left[\min_{x_j^{C_k} \in C_k} proj_r(x_j^{C_k}), \ \max_{x_j^{C_k} \in C_k} proj_r(x_j^{C_k}) \right], \ \forall r$$

Ω는 K개의 군집으로 구성되어 있고, 군집 내 인스턴스는 m개의 속성을 가지므로 모두 $K \cdot m$개의 인터벌이 생성된다.

(3) 인터벌 결합

군집의 관점에서 보면 각 속성마다 인터벌을 하나씩 생성했으므로 군집 C_k에 속하는 모든 인스턴스들에 대한 인터벌은 그것들을 결합하여 다음과 같이 얻을 수 있다.

$$\pi(C_k) = ITVL_1^{C_k} \wedge ITVL_2^{C_k} \wedge \cdots \wedge ITVL_m^{C_k}, \ \forall k$$

(4) *CbH* 생성

이 단계에서는 생성된 H-RTGLs의 크기를 파라메터 v를 이용하여 다음과 같이 fitting한다.

$$Fit_r(\pi(C_k)) = \left[\min_{x_j^{C_k} \in C_k} proj_r(x_j^{C_k}) - v, \ \max_{x_j^{C_k} \in C_k} proj_r(x_j^{C_k}) + v \right] \ \forall v \geq 0, r$$

또한 *CbH*는 $\pi(C_k)$를 모든 속성에 대해서 fitting한 후 이들의 결합에 의해 다음과 같이 생성된다.

$$CbH(C_k) = Fit_1(\pi(C_k)) \wedge Fit_2(\pi(C_k)) \wedge \cdots \wedge Fit_m(\pi(C_k)), \ \forall k$$

이 때, *CbH*의 개수는 $\pi(C_k)$의 개수와 같은 K개이다.

(5) 단일 분류기 구축

데이터 집합 Ω에 대해서 생성된 K개의 *CbH*들의 집합을

$$S_{CbH} = \{ CbH^1, CbH^2, \cdots, CbH^K \}$$

라고 하면, 이를 이용한 단일 분류기는 다음과 같이 정의될 수 있다.

$$f_{CbH}(x) = \sum_{\tau=1}^{K} I_{\{x \in CbH^\tau\}}$$

즉, 새로운 인스턴스 x는 생성된 CbH 집합 S_{CbH}에 의해 커버되는지의 여부로 1(양) 또는 0(음)로 분류된다.

4. 수치 실험

4.1 실험 설계

본 연구에서는 제안된 H-RTGLs 기반 단일 분류기의 성능을 평가하기 위하여 UCI machine-learning repository에 있는 Iris, Wisconsin Breast Cancer(WBC), Liver Disorder(Liver), E.coli 등 총 네 가지 데이터 집합을 활용하였다(Asuncion and Newman, 2007). 그러나 이 데이터 집합들은 본래 여러 개의 클래스로 구성되어 있고, 단일 분류 알고리즘은 오직 한 가지 클래스의 데이터만으로 모델을 형성하는 것이기 때문에 사용할 한 가지 클래스를 정해주어야 한다. 본 연구에서는 이를 타겟 클래스라 하고, 나머지 클래스는 비-타겟 클래스라고 정의한다. 데이터 집합에 대한 요약은 <Table 1>과 같다.

Table 1. Summary of 4 data sets from UCI repository

Data set	*Iris*	*WBC*	*Liver*	*E.coli*
Num. of features	4	9	6	7
Target class	versicolor	malignant	healthy	periplasm
Target size	50	241	145	52
Non-target size	100	458	200	284

단일 분류 모델을 구축하는데 사용되는 학습 데이터는 타겟 클래스에서 임의로 50%를 선택하였다. 분류기 성능 분석을 위한 테스트 데이터는 학습에 사용되지 않은 타겟 클래스의 나머지 데이터와 비-타겟 클래스 데이터 전부를 사용하였다. 예

를 들어 Iris 데이터 집합을 보면, 25개의 versicolor 클래스의 데이터를 학습해 H-RTGL 모델을 형성하였다. 그리고 나머지 25개의 versicolor 클래스 데이터와 나머지 100개의 비-타겟 클래스 데이터 등 총 125개의 데이터를 테스트 데이터로 사용하였다.

또한, 제안된 단일 분류기의 성능을 평가하기 위해서 Receiver Operating Characteristics(ROC) curve를 사용하였다. ROC curve는 true positive rate과 false positive rate를 두 축으로 하는 그래프에서 분류기의 파라메터를 바꿔주면서 true positive rate과 false positive rate의 관계를 그린 curve이다. True positive rate는 분류기 모델이 양으로 분류한 데이터 중에서 실제 양인 데이터의 비율이다. 반면, false positive rate는 분류기 모델이 음으로 분류한 데이터 중에서 실제로는 양인 데이터의 비율이다. 이 때, Area Under the ROC curve(AUC)는 분류 성능의 척도가 된다. 즉, AUC가 1에 가까울수록 분류 모델이 강건하고 분류 정확도가 높다는 의미이고, 0.5보다 작으면 무작위 추측(random guess)보다 못해 분류 모델이 가치 없음을 나타낸다(Juszczak *et al*., 2009).

일반적으로 단일 분류에서 AUC를 계산하기 위해 ROC curve를 그리는 방법은 두 가지가 있다. 하나는 단일 분류기가 인스턴스마다 점수를 계산하고 설정된 역치보다 높으면 양으로, 낮으면 음으로 분류하는 것이다. 이 때 역치를 파라메터로 해서 ROC curve를 그린다. 다른 방법은 인스턴스가 미리 지정된 범위 안에 포함되면 양으로, 포함되지 않으면 음으로 분류하는 것이다. 이 때는 지정하는 범위를 파라메터로 변화시키면서 ROC curve를 그린다. 본 연구에서는 두 번째 방법으로 ROC curve를 그려 AUC를 계산했다.

또한 제안된 분류기는 H-RTGLs의 개수와 크기를 결정하는 두 가지 종류의 파라메터를 가지고 있다. 파라메터 w와 K는 H-RTGLs의 개수를 결정하고 파라메터 v는 H-RTGLs의 크기를 결정한다. 따라서 개수를 고정하고 크기를 조절하거나, 반대로 크기를 고정하고 개수를 조절하면서 서로 다른 ROC curve를 그릴 수 있다. 예를 들어, *MbH*를 이용한 단일 분류기에서 개수를 고정하고 크기를 조절하기 위해서 w를 1.5로 고정하고 v를 변화시키면서 ROC curve를 그린다. 이와 같은 방법으로 두 개의 파라메터에 대한 AUC를 계산하였다. 파라메터 v를 이용해 크기를 조절하면서 얻은 AUC에는 'vol'을, 파라메터 w를 이용해 개수를 조절하면서 얻은 AUC에 'num'을 붙여 두 가지를 구분하였다. 추가적으로 *MbH* 기반 분류기에서 중복되는 인스턴스의 개수에 따라 인터벌 길이를 늘려주는 파라메터인 α는 실험을 통해 너무 값이 크지만 않으면 분류 결과에 많은 영향을 미치지 않는 것으로 파악되었다.

4.2 실험 결과 분석

구축한 단일 분류 모델을 이용하여 4가지 데이터 집합에 대해 분류 실험을 20회씩 반복 적용하여 수행한 후 AUC를 계산하고 그것의 평균과 표준편차를 구했다. <Table 2>는 실험 결과를 나타내며, 성능을 비교하기 위한 다른 분류 알고리즘들의 AUC는 관련 문헌을 참고했다(Juszczak *et al*., 2009; Tax, 2010).

대체적으로 제안된 H-RTGLs를 이용한 단일 분류기가 기존의 다른 단일 분류 방법과 비교해 좋은 성능을 보였다. 특히, *MbH* 기반 단일 분류기는 4개 중 2개의 데이터 집합에서 가장 좋은 분류 성능을 나타내는 AUC 값을 가졌다. 또한 *MbH* 기반 단일 분류기는 *CbH* 기반 단일 분류기보다 더 좋은 분류 성능을 보였다. 그러나 제안된 H-RTGLs를 이용한 단일 분류 방법은 인스턴스를 기반으로 인터벌을 생성하기 때문에 다른 방법들에 비해 표준 편차가 다소 높은 것을 확인할 수 있었다. 특히, *CbH* 기반 단일 분류기의 데이터 의존성이 큰 것을 확인할 수 있었다. 그렇지만 계산된 표준편차의 값은 크게 유의하지 않음을 확인할 수 있었다.

파라메터 감도 분석에 대한 실험 결과로는 H-RTGLs의 개수를 고정하고 크기를 변화시켜 주는 것이, 크기를 고정하고 개수를 변화시키는 것 보다 더 좋은 분류 성능을 가짐을 알 수 있었다. 가장 높은 AUC 값을 갖는 파라메터 값에 대한 분석은 다음과 같다. *MbH* 기반 단일 분류기에서 H-RTGLs의 개수를 결정하는 w와 α의 값이 데이터 집합마다 다른 것은 데이터 집합마다 최적의 H-RTGL 개수가 존재한다는 것을 의미한다. E.Coli 데이터 집합에서 $\alpha = 0$인 것은 인터벌이 여러 개의 인스턴스를 포함해도 인터벌 길이가 커지지 않아야 한다는 것이다. 파라메터 v값들이 전체적으로 크지 않은 것은 H-RTGL의 크기가 너무 크지 않고 데이터를 조밀하게 커버해야 좋다는 것을 의미한다. *CbH* 기반 단일 분류기의 파라메터 값을 보면, K는 대체적으로 작은데 Liver 데이터 집합에서만 47이다. 이는 145개의 타겟 데이터가 47개의 군집으로 나눠졌을 때 분류가 잘 되었다는 의미이며, 타겟 데이터의 특성이 개별적이란 것이다. 이 사실은 Liver 데이터의 AUC 값이 전체적으로 작다는 사실로도 다시 한 번 확인할 수 있다. 또한 여기서도 v가 크지 않은 것으로 보아 H-RTGL의 크기는 작아야 한다는 것을 알 수 있다.

제안된 H-RTGLs를 이용한 단일 분류기의 장점은 높은 분류 성능을 가지면서 동시에 분류에 대한 해석이 가능하다는 점이다. 좀 더 자세히 설명하면, 양으로 분류되는데 있어서 인스턴스가 갖는 값들이 분류에 어떤 기여를 하는지 파악 할 수 있다는 것이다. 일반적으로 분류에 대한 해석을 위해서는 룰 기반의 알고리즘을 사용하거나, 룰 기반이 아닌 알고리즘에서 룰을 추출하는 방법을 고안하여 사용한다. 본 연구에서 제안된 방법은 룰 기반의 알고리즘으로, H-RTGLs의 인터벌들이 룰의 역할을 한다. 따라서 H-RTGL은 룰의 논리곱으로 표현된 데이터 집합의 패턴이라고 할 수 있다.

예를 들어, <Table 3>은 Breast 데이터 집합을 대상으로 만들어진 *MbH* 기반 단일 분류기의 일부를 나타낸다. 구체적으로, malignant 클래스에서 *MbH* 기반 단일 분류기($\alpha = 0.06$, $w = 8$, $v = 0.01$)로 생성된 26개의 *MbH*들 중 커버하는 인스턴스가 많은 세 가지 *MbH*의 예이다.

Table 2. Experimental results of the proposed algorithms and other one-class classifiers

Classifiers \ Data set	*Iris*	*Breast*	*Liver*	*E.coli*
	AUC×100(standard deviation)			
*MbH_**vol***	98.5(0.8)	95.1(1.2)	**61.6(1.6)**	**94.6(1.5)**
	$\alpha = 0.2,\ w = 2$	$\alpha = 0.06,\ w = 12$	$\alpha = 0.02,\ w = 8$	$\alpha = 0,\ w = 16$
*MbH_**num***	97.5(1.1)	96.1(0.9)	58.8(2.1)	90.9(2.5)
	$\alpha = 0.2,\ v = 22$	$\alpha = 0.06,\ v = 0.1$	$\alpha = 0.2,\ v = 0.1$	$\alpha = 0,\ v = 0.012$
*CbH_**vol***	98.6(0.4)	85.8(11.1)	59.2(2.3)	92.3(1.7)
	$K = 1$	$K = 9$	$K = 47$	$K = 5$
*CbH_**num***	93.9(1.4)	75.2(13.7)	52.6(0.8)	82.8(2.6)
	$v = 0.25$	$v = 0.01$	$v = 0.1$	$v = 0.03$
MoG	98.8(0.6)	78.5(1.3)	60.7(0.6)	92.0(0.4)
Naive Parzen	98.3(0.6)	**96.5(0.4)**	61.4(0.7)	93.0(0.8)
Parzen	**99.0(0.3)**	72.3(0.5)	59.0(0.3)	92.2(0.4)
k-Means	98.4(1.0)	84.6(3.5)	57.8(1.0)	89.1(1.6)
1-NN	98.3(0.2)	69.4(0.6)	59.0(0.9)	90.2(0.9)
k-NN	98.3(0.2)	69.4(0.6)	59.0(0.9)	90.2(0.9)
Auto-Encoder	97.3(0.5)	38.4(0.9)	56.4(0.9)	87.8(1.0)
PCA	92.6(2.4)	30.3(1.0)	54.9(0.5)	66.9(1.1)
SOM	98.3(0.5)	79.0(2.3)	59.6(0.7)	89.0(1.1)
MST_CD	98.5(0.1)	75.6(1.8)	58.0(0.9)	89.7(0.9)
k-Centers	97.3(1.0)	71.5(12.4)	53.7(4.1)	86.3(1.2)
SVDD	98.2(0.6)	70.0(0.6)	4.7(1.4)	89.4(0.8)
MPM	97.3(0.5)	69.4(0.6)	58.7(0.9)	80.2(0.5)
LPDD	97.8(0.5)	80.0(0.5)	56.4(2.6)	89.6(0.5)

MoG = Mixture of Gaussians.
PCA = Principal Component Analysis.
SOM = Self-Organizing Map.
MST_CD = Minimum Spanning Tree Class Descriptor.
MPM = Minimax Probability Machine.
LPDD = Linear Programming Dissimilarity-data Description.

MbH_1은 속성 #1에서 [3~10], 속성 #2에서 [2~10] 등으로 속성 #9까지 인터벌을 갖는다. 그 결과 MbH_1에 의해 커버되는 테스트 데이터는 총 37개이다. 그 중 33개의 클래스 레이블이 malignant 클래스이고 4개가 benign 클래스였다. 이를 바탕으로 만약 새로운 인스턴스가 MbH_1에 의해 커버되면 유방암일 확률이 33/37 = 0.89라고 할 수 있다. 그리고 그러한 분류 요인으로는 인스턴스의 Bare Nuclei(속성 #6) 값이 6~10사이에 있기 때문이라고 분류 요인을 이해할 수 있다. 또한 세 개의 *MbH*에서 marginal adhesion(속성 #4)의 인터벌은 모두 [1~10] 전체이기 때문에 이 속성은 malignant 클래스를 결정하는데 중요하지 않을 것이라는 예측이 가능하다. 이와 같이 생성된 H-RTGLs는 분류를 수행하면서 해석도 가능한 데이터의 패턴들이기 때문에 사용자는 H-RTGLs를 이용해 데이터의 특성을 분석할 수 있다.

Table 3. Examples of *MBH* for malignant class in Wisconsin breast cancer data set

Features	MbH_1	MbH_2	MbH_3
1. Clump Thickness : 1~10	3~10	2~10	2~10
2. Uniformity of Cell Size : 1~10	2~10	3~7	2~10
3. Uniformity of Cell Shape : 1~10	2~10	3~10	3~10
4. Marginal Adhesion : 1~10	1~10	1~10	1~10
5. Single Epithelial Cell Size : 1~10	2~10	2~10	2~8
6. Bare Nuclei : 1~10	6~10	7~10	4~5
7. Bland Chromatin : 1~10	1~10	3~10	2~10
8. Normal Nucleoli : 1~10	1~5	6~10	3~5
9. Mitoses : 1~10	1~4	1~5	1~3
Total number of instances covered in test data	37	29	5
True Positve(Correct)	33	28	4
False Positive(Misclassified)	4	1	1

5. 결 론

기존의 단일 분류 알고리즘들은 분류를 수행하지만 왜 그렇게 분류했는지를 알 수 없는 블랙박스(black box) 형태라는 한계를 가진다. 본 연구에서는 이러한 해석 불가능성을 해결하고 높은 분류 성능을 갖는 단일 분류 알고리즘을 위해, Hyper-rectangles를 이용한 두 가지 단일 분류 방법을 제안했다. *MbH* 기반 단일 분류기는 인터벌을 생성 및 병합한 뒤 H-RTGLs를 생성하고, *CbH* 기반 단일 분류기는 군집 생성 기법을 이용해 H-RTGLs를 생성한다. UCI repository에 있는 실험 데이터 집합을 활용하여 제안된 알고리즘들의 분류 성능을 검증하였다. 그 결과 제안된 단일 분류기는 우수한 분류 성능을 보였는데 특히, *MbH* 기반 단일 분류기는 실험한 4개의 데이터 집합 중 2개에서 가장 높은 분류 성능을 보였다. 또한 생성된 H-RTGLs는 타겟 클래스의 패턴이 되기 때문에 분류 요인에 대한 해석 능력도 제공할 수 있었다.

본 논문에서 실험한 데이터 집합은 차원이 낮고, 데이터 개수가 아주 많지는 않았다. 따라서 추후 연구로는 제안된 알고리즘을 이용해 고차원 및 사이즈가 아주 큰 데이터 집합에서 실험을 진행하여 제안된 단일 분류기의 성능을 검증할 것이다. 그리고, H-RTGL이 분류에 대한 해석에 용이하다는 점을 발전시켜 분류에 주요한 속성들을 자동적으로 선별해 내는 방법론을 연구할 예정이다. 그렇게 되면 타겟 클래스에 대한 특성을 더 잘 파악할 수 있게 될 것이고, 또한 보이지 않는 데이터에 대한 예측도 가능할 것으로 기대한다.

참고문헌

Asuncion, A. and Newman, D. (2007), UCI machine learning repository, URL http://www.ics.uci.edu/~mlearn/MLRepository.html.

Baehrens, D., Schroeter, T., Harmeling, S., Kawanabe, M., Hansen, K., and Müller, K. R. (2010), How to explain individual classification decisions, *The Journal of Machine Learning Research*, **11**, 1803- 1831.

Bosco, G. L. and Pinello, L. (2009), A fuzzy one class classifier for multi layer model, *Fuzzy Logic and Application*, Lecture Notes in Computer Science, **5571**, 124-131.

Breiman, L. (2001), Random forests, *Machine Learning*, **45**(1), 5-32.

Breiman, L., Friedman, J., Olshen, R., Stone, C., Steinberg, D., and Colla, P. (1983), *CART : Classification and regression trees*, Wadsworth : Belmont, CA, **156**.

Cortes, C. and Vapnik, V. (1995), Support-vector networks, *Machine Learning*, **20**(3), 273-297.

Désir, C., Bernard, S., Petitjean, C., and Heutte, L. (2012), A random forest based approach for one class classification in medical imaging, *Machine Learning in Medical Imaging*, Lecture Notes in Computer Science, **7588**, 250-257.

Domingos, P. and Hulten, G. (2000), Mining high-speed data streams, *Proceedings of the 2000 ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 71-80.

Hüllermeier, E. (2011), Fuzzy sets in machine learning and data mining, *Applied Soft Computing*, **11**(2), 1493-1505.

Jeong, I. K. and Choi, J. Y. (2015), One-class classification using hyper-rectangles, *Proceedings of the KORMS/KIIE/ESK/KSS 2015 Spring Conference*, Jeju, 2265-2276.

Juszczak, P., Tax, D. M. J., Pekalska, E., and Duin, R. P. W. (2009), Minimum spanning tree based one-class classifier, *Neurocomputing*, **72**(7～9), 1859-1869.

Kemmler, M., Rodner, E., Wacker, E.-S., and Denzler, J. (2013), One-class classification with gaussian processes, *Pattern Recognition*, **46**(12), 3507-3518.

Khan, J., Wei, J. S., Ringner, M., Saal, L. H., Ladanyi, M., Westermann, F., and Meltzer, P. S. (2001), Classification and diagnostic prediction of cancers using gene expression profiling and artificial neural networks, *Nature medicine*, **7**(6), 673-679.

Khan, S. and Madden, M. G. (2014), One-class classification : taxonomy of study and review of techniques, *The Knowledge Engineering Review*, **29**(3), 345-374.

Letouzey, F., Denis, F., and Gilleron, R. (2000), Learning from positive and unlabeled examples, *Proceedings of 11th International Conference on Algorithmic Learning Theory*, Sydney, Australia.

Li, C., Zhang, Y., and Li, X. (2009), OcVFDT : one-class very fast decision tree for one-class classification of data streams, *Proceedings of the 3rd International Workshop on Knowledge Discovery from Sensor Data*, 79-86.

MacQueen, J. (1967), Some methods for classification and analysis of multivariate observations, *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, **1**(14), 281-297.

Manevitz, L. and Yousef, M. (2000), Document classification on neural networks using only positive Examples, *Proceedings of 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 304-306.

Manevitz, L. and Yousef, M. (2007), One-class document classification via Neural Networks, *Neurocomputing*, **70**, 1466-1481.

Schölkopf, B., Williamson, R., Smola, A., Taylor, J. S. and Platt, J. (2000), Support vector method for novelty detection, *Advances in Neural Information Processing Systems*, **12**, 582-588.

Schmidhuber, J. (2015), Deep learning in neural networks : An overview, *Neural Networks*, **61**, 85-117.

Skabar, A. (2003), Single-class classifier learning using neural networks : an application to the prediction of mineral deposits, *Proceedings of the Second International Conference on Machine Learning and Cybernetics*, **4**, 2127-2132.

Tax, D. M. J. and Duin, R. P. W. (1999a), Data domain description using support vectors, *Proceedings of European Sysmposium on Artificial Neural Networks*, Brussels, 251-256.

Tax, D. M. J. and Duin, R. P. W. (1999b), Support vector domain description, *Pattern Recognition Letters*, **20**, 1191-1199.

Tax, D. M. J. (2001), *One-class Classification*, PhD thesis, Delft University of Technology.

Tax, D. M. J. (2010), One-class classifier results, URL http://homepage. tudelft.nl/n9d04/occ/.

Quinlan, J. R. (1993), C4.5 : Programs for Machine Learning, *Morgan Kaufmann*, California.

Utkin, L. V. (2012), Fuzzy one-class classification model using contamination neighborhoods, *Advances in Fuzzy Systems*, **22**, doi: 10.1155/2012/984325.