놈(Norm)에 따른 *k*-최근접 이웃 학습의 성능 변화

김두혁, 김찬주, 황규백
숭실대학교 컴퓨터학부
{dhkim, cjkim}@ml.ssu.ac.kr, kbhwang@ssu.ac.kr

k-Nearest Neighbor Learning with Varying Norms

Doo-Hyeok Kim, Chanju Kim, and Kyu-Baek Hwang
Division of Computing, Soongsil University

요 약

예제 기반 학습(instance-based learning) 방법 중 하나인 *k*-최근접 이웃(*k*-nearest neighbor, *k*-NN) 학습은 간단하고 예측 정확도가 비교적 높아 분류 및 회귀 문제 해결을 위한 기반 방법론으로 널리 적용되고 있다. *k*-NN 학습을 위한 알고리즘은 기본적으로 유클리드 거리 혹은 2-놈(norm)에 기반하여 학습예제들 사이의 거리를 계산한다. 본 논문에서는 유클리드 거리를 일반화한 개념인 *p*-놈의 사용이 *k*-NN 학습의 성능에 어떠한 영향을 미치는지 연구하였다. 구체적으로 합성데이터와 다수의 기계학습 벤치마크 문제 및 실제 데이터에 다양한 *p*-놈을 적용하여 그 일반화 성능을 경험적으로 조사하였다. 실험 결과, 데이터에 잡음이 많이 존재하거나 문제가 어려운 경우에 *p*의 값을 작게 하는 것이 성능을 향상시킬 수 있었다.

1. 서론

k-최근접 이웃(*k*-nearest neighbor, *k*-NN) 학습은 예제 기반 학습(instance-based learning) 방법 중 하나이다. *k*-NN 알고리즘은 주어진 학습데이터를 단순히 저장한다. 이후 주어진 문제[test instance]를 분류할 때, 저장하고 있는 학습데이터 중 가장 가까운 *k*개의 예제의 목표 속성(target attribute) 값에 기반하여 답을 결정한다. 이 학습 방법의 결과는 설정된 *k*값과 정의된 거리함수(distance function)에 따라 성능이 달라진다. 예를 들어, 데이터에 잡음이 많이 존재하는 경우, *k* 값을 크게 함으로써 일반적으로 성능을 향상시킬 수 있다[1]. 또한, 잡음이 많이 존재하는 실제 상황에서 유클리드 거리가 아닌 *p*-놈(norm)을 적용하여 *k*-NN 학습의 성능을 향상시킨 사례도 있다[2]. 본 논문에서는 *k*-NN 학습의 성능에 *p*-놈의 종류가 미치는 영향을 경험적으로 밝힌다. 문제의 특성을 조절할 수 있는 합성데이터, 다양한 기계학습의 벤치마크 데이터 및 실제 데이터마이닝 문제를 이용해서 거리함수의 변화가 주어진 상황에 따라 분류 성능에 어떤 영향을 주는지 조사한다.

2. *k*-NN 학습

2.1 놈(norm)

놈(norm)은 벡터 공간의 벡터들에 대해 길이 혹은 크기를 부여하는 함수이다. 영벡터(zero vector)의 놈은 0이며, 그 외의 모든 벡터의 놈은 양의 실수가 된다. 일반적으로 *p*–놈은 다음과 같이 정의된다. *p*가 $p \geq 1$인 실수일 때,

$$\|\mathbf{x}\|_p := \left(\sum_{i=i}^{n} |x_i|^p\right)^{\frac{1}{p}}. \quad (1)$$

여기서 **x**는 *n*차원의 벡터이며 x_i는 벡터의 *i*번째 원소이다. 특별히 $p = 2$인 경우에는 유클리드 거리에 해당하며, $p = 1$인 경우에는 맨하탄 거리에 해당한다. 수식 (1)에서 *p*의 값이 커질수록 한 원소의 크기가 더 많이 반영되며, *p*의 값이 작아질수록 한 원소의 크기가 놈의 값에 상대적으로 덜 반영된다. *p*의 값이 1보다 작을 때는 삼각부등관계(triangle inequality)를 만족하지는 않지만, 벡터의 특정 원소의 절대값이 잡음으로 인해 커지는 경우 이의 영향을 줄일 수 있다. 본 논문에서는 *p*의 값이 1보다 작은 경우에 대해서도 실험을 하였으며, 많은 문제에서 *p*의 값이 1보다 작을 때 가장 높은 정확도를 얻을 수 있었다.

2.2 *k*-NN 학습 방법

k-최근접 이웃 학습 기법은 모든 예제들이 *n*차원의 공간 R^n의 한 점에 대응된다고 가정한다. 이때 임의의 예제 **x**는 다음과 같이 표현할 수 있다.

$$\langle a_1(\mathbf{x}), a_2(\mathbf{x}), \cdots a_n(\mathbf{x})\rangle. \quad (2)$$

여기서 $a_r(\mathbf{x})$ 는 예제 **x**의 *r*번째 속성을 나타낸다. 그러면 두 예제 $\mathbf{x}_i$와 $\mathbf{x}_j$ 사이의 거리 $d(\mathbf{x}_i, \mathbf{x}_j)$는 다음과 같이 *p*-놈으로 정의할 수 있다.

$$d(\mathbf{x}_i, \mathbf{x}_j) := \|\mathbf{x}_i - \mathbf{x}_j\|_p = \left(\sum_{r=1}^{n} |a_r(\mathbf{x}_i) - a_r(\mathbf{x}_j)|^p\right)^{\frac{1}{p}}. \quad (3)$$

3. 실험

3.1 합성데이터

합성데이터에 대한 실험은 아래와 같은 과정으로 진행되었다.

표 1. 합성데이터 생성 및 실험 과정

1. 2차원 평면에 두 가지 클래스를 가지는 임의의 필드를 생성한다. (그림 1, 2 참조) 분류 문제의 복잡도는 임의의 반평면을 생성하는 XOR 연산을 복잡도만큼 반복 수행함으로써 설정한다[3].

2. 두 클래스에 해당하는 예제를 필드의 임의의 위치에 생성한다. 생성된 예제는 자신이 위치하는 필드의 속성에 따른 클래스 값을 가진다. 단, 잡음에 해당하는 경우는 반대의 클래스 값을 가진다.

3. *k*-NN 학습 알고리즘을 이용해 예제를 분류하고, 분류된 예제에 의한 새로운 필드를 생성한다.

4. 처음 만들어진 필드와 학습된 필드를 비교하여서 *k*-NN 학습의 오류율(error rate)를 측정한다.

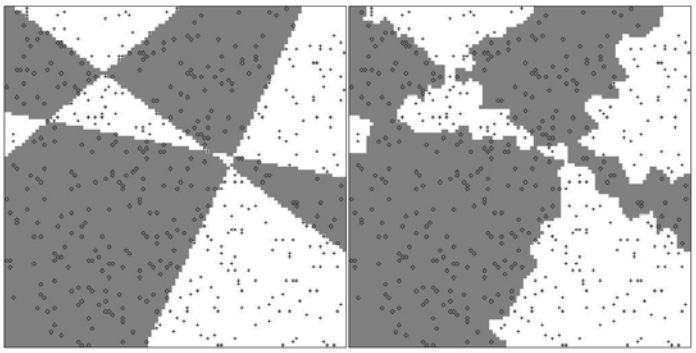

그림 1. 복잡도: 4, 표본크기: 500, 잡음비율: 0%, *p* 값: 2인 경우의 문제(왼쪽 그림) 및 학습 결과(오른쪽 그림)

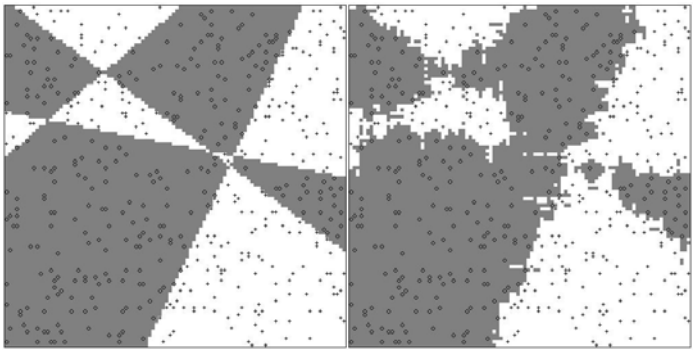

그림 2. 복잡도: 4, 표본크기: 500, 잡음비율: 0%, *p* 값: 0.5인 경우의 문제(왼쪽 그림) 및 학습 결과(오른쪽 그림)

합성데이터에 대한 실험은 다양한 복잡도, 학습데이터 크기, 잡음의 비율 및 *p* 값에 대해 행해졌다. 합성데이터의 경우, 원래 문제가 유클리드 공간에 정의되었기 때문에 *p* 의 값이 2인 경우에 가장 높은 학습 성능을 보일 것으로 예상할 수 있다. 실험 결과는 예상과 비슷하였으나, 그 양상은 학습데이터에 존재하는 잡음의 비율과 밀접한 관련이 있었다.

그림 3과 4는 각각 학습데이터의 크기가 1000이고 문제의 복잡도가 4 및 16인 경우의 놈에 따른 학습성능 변화 추이를 각기 다른 잡음비율(0% 및 30%)에 대해 보여준다. 그림에서 표준오류율은 각 놈에서의 오류율(error rate)과 전체 평균 오류율과의 차이를 나타낸다.[1] 각 놈에서의 오류율은 독립적으로 데이터 생성을 10번 반복하여 그 평균을 취하였다.

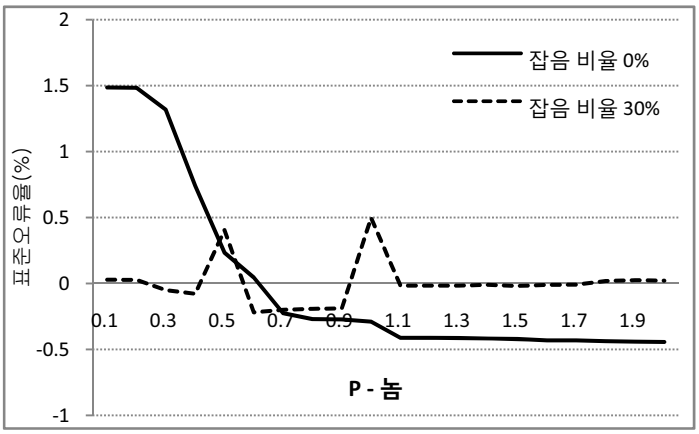

그림 3. *p* 값에 따른 오류율 변화 (복잡도: 4, 표본크기: 1000)

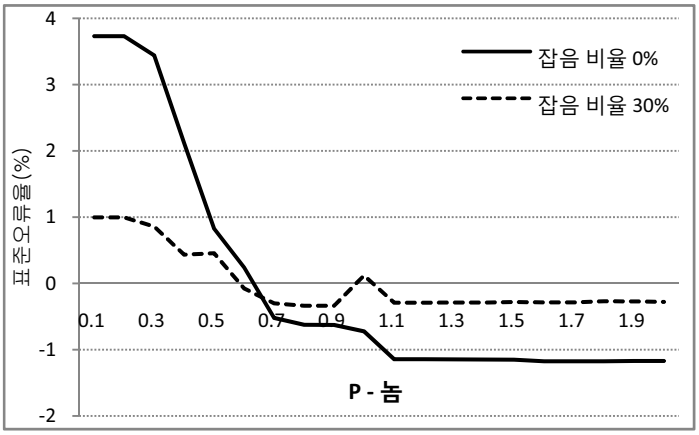

그림 4. *p* 값에 따른 오류율 변화 (복잡도: 4, 표본크기: 1000)

실험 결과 데이터에 잡음이 없을 때는 *p* 값이 0.1에서 2.0에 가까워질수록 학습성능이 좋아지는 것을 알 수 있다. 하지만, 데이터에 잡음이 30% 존재하는 경우는 *p*의 값이 2에 가까워질수록 성능이 크게 개선되지 않으며, 최적의 학습성능이 유클리드 거리가 아닌 경우에 얻어지기도 한다. 이는 잡음에 해당하는 데이터 예제의 경우 유클리드 거리로 계산했을 때 올바른 클래스 레이블을 얻을 수 없기 때문으로 생각된다. 정리하자면, 잡음이 없는 데이터의 경우 원래 유클리드 공간 상의 문제이므로, *p*의 값이 2에 가까워질수록 성능이 좋아지나, 데이터의 잡음은 이러한 현상을 방해하는 것으로 볼 수 있다. 또한, 잡음의 정도에 따라

1 이는 잡음비율의 정도에 따른 오류율의 차이를 고려하여 비교하기 위해서이다.

우연히 p의 값이 작은 경우에 더 높은 학습성능을 내기도 하는 것으로 여겨진다. 문제의 복잡도나 학습데이터의 크기가 다른 경우에도 대체적으로 비슷한 결과를 보였다. 다음 절에서는 기계학습 벤치마크 데이터에 대해 실험한 결과를 제시한다.

3.2 기계학습 벤치마크 데이터

실험에 사용된 벤치마크 데이터는 다음과 같다[4].

Iris Plants Database (Iris)

아이리스 식물의 타입을 나타내는 3개의 클래스(Iris Setosa, Iris Versicolour, Iris Virginica)가 있고 꽃받침의 길이, 넓이, 꽃잎의 길이와 넓이와 같은 아이리스의 속성이 있다. 각 클래스 별로 50개씩 총 150개의 학습 예제가 있다.

Wine Recognition Data (Wine)

서로 다른 종이지만 이탈리아의 같은 지역에서 재배된 와인의 화학적인 분석 값으로, 이 분석 값들은 3종류의 와인에서 얻어진 13개 성분의 양이다. 총 178개의 학습 예제가 있다.

Wisconsin Breast Cancer Database (WBCD)

유방암 진료 사례를 바탕으로 제작한 것으로 699개의 학습 예제가 있다. 각 예제는 유방암에 대한 10개의 특성과 양성(malignant) 및 음성(benign) 여부를 가지고 있다. 실험에서는 missing attribute value가 있는 16개의 예제는 제외하였다.

Wisconsin Prognostic Breast Cancer (WPBC)

32개의 특성과 유방암의 재발 여부를 속성으로 가지고 있다. 총 198개의 학습 예제 중 missing attribute value가 있는 4개는 제외하였다.

벤치마크 데이터들의 특징을 정리하면 표 2 와 같다.

표 2. 실험에 사용된 벤치마크 데이터

학습 데이터	속성값	속성 개수	클래스 개수	학습 예제 개수
Iris	실수.	4	3	150
Wine	실수	13	3	178
WBCD	실수	10	2	683
WPBC	실수	32	2	194

각 데이터에 대하여 p의 값을 0.1부터 2까지 0.1씩 변화시켜가며 k-NN 학습을 적용하였다. 성능의 평가는 10-folds cross validation을 이용하였다. 벤치마크 데이터의 경우 합성데이터와는 달리 잡음의 정도를 인위적으로 조절할 수 없다. 본 논문에서는 naïve Bayes 분류기와 TAN(tree augmented naïve Bayes) 분류기를 적용하여 문제의 복잡도에 대한 추정을 하였다. 베이지안망(Bayesian network) 분류기의 경우 허용되는 부모 노드의 최대 개수에 따라 그 표현력이 결정된다[5]. TAN 분류기의 경우 허용되는 부모의 최대 개수는 2이며 naïve Bayes의 경우는 1이다. 이에 기반하여 TAN 분류기가 naïve Bayes 분류기보다 성능이 높게 나오는 문제를 더 복잡한 문제로 추정하였다. 실험에서 naïve Bayes 및 TAN 분류기는 WEKA(http://www.cs.waikato.ac.nz/ml/weka/)를 활용하였다.

그림 5 – 8은 각각 Iris, Wine, WBCD 및 WPBC 데이터에 대한 실험 결과를 보이고 있다.

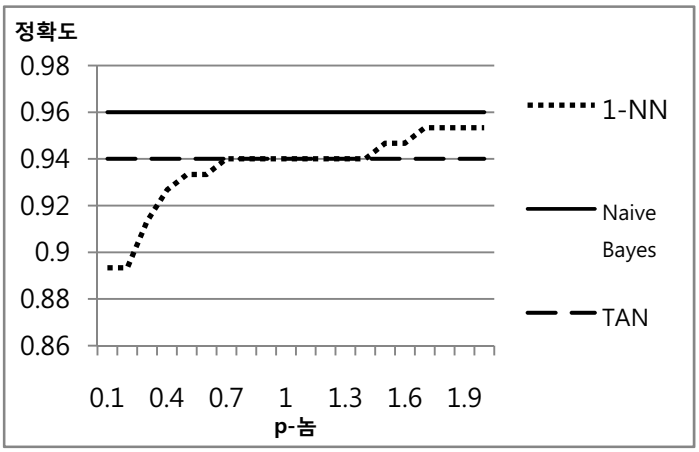

그림 5. p 값에 따른 정확도 변화 (Iris 데이터)

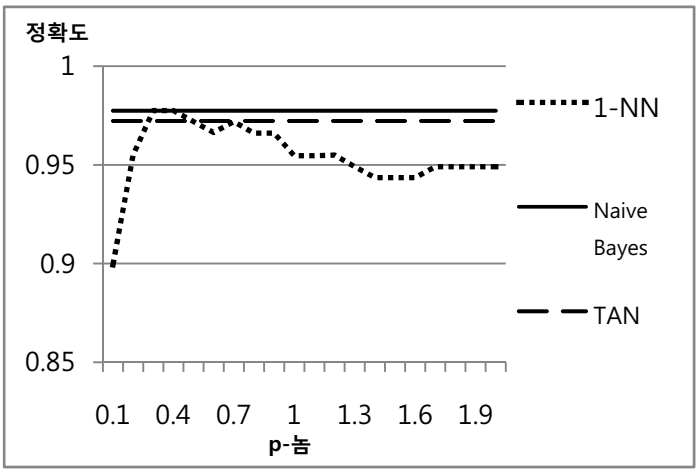

그림 6. p 값에 따른 정확도 변화 (Wine 데이터)

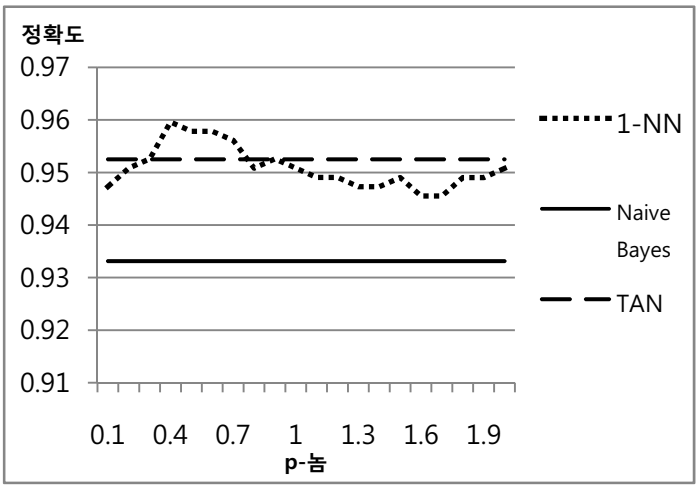

그림 7. p 값에 따른 정확도 변화 (WBCD 데이터)

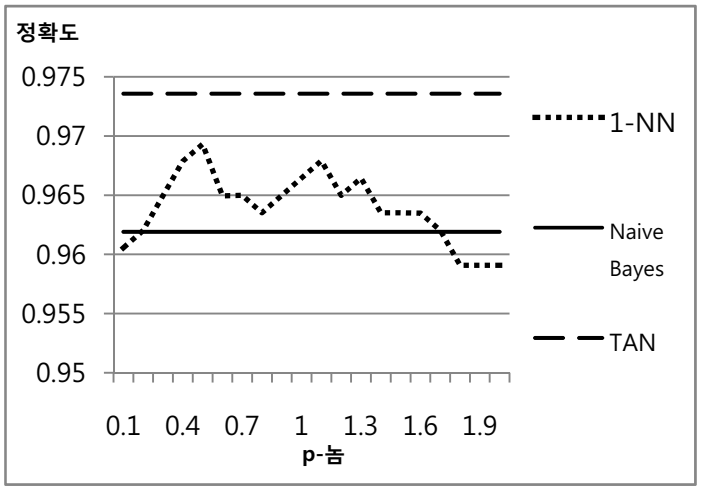

그림 8. p 값에 따른 정확도 변화 (WPBC 데이터)

실험 결과에 따른 4개의 벤치마크 문제의 복잡도는 다음과 같이 추정할 수 있다. Iris 데이터의 경우 naïve Bayes 분류기의 정확도가 TAN 분류기에 비해 높게 나왔다. Wine 데이터의 경우 naïve Bayes 분류기의 성능이 TAN 분류기에 비해 높다. 나머지 WBCD 및 WPBC 데이터의 경우 TAN 분류기의 성능이 더 높다. 정리하면, 문제의 복잡도는 WBCD, WPBC 문제가 Iris나 Wine 데이터에 비해 더 높다고 추정하였다. 물론 이는 추정이므로 실제 복잡도와는 다를 수도 있음을 밝혀둔다.

각 문제에 대한 놈에 따른 성능 변화 추이는 다음과 같다. Iris 문제의 경우 유클리드 거리에 기반하는 경우의 정확도가 가장 높다. 그 외 나머지 데이터의 경우 유클리드 거리에서 가장 성능이 높지 않으며, 특이하게도 p의 값이 0.4 근처일 때 가장 높은 정확도를 보이고 있다. 유클리드 거리에서 높은 성능을 보이지 않는 3개의 벤치마크 문제 중 둘은 복잡도가 높은 편으로 추정되었으며, 나머지 하나는 그렇지 않았다. 유클리드 거리에서 가장 정확한 학습 결과를 보인 Iris 문제는 문제의 복잡도가 상대적으로 낮은 것으로 추정되었다. 조심스러운 결론을 내리자면, 문제의 복잡도가 높은 경우에 유클리드 거리가 아닌 다른 p-놈을 사용하여 k-NN 학습 결과의 성능을 높일 수 있다고 볼 수 있다. 다음 절에서는 실제 데이터마이닝 문제에 같은 실험을 적용하고 그 결과를 정리한다.

3.3 데이터마이닝 문제

IEEE ICDM2007 학회의 Data Mining Contest[2]에 출제되었던 데이터로 101개의 AP(access point)에서 얻어진 라디오 신호 값과 위치 정보가 들어있다. 문제는 AP에서 얻어진 신호의 양상에 기반하여 현재의 위치를 예측하는 것이다. 총 505개의 학습 예제와 2137개의 테스트데이터로 이루어져 있다. 학습데이터는 극히 희박(sparse)하며 잡음을 많이 포함하고 있다. 그림 9는 놈에 따른 분류 성능 추이를 보이고 있다.

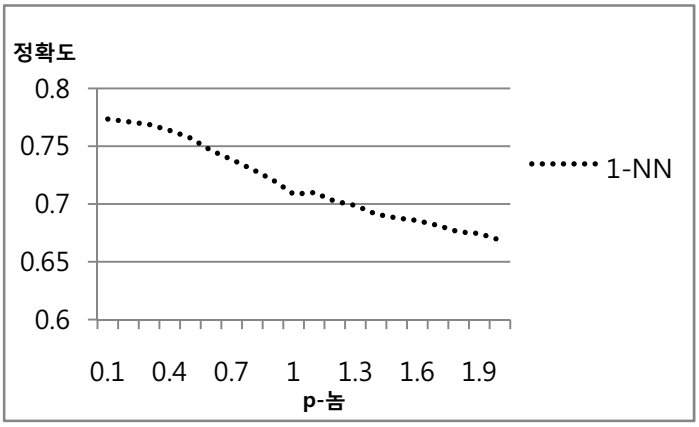

그림 9. p 값에 따른 정확도 변화 (실제 데이터마이닝 문제)

이 문제의 경우 유클리드 거리에서 가장 성능이 낮으며 p의 값이 작을수록 높은 정확도를 보이고 있다. 이는 실제 문제가 상당히 복잡하며 잡음을 다수 포함하고 있기 때문으로 생각된다.

4. 결론

본 논문에서는 기계학습의 기반 방법론으로 널리 사용되는 k-최근접 이웃 학습의 거리함수에 따른 성능 변화를 문제의 복잡도 및 잡음의 정도의 관점에서 관찰하였다. 합성데이터 및 기계학습 벤치마크 데이터에 대한 실험 결과, 데이터에 잡음이 많이 포함되어 있거나 문제가 복잡할수록 낮은 p 값에서 높은 정확도를 보이는 경향을 관찰할 수 있었다. 이는 p 값이 낮은 경우 잡음으로 인해 생기는 큰 차이를 감쇄할 수 있기 때문으로 생각된다. 실제 데이터마이닝 문제에 대한 실험 결과, p의 값이 낮을수록 높은 정확도를 보였으며, 이는 합성데이터 및 벤치마크 데이터에 대한 실험결과에 부합한다.

감사의 글

본 연구는 21세기 프론티어 연구개발사업의 일환으로 추진되고 있는 지식경제부의 유비쿼터스컴퓨팅 및 네트워크 원천기반기술 개발사업의 지원에 의한 것임

참고문헌

[1] I.H. Witten and E. Frank, *Data Mining – Practical Machine Learning Tools and Techniques*, Morgan Kaufmann Publishers, 2000.

[2] Q. Yang, J.J. Pan, and V.W. Zheng, Estimating location using Wi-Fi, *IEEE Intelligent Systems*, vol. 23, no. 1, pp. 8-13, 2008.

[3] J. Zhu, http://www-2.cs.cmu.edu/~zhuxj/courseproject/knndemo/KNN.html, Carnegie Mellon University, 2000.

[4] A. Asuncion and D.J. Newman, UCI Machine Learning Repository[http://www.ics.uci.edu/~mlearn/MLRepository.ht

ml], Irvine, CA: University of California, School of Information and Computer Science, 2007.

[5] C.X. Ling and H. Zhang, The representational power of discrete Bayesian networks, *Journal of Machine Learning Research*, vol. 3, pp. 709-721, 2002.