JDCS
디지털콘텐츠학회논문지
Journal of Digital Contents Society
Vol. 21, No. 3, pp. 587-592, Mar. 2020

Check for updates

개인화 추천을 위한 비모형 차원 축소 기법

유 찬 우[1] · 김 희 천[2*]
[1]라인플러스
[2]한국방송통신대학교 컴퓨터과학과 교수

Dimensionality Reduction Method without Model for Personalized Recommendation

Chan-Woo Yoo[1] · Hee-Chern Kim[2*]
[1]Line Plus Corporation
[2]Professor, Department of Computer Science, Korea National Open University, Seoul, Korea

[요 약]

개인화 추천을 위한 머신러닝 모델에 사용될 사용자 또는 품목에 대한 임베딩을 만들어 내기 위해서 차원 축소 기법이 널리 사용된다. 그러나 특이값 분해와 같은 모형을 사용하는 차원 축소 기법은 그 성질이 잘 알려져 있다는 장점에도 불구하고 산업계에서 발생하는 빅데이터에는 수행시간이나 메모리 측면에서 적용하기 어렵다는 단점이 있다. 이 경우에는 모형을 사용하지 않는 랜덤 투영과 같은 보다 빠르고 자원을 소모하지 않는 기법이 사용된다. 본 논문에서는 랜덤 투영과 같이 모형을 사용하지 않아 가벼운 방법이지만 랜덤 투영보다 본래 데이터 포인트 사이의 코사인 유사도를 더 잘 보존할 수 있는 방법을 제안하였다. 개인화 추천 분야의 공개된 데이터셋에 대해서 실험을 한 결과 제안한 방법이 모든 차원에서 랜덤 투영보다 코사인 유사도를 더 잘 보존하고, 수행 시간 면에서도 우위에 있는 것으로 나타났다. 또한 모형을 생성하는 특이값 분해와도 비교를 수행하였는데, 제안한 방법이 일정 이상의 차원에서는 특이값 분해와 코사인 유사도 보존 측면에서 차이가 없었고, 수행 시간에서는 우위를 보였다.

[Abstract]

Dimensionality Reduction methods are used to generate embeddings of users or items in machine learning models for personalized recommendation. Methods like SVD(singular valued decomposition) are hard to apply when big data is given in industrial fields because of their time and space complexity, although their mathematical properties are well known. In such a case, dimensionality reduction methods without models like random projection, which is fast and requires small space, are used. In this paper, another fast dimensionality reduction method without a model is proposed, which is better than Random Projection in cosine similarity preservation perspective. Experiments are performed with open datasets in personalized recommendation domain. The results shows the proposed method is better than random projection in cosine similarity preservation and execution time in all dimensions. A comparison between the proposed method and SVD, which creates a mkodel, is also performed. The proposed method appears that it achieves the similar degrees of cosine similarity preservation to SVD when the dimension is higher than some level and better execution time.

색인어 : 코사인 유사도, 머신러닝, 개인화 추천, 특이값 분해, 변수 선택
Key word : Cosine Similarity, Machine Learning, Personalized Recommendation, Singular Value Decomposition, Variable Selection

http://dx.doi.org/10.9728/dcs.2020.21.3.587

Received 06 February 2020; **Revised** 15 March 2020
Accepted 25 March 2020

***Corresponding Author; Hee-Chern Kim**

Tel: +82-2-3668-4657
E-mail: hckim@knou.ac.kr

Ⅰ. 서 론

개인화 추천을 위한 머신러닝 모델의 훈련 데이터는 보통 매우 큰 사용자(user)-품목(item) 차원을 가지고 있기 때문에, 계획 행렬(design matrix)의 행이나 열을 그대로 사용하기는 어렵다. 따라서 행이나 열의 차원을 축소하여 낮은 계수의 임베딩(low rank embedding)을 만들어내야 하는 상황이 많고, 이를 위해 차원 축소 기법이 사용된다.

개인화 추천 머신러닝에서 사용되는 차원 축소 기법 중 가장 널리 알려진 것은 특이값 분해(singular value decomposition)[1]라고 할 수 있다. 특이값 분해를 이용한 차원 축소는 주어진 계획 행렬 A에 대한 낮은 계수 근사 행렬(low rank approximation matrix)을 $A_r = \sum_{i=1}^{r} \sigma_i u_i v_i^T$와 같이 가장 큰 특이값에 해당하는 특이벡터들의 곱으로 나타냄으로써 이루어진다. 특이값 분해는 원래 행렬과의 놈(norm) 차이를 최소화하는 근사 행렬을 구하게 해 준다는 점에서 본래의 정보를 잘 보존하는 측면이 있지만, 행렬의 차원이 수십만 이상으로 커지게 되면 로컬 머신에서 실행이 어렵게 되며, 병렬화가 어려운 알고리즘이라 빅데이터 클러스터가 있어도 큰 도움이 되지 않는다는 단점이 있다. 산업계의 추천 문제에서는 등장하는 데이터의 차원이 수천만에서 1억을 넘어가는 일도 많기 때문에 수학적으로 여러 가지 성질이 검증되어 있다는 장점에도 불구하고 특이값 분해를 적용하기 어려운 경우가 많다.

데이터가 매우 큰 상황에서는 모형을 생성하지 않는 차원 축소 기법을 적용할 수 있는데, 대표적으로 랜덤 투영(random projection)[2] 방법이 있으나, 특이값 분해에 비해 본래 데이터의 정보를 잘 보존하지 못한다는 단점이 있다.

또한 개인화 추천 분야의 데이터를 차원 축소할 때 중요한 것은 데이터 포인트 사이의 코사인 유사도[3] 순위가 보존되는 정도라는 것을 강조할 필요가 있다. 차원 축소 후에도 가까운 항목을 가깝게, 먼 항목을 멀게 유지하되, 상대적인 유사도의 순위만 바뀌지 않는다면, 차원 축소 후 근접 이웃 방법을 적용할 때 정확도의 손실이 일어나지 않기 때문에, 차원 축소 전후에 유사도 순위가 유지되는 비율을 가지고 차원 축소 방법의 성능을 측정할 필요가 있는데, 기존 연구 중에서 이와 같은 연구를 찾기 어려웠다.

본 논문에서는 모형을 생성하지 않는 차원 축소 방법 중, 랜덤 투영 방법보다 차원 축소 후 유사도 순위를 더 잘 유지할 수 있는 방법을 제시하고, 두 방법을 유사도 손실과 수행 시간 측면에서 비교하고자 한다. 또한 모형을 생성하는 특이값 분해 방법과도 성능을 비교하여 실용적인 차원 이상의 임베딩에서는 정확도에 차이가 없음을 보이고자 한다.

본 논문은 2장에서 랜덤 투영 방법과 특이값 분해에 의한 차원 축소에 대해 설명하고, 3장에서 랜덤 투영 방법 및 특이값 분해와 본 논문에서 제시하는 차원 축소 방법을 코사인 유사도 보존과 수행 시간 측면에서 비교하며, 4장에서 결론을 맺는다.

Ⅱ. 관련 개념

모형을 사용하는 차원 축소 기법으로는 특이값 분해나 ALS(alternating least square)[4]와 같은 방법들이 있다. 하지만 모형을 훈련하는데 많은 시간이 걸리는 데다가, 새로운 데이터가 들어오게 되면 계속해서 모형을 갱신해 나가야 하고, 빅데이터 환경에서 계획 행렬이 매우 클 때는 모형을 적용하는 것 자체가 어려울 수 있다. 이 경우에는 모형을 사용하지 않는 차원 축소 기법이 필요한데, 이를 위해 랜덤 투영 방법이 널리 사용된다[5], [6]. 이 방법은 임의의 기저(basis) 벡터를 생성한 뒤 생성된 벡터와 직교하는 벡터를 반복적으로 생성한다. 그 다음 생성된 벡터들을 기저로 해서 원래의 데이터를 투영하게 된다. 본래는 이와 같은 방식으로 사용되지만 고차원의 랜덤 벡터는 대체로 서로 직교하는 경향이 있기 때문에[7] 기존 벡터와 직교하는 벡터를 구하지 않고 모든 기저 벡터를 한꺼번에 랜덤으로 생성하여 적용하는 방법이 사용되기도 한다.

랜덤 투영 방법은 존슨-린든스트라우스 보조정리[8], [9]에 기반을 두고 있는데, 존슨-린든스트라우스 보조정리는 $0 < \epsilon < 1$이고, R^N에 속하는 m개의 점들의 집합 X가 주어졌을 때, $n > 8\ln(m)/\epsilon^2$ 을 만족시킨다면, X에 속하는 모든 (u, v)에 대해 식 (1)을 만족하는 선형 변환 $f : R^N \rightarrow R^n$가 존재한다는 것이고, 랜덤 투영 방법은 이와 같은 조건을 만족하는 선형 변환이라고 할 수 있다.

$$(1-\epsilon) \parallel u-v \parallel^2 \leq \parallel f(u)-f(v) \parallel^2 \leq (1+\epsilon) \parallel u-v \parallel^2 \quad (1)$$

개인화 추천을 위한 머신러닝 모델 생성 시, 특이값 분해를 이용한 차원 축소는 다음과 같은 방식으로 이루어진다. 행이 사용자, 열이 품목을 나타내며 원소는 사용자의 품목에 대한 평가를 나타내는 계획 행렬 A가 아래 수식과 같을 때, 데이터 포인트(예: 사용자)가 식 (2)와 같이 a_i^T와 같은 행 벡터로 나타난다고 가정하자.

$$A = \begin{bmatrix} - a_1^T - \\ - a_2^T - \\ \cdots \\ - a_m^T - \end{bmatrix} \quad (2)$$

$A \in R^{m \times n}$, $m > n$일 때, 특이값 분해는 $A = U \Sigma V^T$ 와 같이 나타나고, Σ는 특이값들의 대각행렬, U와 V는 직교 행렬이 될 수 있다. 차원을 k로 축소하는 경우, 가장 큰 k개의 특이값에 대응되는 우 특이벡터(right singular vector)를 기저로 $a_i V_k^T$과 같이 데이터 포인트를 투영하게 된다. 만약 $k = n$인 경우에는 V가 직교행렬이기 때문에 데이터 포인트의 길이 변화 없이 회전만을 시켜서 식 (3)과 같이 코사

인 유사도에 변화를 가져오지 않지만, $k < n$과 같이 차원을 축소시키는 경우에는 코사인 유사도에 변화가 생기게 되고, 그에 따라 코사인 유사도 순위에도 변동이 일어나게 된다.

$$\frac{a_i^T V V^T a_j}{\| a_i^T V \| \| a_j^T V \|} = \frac{a_i^T a_j}{\| a_i \| \| a_j \|} \quad (3)$$

Ⅲ. 모형을 사용하지 않는 차원 축소 기법

본 논문에서는 개인화 추천에서 모형을 사용하지 않는 차원 축소 방법 중 랜덤 투영 방법이 코사인 유사도 보존 측면에서 가장 좋은 성능을 내는지 검토하고자 하였다. 특이값 분해가 가장 분산이 커지는 방향의 기저 벡터로의 투영이기 때문에 같은 차원으로 차원 축소를 할 때 상대적으로 많은 정보를 담게 되지만 모형을 만들어야 해서 큰 데이터에 적용이 어렵다는 단점이 있었다. 그렇다면 차원을 r로 줄이기 위해서 계획 행렬의 열 중 가장 분산이 큰 r개의 열만을 선택하는 식으로 차원을 줄인다면, 상대적으로 모형 없는 간단한 방법으로 가장 많은 정보를 담을 수 있지 않을까라는 발상을 하게 되었고, 랜덤 투영 방법과 실험을 통해 그 성능을 비교하고자 하였다. 계획 행렬 A를 $A \in R^{m \times n}$과 같은 행렬이라고 하고, 식 (4)와 같이 c_i는 A의 i번째 열 벡터를, c_{ij}는 A의 i번째 열, j번째 행의 원소라고 하면 가장 분산이 큰 r개 열의 인덱스를 선택하는 것은 식 (5)와 같이 이루어진다.

$$A = \begin{bmatrix} | & | & & | \\ c_1 & c_2 & \vdots & c_n \\ | & | & & | \end{bmatrix} \quad (4)$$

$$\arg\max_X \sum_{i \in X} Var(c_i) \ \ s.t. \ \ | X | = r \quad (5)$$

$$where \ Var(c_i) = \frac{\sum_{j=1}^{m} c_{ij}^2}{m} - \left(\frac{\sum_{j=1}^{m} c_{ij}}{m} \right)^2$$

차원 축소 방법의 성능을 비교하는데 사용한 지표는 상대 순위 정확도(relative ranking precision)인데, 이는 계획 행렬 $A \in R^{m \times n}$의 데이터 포인트들이 a_i^T와 같은 행 벡터로 식 (2)와 같이 표현되고 $I()$는 지시 함수(indicator function)를, $sgn()$은 값의 부호를 나타낸다고 할 때, 식 (6)과 같이 정의된다.

상대 순위 정확도의 의미는 차원 축소 전과 후에 상대적인 코사인 유사도의 순위가 바뀌지 않고 보존되는 삼중항(triplet)의 수가 전체 삼중항의 수에서 차지하는 비율이라고 할 수 있다. 코사인 유사도의 순위가 바뀌지 않을수록 가까운 이웃을 보다 정확하게 줄 수 있게 되기 때문에, 보다 정확한 추천이 가능해진다.

실험에는 표 1과 같은 공개 데이터셋이 사용되었다. 사용된 데이터셋은 사용자가 품목에 대해 평가 점수를 매긴 것으로 개인화 추천 머신러닝 분야에서 벤치마크를 위해 많이 사용되는 데이터셋들이라고 할 수 있다.

실험은 랜덤 투영 방법과 식 (5)와 같이 분산에 따른 변수 선택 기법 두 가지의 상대 순위 정확도를, 차원 축소를 적용할 차원의 수를 달리해 가면서 측정하는 방식으로 진행되었다. 랜덤 투영 방법의 적용을 위해서는 sklearn의 random_projection 라이브러리[14]를 사용하였다. 실험 결과는 그림 1과 같다. x 축은 축소한 차원의 크기를, y 축은 상대 순위 정확도를 나타내며, 붉은색 ×는 분산이 큰 순으로 계획 행렬의 열을 선택하여 차원을 줄인 결과의 상대 순위 정확도를, 푸른색 점은 랜덤 투영 방법을 사용해 차원을 줄인 결과의 상대 순위 정확도를 나타낸다.

표 1. 실험에 사용된 공개 데이터셋
Table 1. Datasets used in experiments

Dataset	User	Item	Review	Rating Range
Movielens[10]	943	1650	80000	[1, 5]
Boardgamegeek[11]	2000	16034	495845	[0, 10]
Jester[12]	2000	140	246573	[-10, 10]
Datafinti[13]	408	207	2605	[1, 5]

표 2. 랜덤 투영 방법과 변수 선택 기법의 상대 순위 정확도에 대한 대응 표본 T 검정 결과
Table 2. Paired t-test compares relative ranking precision between random projection and feature selection method

Dataset	t-statistic	p-value
Movielens	4.502203	0.001139
Boardgamegeek	5.012183	0.000237
Jester	8.936539	4.4665e-05
Datafinti	3.557947	0.009244

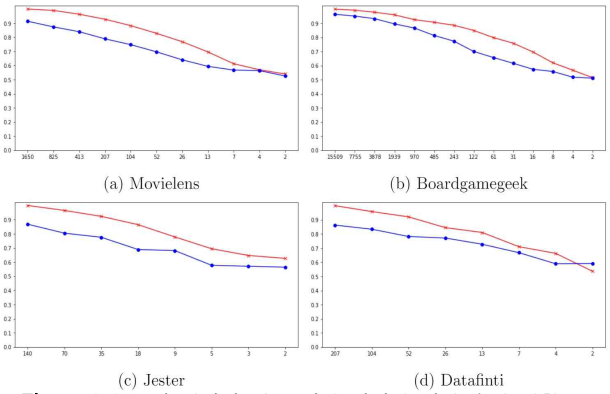

그림 1. 랜덤 투영 방법과 변수 선택 기법의 상대 순위 정확도 비교
Fig. 1. Comparisons of relative ranking precision between random projection and feature selection method

표 3. 랜덤 투영 방법과 변수 선택 기법의 수행시간 비교 (ms)
Table 3. Comparisons of execution time between random projection and feature selection method (ms)

Dataset	Random Projection	Feature Selection
Movielens	0:00:05.221909	0:00:05.291127
Boardgamegeek	0:01:27.717332	0:00:55.108768
Jester	0:00:06.647170	0:00:06.832270
Datafinti	0:00:01.636178	0:00:01.412616

$$\frac{\sum_{(a_i^T, a_i^T, a_i^T) \in S} I(equality_0 = equality_1)}{N} \quad (6)$$

$$where$$

$$equality_0 = sgn\left(\frac{a_i^T a_j}{\| a_i \| \| a_j \|} - \frac{a_i^T a_k}{\| a_i \| \| a_k \|} \right)$$

$$equality_1 = sgn\left(\frac{a_i^T V_r V_r^T a_j}{\| a_i^T V_r \| \| a_j^T V_r \|} - \frac{a_i^T V_r V_r^T a_k}{\| a_i^T V_r \| \| a_k^T V_r \|} \right)$$

거의 모든 차원에서 식 (5)를 이용한 변수 선택 기법이 같거나 더 높은 상대 순위 정확도를 보여주었다. 각각의 데이터셋에 대해서 차원의 크기 별로 랜덤 투영 방법을 적용한 상대 순위 정확도와 식 (5)의 방법을 적용한 상대 순위 정확도 값의 대응 표본 T 검정(paired t-test)을 실행한 결과는 표 2와 같다. 모든 유의확률이 1% 미만으로, 두 방법의 상대 순위 정확도 사이에는 유의미한 차이가 있다고 결론내릴 수 있다.

그림 1에 나타난 값들을 계산하는데 소요된 두 방법의 수행 시간을 표 3과 같이 비교하였다. Movielens, Jester, Datafinti 데이터셋에서는 시간이 거의 차이가 나지 않았고, Boardgamegeek과 같이 차원이 큰 데이터에서는 변수 선택 기법의 수행 시간이 더 적은 것으로 나타났다. 따라서 상대 순위 정확도와 수행 시간 두 가지 면에서 변수 선택 기법이 랜덤 투영 방법에 비해 더 낫다고 결론을 내릴 수 있다.

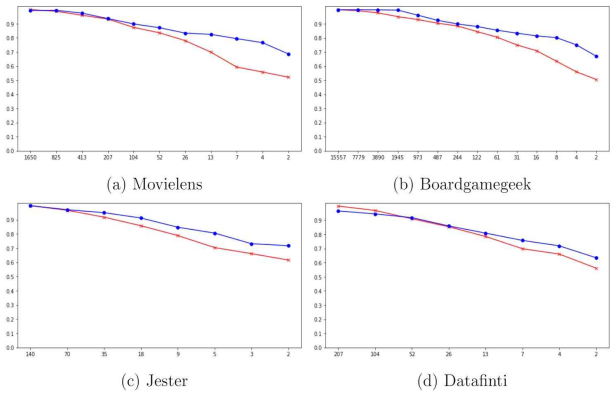

그림 2. 특이값 분해와 변수 선택 기법의 상대 순위 정확도 비교
Fig. 2. Comparisons of relative ranking precision between singular value decomposition and feature selection method

변수 선택 기법이 특이값 분해에 비해서는 어느 정도의 상대 순위 정확도를 가지는지 그림 2를 통해 비교해 보았다. 높은 차원에서는 특이값 분해와 변수 선택이 유사한 상대 순위 정확도를 보이지만, 차원이 매우 작아지게 되면 특이값 분해가 보다 높은 정확도를 보이는 것을 알 수 있다.

특이값 분해와 변수 선택 기법의 상대 순위 정확도에 대한 대응 표본 T 검정 결과는 표 4와 같다. 유의수준 1%에서 Movielens와 Datafinti의 경우에는 방법간에 유의미한 차이가 있음이 기각되고, Boardgamegeek과 Jester의 경우에는 유의미한 차이가 있음을 볼 수 있다. 그러나 보통 기존 연구나 산업계에서 사용되는 임베딩 차원이 100 이상이라고 볼 때[15]-[17], 100 차원 이상에서는 특이값 분해에 비해 변수 선택 기법의 정확도가 크게 떨어지지 않기 때문에, 아래에서 언급할 성능면의 장점을 고려한다면 특이값 분해 대신 사용되어도 무리가 없다고 보인다.

그림 1에 나타난 값들을 특이값 분해와 변수 선택 기법으로 계산하는데 소요된 수행시간을 비교한 결과는 표 5와 같다. 상대적으로 작은 차원을 가진 Movielens, Jester, Datafinti 데이터셋에서는 큰 차이가 없지만 차원이 큰 Boardgamegeek 데이터에서는 특이값 분해의 수행 시간이 확연히 큰 것을 볼 수 있다.

빅데이터 환경에서 개인화 추천을 위한 임베딩 생성을 위해 특이값 분해를 적용하는 것은 수행시간보다도 메모리적인 면에서 실제적인 적용이 어려울 수 있다. 이 경우에 사용될 수 있는 것은 랜덤 투영이나 본 논문에서 제시한 분산에 의한 변수 선택 기법인데, 위의 실험을 통해 변수 선택 기법이 랜덤 투영 방법보다 상대 순위 정확도와 수행 시간 면에서 모두 우위를 가지고 있음을 알 수 있다. 원본 데이터의 차원이 매우 커지는 상황이 되면, 특이값 분해는 많은 자원을 필요로해 적용이 어려워지게 되고, 랜덤 투영 방법도 차원이 큰 데이터 벡터와 기저 벡터의 내적 연산을 수행하여야 하므로 필요로 하는 자원이 증가하게 된다. 그러나 변수 선택 기법의 경우에는 열 벡터마다 모든 원소에 대해 분산을 구하지 않고, 표본을 추출하여 모분산을 추정하는 방법을 적용할 수 있고, 분산을 계산한 후에 필요한 연산은 행렬에서 필요한 열을 선택하는 것이 전부이기 때문에 다른 방법들에 비해서 연산 자원 면에서 유리하고, 빅데이터 환경에 적합한 방법이라고 할 수 있다.

표 4. 특이값 분해와 변수 선택 기법의 상대 순위 정확도에 대한 대응 표본 T 검정 결과
Table 4. Paired t-test compares relative ranking precision between singular value decomposition and feature selection method

Dataset	t-statistic	p-value
Movielens	-3.025611	0.012772
Boardgamegeek	-3.804806	0.002187
Jester	-3.525375	0.009657
Datafinti	-1.056642	0.325771

표 5. 특이값 분해와 변수 선택 기법의 수행시간 비교 (ms)
Table 5. Comparisons of execution time between singular value decomposition and feature selection method (ms)

Dataset	SVD	Feature Selection
Movielens	0:00:07.178936	0:00:06.314777
Boardgamegeek	0:02:18.422520	0:00:49.104610
Jester	0:00:06.882968	0:00:06.614407
Datafinti	0:00:01.774745	0:00:01.479170

Ⅳ. 결 론

본 논문에서는 모형을 생성하지 않는 차원 축소 기법으로 랜덤 투영 방법 외에 식 (5)와 같이 계획 행렬에서 분산이 가장 큰 열을 선택하는 변수 선택 기법이 있음을 제시하고, 변수 선택 기법이 코사인 유사도의 상대 순위 정확도와 수행 시간 면에서 랜덤 투영 방법보다 우위에 있음을 보였다. 추가적으로 모형을 생성하는 특이값 분해와도 성능을 비교하여 차원이 일정 이상일 때는 상대 순위 정확도에서는 큰 차이가 나지 않고, 수행 시간 면에서는 유리함을 보였다. 빅데이터 환경에서 모형을 이용하는 차원 축소 기법의 적용이 쉽지 않다는 점을 고려할 때, 모형을 생성하지 않는 차원 축소 기법이 실제적으로 유용하게 사용될 수 있다고 기대되며, 기존의 랜덤 투영 방법보다 빠르면서도 코사인 유사도 보존 측면에서 더 나은 방법을 제시했다는 데 본 논문의 의의가 있다고 생각된다.

참고문헌

[1] P. Cremonesi, Y. Koren, and R. Turrin, “Performance of recommender algorithms on top-n recommendation tasks,” in Proceedings of the fourth ACM conference on Recommender systems, pp. 39-46, 2010.

[2] A. K. Menon, “Random projections and applications to dimensionality reduction,” School of Information Technologies, The University of Sydney, 2007.

[3] Cosine similarity, [Online]. Available:https://en.wikipedia.org/wiki/Cosine_similarity.

[4] G. Takacs and D. Tikk, “Alternating least squares for personalized ranking,” in Proceedings of the sixth ACM conference on Recommender systems, pp. 83-90, 2012.

[5] S. Dasgupta, “Experiments with random projection,” arXiv preprint arXiv:1301.3849, 2013.

[6] E. Bingham and H. Mannila, “Random projection in dimensionality reduction: Applications to image and text data,” in Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and datamining, pp. 245-250, 2001.

[7] G. Strang, Linear algebra and learning from data. Wellesley-Cambridge Press, 2019.

[8] W. B. Johnson and J. Lindenstrauss, “Extensions of lipschitz mappings into a hilbert space,” *Contemporary mathematics*, vol. 26, no. 189-206, p. 1, 1984.

[9] S. Dasgupta and A. Gupta, “An elementary proof of the johnson-lindenstrauss lemma,” International Computer Science Institute, *Technical Report,* vol. 22, no. 1, pp. 1-5, 1999.

[10] Movielens 100k dataset, [Online]. Available: https://www.kaggle.com/prajitdatta/movielens-100k-dataset

[11] Boardgamegeek reviews, [Online]. Available: https://www.kaggle.com/jvanelteren/boardgamegeek-reviews.

[12] Jester 1.7m jokes ratings dataset, [Online]. Available:https://www.kaggle.com/vikashrajluhaniwal/jester-17m-jokes-ratings-dataset.

[13] Grammar and online product reviews, [Online]. Available: https://www.kaggle.com/datafiniti/grammar-and-online-product-reviews.

[14] Random projection, [Online]. Available: https://scikit-learn.org/stable/modules/random_projection.html.

[15] J. Pennington, R. Socher, and C. D. Manning, “Glove: Global vectors forword representation,” in Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP), pp. 1532-1543, 2014.

[16] P. Covington, J. Adams, and E. Sargin, “Deep neural networks for youtube recommendations,” in Proceedings of the 10th ACM conference on recommender systems, pp. 191-198, 2016.

[17] D. C. Liu, S. Rogers, R. Shiau, D. Kislyuk, K. C. Ma, Z. Zhong, J. Liu, and Y. Jing, “Related pins at pinterest: The evolution of a real-world recommender system,” in Proceedings of the 26th International Conference on World Wide Web Companion, pp. 583-592, 2017.

유찬우(Chan-Woo Yoo)

2003년 : 서울대학교 (컴퓨터공학 학사, 경영학 학사)
2012년 : 서울대학교 대학원 (공학박사 - 컴퓨터공학)

2012년～2014년: LG전자
2016년～2018년: 네무스텍
2018년～현 재: 라인플러스 연구원
※관심분야 : 머신러닝, 온라인 러닝, 추천 알고리즘

김희천(Hee-Chern Kim)

1989년 : 서울대학교 (이학사)
1991년 : 서울대학교 대학원 (이학석사)
1998년 : 서울대학교 대학원 (이학박사)

2004년～현 재: 한국방송통신대학교 컴퓨터과학과 교수
※관심분야 : 머신러닝, 컴퓨터교육, 소프트웨어공학