Neighborhood-Based Collaborative Filtering (Application)

Young Jun Jang

Overview

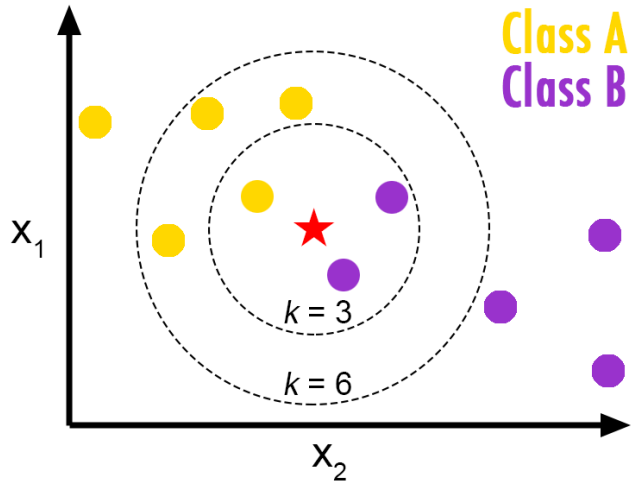
1) Prerequisite

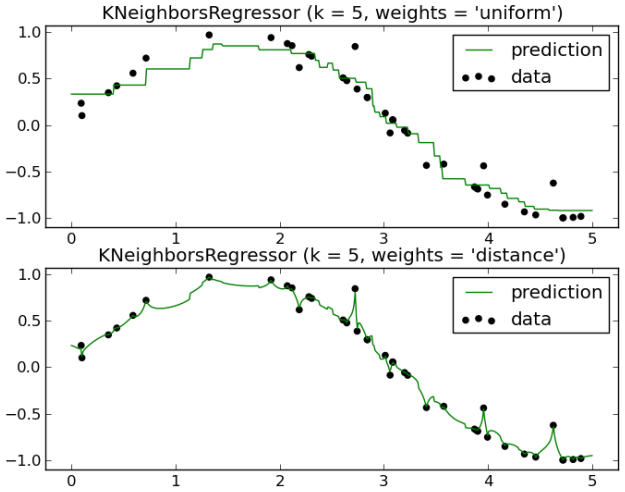
2) Neighborhood-based collaborative filtering

3) Amazon’s Recommender System Case study

4) Reference

01 Prerequisite



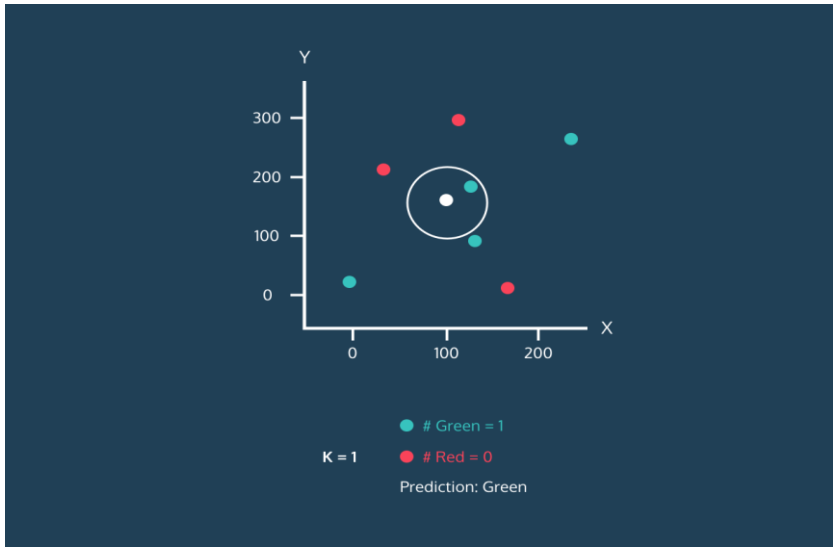


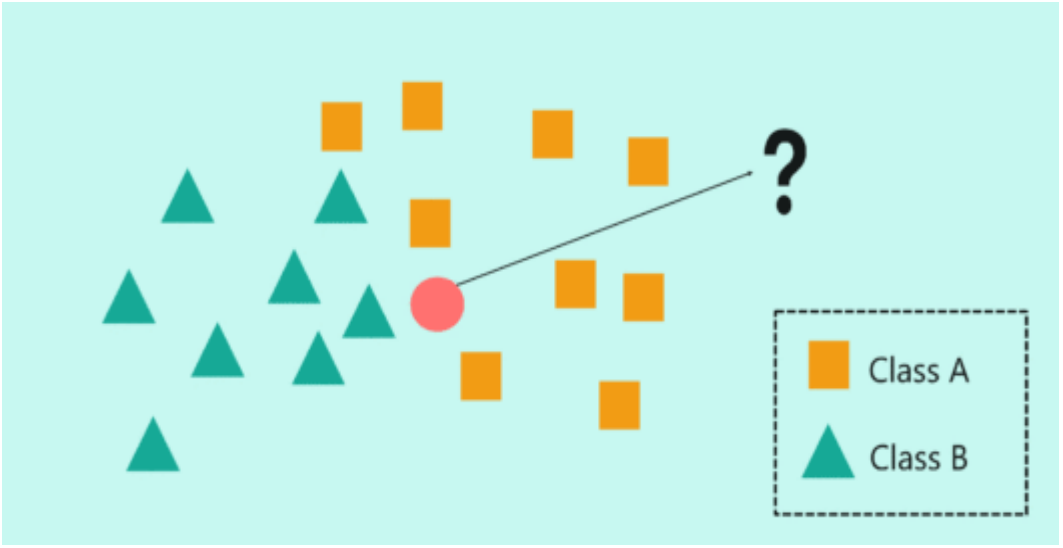
What is KNN approach?

- KNN은 새로운 데이터가 주어졌을 때 기존 데이터 가운데 가장 가까운 K개 이웃의 정보로 새로운 데이터를 예측
- 이웃기반 접근 방법 (Classification, Regression based)

- 새로운 데이터가 들어온 후에, 데이터 사이의 거리를 재서 이웃을 뽑는 개념
- KNN을 모델을 별도로 구축하지 않음으로 lazy model이라고도 불림 (혹은 instanced based learning)
- 별도 모델 생성과정 없이 각각의 관측치(instance)만을 이용하여 분류/회귀 task 수행
- 하이퍼 파라미터는 탐색할 이웃 수(k), 거리 측정 방법 2가지 사용

01 Prerequisite





KNN의 장,단점

- 탐색할 이웃의 수(k)가 작을 경우에는 데이터의 지역적 특성을 지나치게 반영 (overfitting)
- 반대로 k가 매우 클 경우에는 모델이 과하게 정규화 되는 경향 발견 (underfitting)

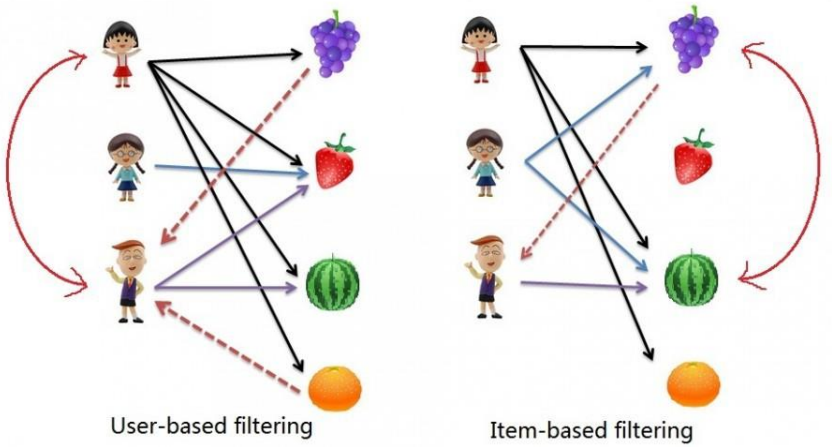
- KNN은 학습 데이터 내에 끼어 있는 노이즈의 영향을 크게 받지 않음, 학습 데이터가 많을수록 효과적인 알고리즘
- 거리 기반 방법론 中에 마할라노비스 거리와 같이 데이터의 분산을 고려할 경우 매우 robust한 방법론으로 알려짐

- 최적의 이웃 수(k)와 어떤 거리 척도가 분석에 적합한지 불분명해 데이터 각각의 특성에 맞게 연구자가 임의로 선정
- 새로운 관측치와 각각의 학습 데이터 사이의 거리를 전부 측정해야 하므로 계산 시간이 오래 걸림.

- Locality Sensitive Hashing 방법을 통하여 계산 복잡성을 줄일 수 있음. (Reformer 모델에서 LSH 사용)

02 Neighborhood-based collaborative filtering

Collaborative filtering



Neighborhood-based collaborative filtering

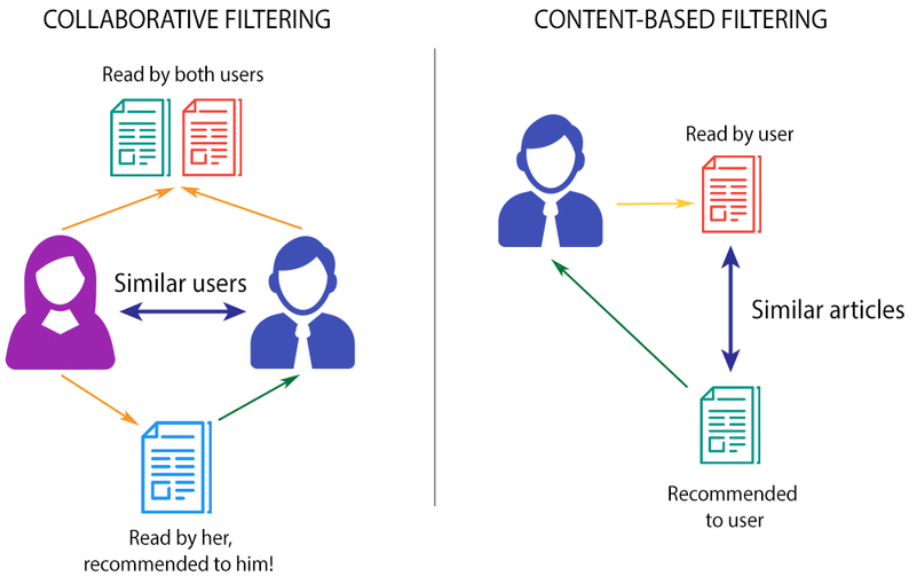
1) User-based filtering

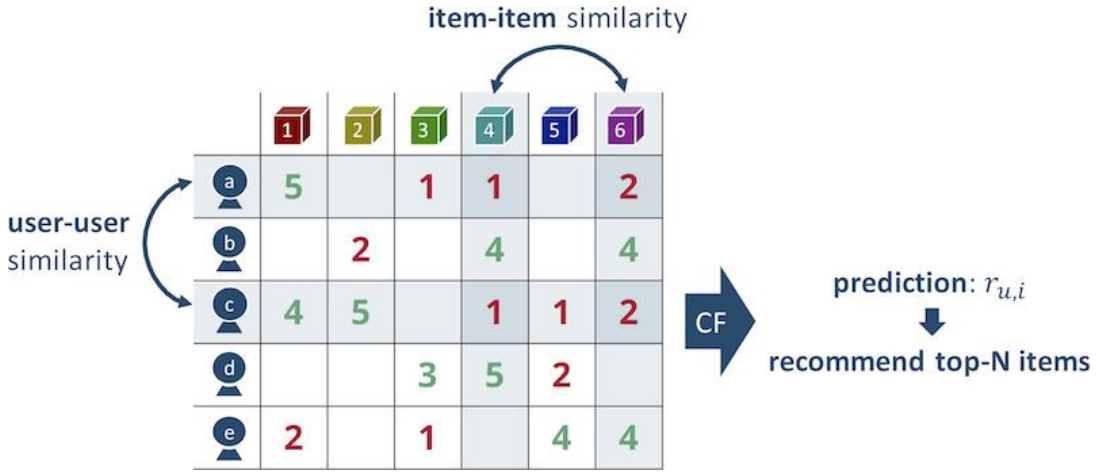
- User a와 유사한 User가 제공한 등급을 가지고 추천리스트 구축
- A의 예상 등급은 각 "피어 그룹" 등급의 가중 평균으로 계산

2) Item-based collaborative filtering

- Item user 행렬을 가지고 있다고 가정
- Item A & B는 서로 비슷한 평점 분포를 가지고 있다고 가정, Item A = B 동일

02 Neighborhood-based collaborative filtering





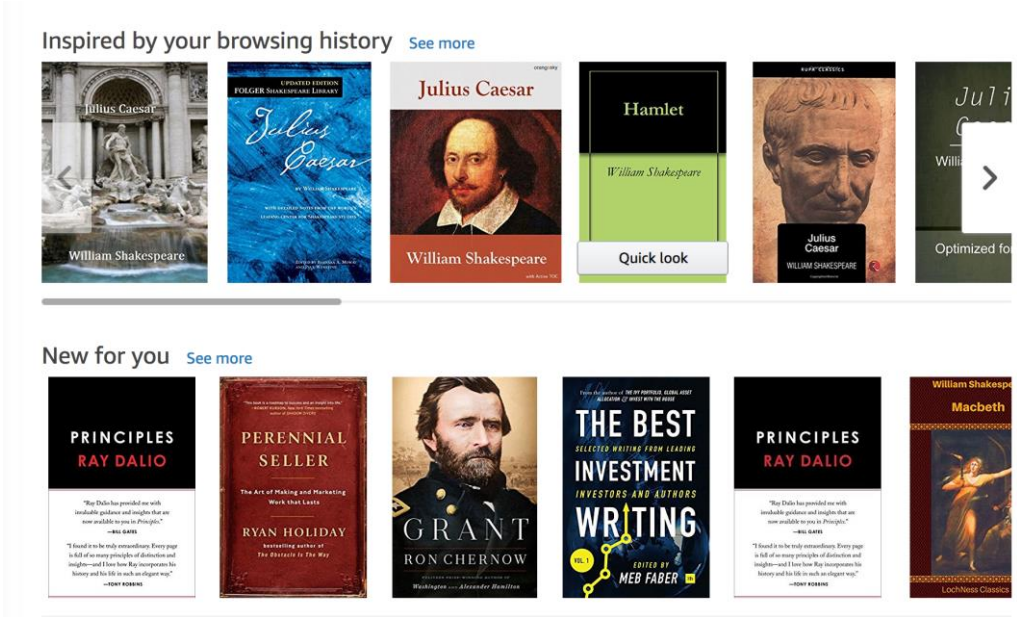
Collaborative filtering

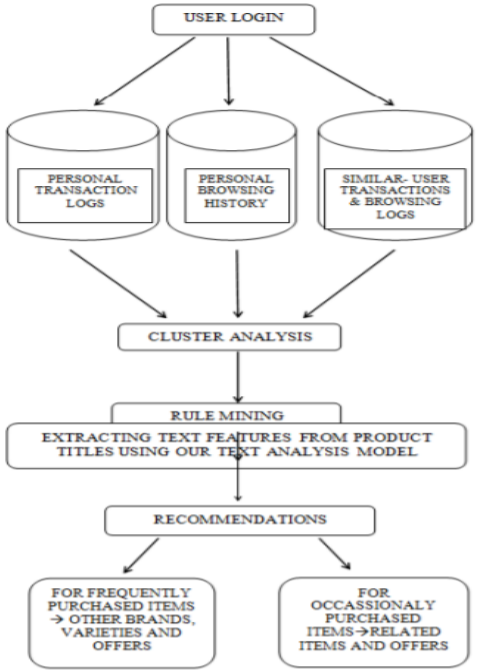
- Collaborative Filtering은 사용자에게 상품을 추천하기 위해서 비슷한 취향을 가진 사용자를 고려하거나,
  또는 사용자가 관심을 갖고 있는 상품과 비슷한 특징을 가진 상품을 고려하여 사용자가 관심을 갖을 만한 상품을 예측하는 방법

- Collaborative Filtering은 2가지 1) User-based filtering, 2) Item-based filtering로 분류.

- User-based 방법은 비슷한 취향을 가진 사용자(User)들을 고려하여 추천을 하고, Item-based 방법은 비슷한 특징의 상품(Item)을
  고려하여 추천

02 Amazon's Recommender System Case Study





Item-based collaborative filtering

- Amazon에서 주로 활용하는 방법론
- 이전에는 Browsing history를 기반으로 item을 추천
- User Login으로 workflow 시작하여 log, browsing history들을 기반으로 군집화
- 군집화 후에는 Rule mining을 통한 연관규칙을 찾아냄

02 Amazon's Recommender System Case Study

Amazon's solution

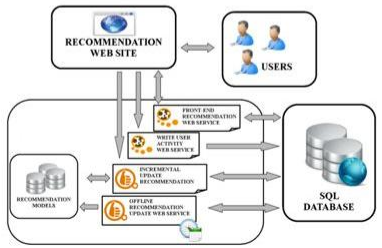
1. Amazon Recommendation Engine

- Amazon's model that implements recommendation algorithm.
- Recommendation algorithm is designed to personalize the online store for each customer.

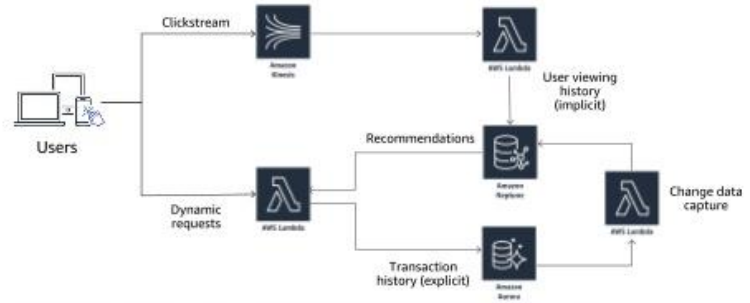
2. Algorithm feature

- Most recommendation algorithms start by finding a set of similar customers whose purchased and rated items overlap the user's purchased and rated items.
- The Amazon's item-to-item collaborative filtering is focusing on finding similar items instead of similar customers.

3. Recommendation Engine Workflow



Collaborative filtering architecture using Amazon Neptune



https://github.com/aws-samples/amazon-neptune-samples/tree/master/gremlin/collaborative-filtering

re:Invent

aws

Recommendation Engine

- 고객을 위한 개인화 서비스로 추천 시스템 제공
- 고객들끼리 구매이력 중에서 서로 겹치는 것이 많은 아이템을 구매한 것을 기반으로 알고리즘 고안(기존)
- Item to item collaborative filtering을 사용
- 비슷한 고객을 찾는 것보다 비슷한 아이템에 집중

02 [Amazon] Item to Item collaborative filtering

Amazon.com Recommendations

Item-to-Item Collaborative Filtering

Greg Linden, Brent Smith, and Jeremy York • *Amazon.com*

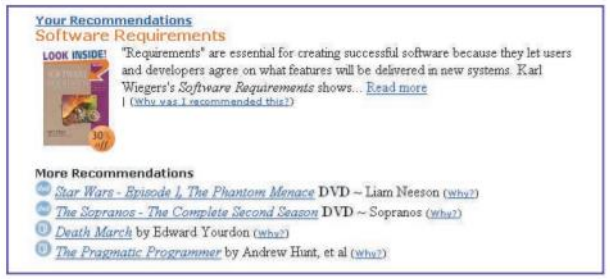


Figure 1. The "Your Recommendations" feature on the Amazon.com homepage. Using this feature, customers can sort recommendations and add their own product ratings.

Figure 2. Amazon.com shopping cart recommendations. The recommendations are based on the items in the customer's cart: The Pragmatic Programmer *and* Physics for Game Developers.

Item to Item collaborative filtering

- 비슷한 사용자를 매칭하는 것 대신에 Item-to-Item CF는 사용자가 구매하거나 평가한 아이템들의 유사성을 찾은 후에 비슷한 아이템을 추천 리스트로 생성
- 가장 유사한 아이템을 정의하기 위해 알고리즘은 유사한 아이템 테이블을 만들어 사용자들이 함께 구매하는 경향이 있는 아이템을 찾음.
- 아이템 Pair와 정확히 일치하는 공통의 고객은 흔하지 않으므로 계산에 있어 시간이나 메모리 사용이 비효율적

02 [Amazon] Item to Item collaborative filtering

$$similarity(\vec{A}, \vec{B}) = \cos(\vec{A}, \vec{B}) = \frac{\vec{A} \bullet \vec{B}}{\|\vec{A}\| * \|\vec{B}\|}$$

```
for each item in product catalog (I1)
    for each customer who bought I1 (C1)
        for each item bought by customer (I2)
            similarity(I1, I2)
```

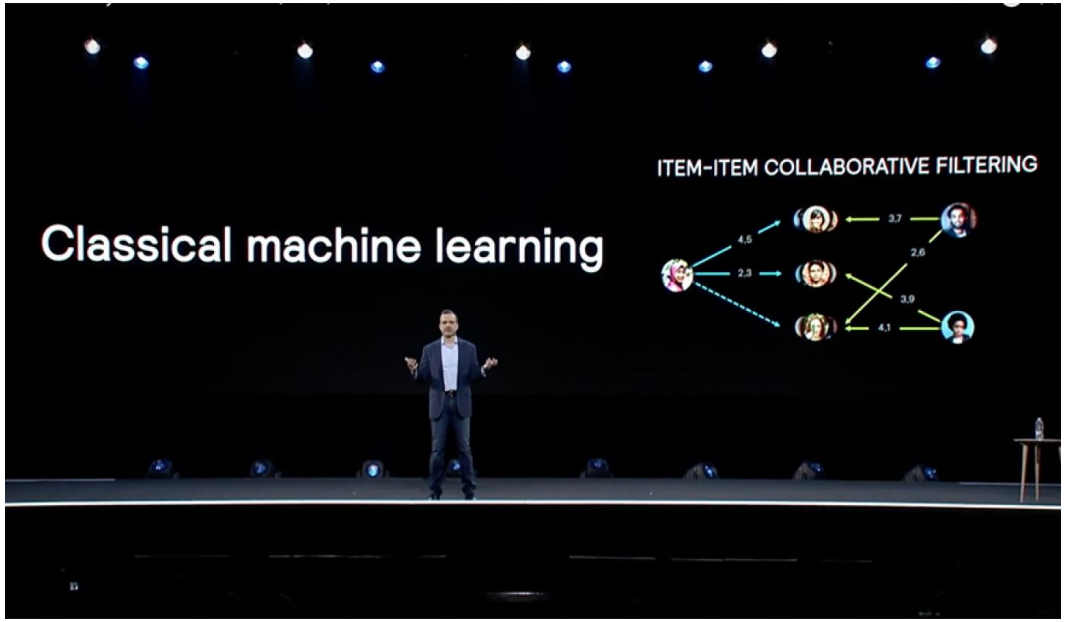
- 전통적인 CF에서는 전체 카탈로그 N개에 대해 사용자마다 N차원의 벡터를 만든 후, 그 벡터 값은 평점으로 채움.
  (넷플릭스 영화의 경우 1~5점 scale의 척도를 가지고 있음)

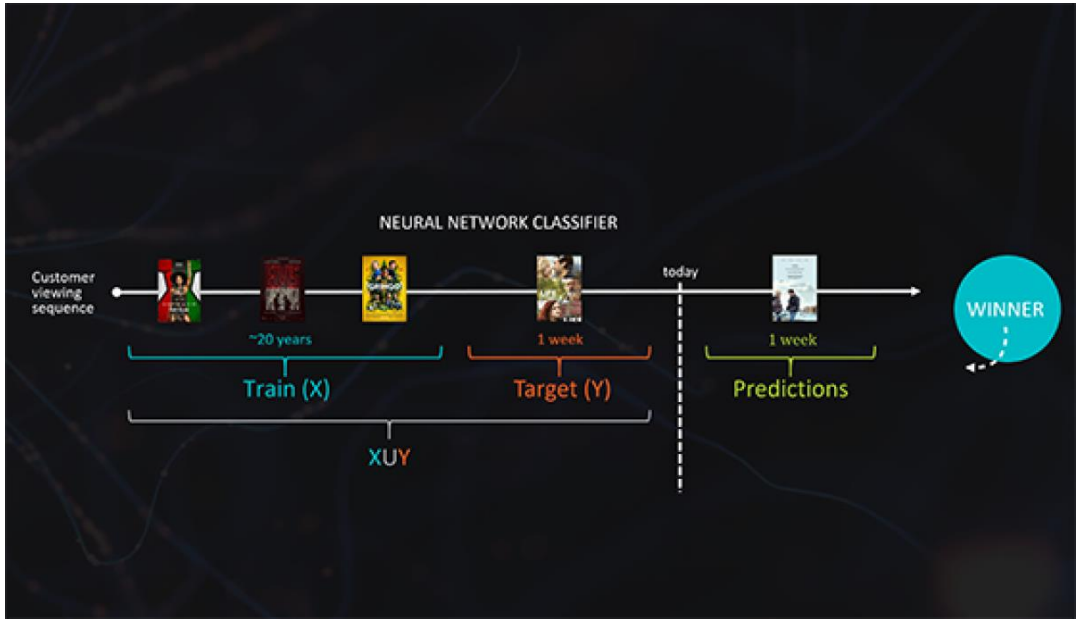
- 하지만 문제는 대부분의 고객은 이 벡터가 매우 Sparse함.

- 알고리즘은 사용자와 비슷한 몇몇 고객 기반으로 추천 생성

- 고객 A와 고객 B의 유사도를 구하는 일반적인 방법은 두 벡터의 코사인 유사도 측정

03 Amazon's Recommender System Case Study





Item-based collaborative filtering

- 머신 러닝 기반으로 예측
- Train (고객의 시청한 규칙, 20년 기준) U Target (1주일 동안 시청) = prediction(예측)

04 Reference

[1] Chandra Sekharan, Sindhu. (2018). RECOMMENDER SYSTEM BASED ON USER'S TRANSACTION AND BROWSING HISTORY USING TEXT ANALYSIS.

[2] https://aws.amazon.com/ko/getting-started/hands-on/recommendation-engine-for-games-amazon-neptune/

[3] https://aws.amazon.com/ko/blogs/korea/amazon-neptune-a-fully-managed-graph-database-service/

[4] https://www.cs.umd.edu/~samir/498/Amazon-Recommendations.pdf

[5] https://lsjsj92.tistory.com/

[6] https://ratsgo.github.io/

[7] https://www.slideshare.net/AmazonWebServices/realtime-personalized-customer-experiences-at-bonobos-ret203-aws-reinvent-2018

THANK YOU