# Introduction to Recommender Systems

## Content-Based Filtering -- Part II, Broad Topics

발제자 15기 염정운

# OBJECTIVES

- **Rating Prediction**(Scoring) + **Recommendation**, Unified Model
- **Searching / Context Aware** + Unified Model

# 1. Review: 추천을 위한 작업

**이 강의에서 다루는 범위의 추천시스템을 만들기 위해선 진행해야 할 핵심적인 작업이 두가지 있습니다.**

1. User가 특정한 Item을 얼마나 좋아할 것인지에 대한 예측(Rating Prediction, or Scoring)하고,
2. 그리고 이를 기반으로 사용자가 실제로 좋아할 Item을 사용자에게 추천(Recommendation)합니다

**단순하게 가장 점수가 높은 Item을 추천하면 되는 것 아니냐는 생각이 들 수 있지만, 꼭 그렇지는 않습니다.**
**자세한 이야기는 뒤에서 다시 한번 이야기 하겠습니다.**

# 2. Scoring Function

먼저 Scoring에 대해서 알아봅시다. 우리가 예측할 Score를 도출하는 함수가 있다고 한다면, 수식적으로 다음과 같이 표현하기로 하겠습니다.

$$s(i;u) = s(u,i)$$

특정한 User $u$ 의 Item $i$ 에 대한 Score function

그런데 단순히 User 하나의 변수로는 충분히 좋은 수준의 추천을 하기 어려운 것 같습니다. 그래서 여기에 두 가지 특성을 더 고려해 **확장된 개념의 Score**를 계산하려 합니다.

# 2. Expanding Scoring

이 두가지 특성이란 앞서 언급한 Search와 Context를 말합니다.

$$s(i;u) = s(u,i)$$

특정한 User $u$ 의 Item $i$ 에 대한 Score function

+

**Search (Query Terms)**

User 입장에서 원하는 조건
" 나는 오늘 점심시간에 코미디가 보고 싶어! "

**Current Context**

User 외부에서 주어지는 환경, 조건
- 상영 영화의 종류, 시간 -

# 2. Full Scoring Function

다시 말해, 점수를 계산하는 함수에 두 가지 변수가 더 더해진 셈이니 다음과 같이 표현할 수 있을 겁니다.

$$s(i; u, q, x)$$

$q$ : Query $x$ : Context

사실, 우리가 지금 다루고 있는 추천시스템 이외에 위 변수들을 적절히 조합해 만든 이전 사례가 몇가지 있습니다.

# 2. Full Scoring Function

| | | |
|---|---|---|
| $s(i; u, q, x)$ | Traditional Recommender | 나는 이 영화를 3000만큼 좋아할 것이다 |
| $s(i; u, q, x)$ | Traditional Search | JAVA date formatting |
| $s(i; u, q, x)$ | Personalized Search | Python, Programming or Snake? |
| $s(i; u, q, x)$ | Context-aware Recommender | 오늘 상영하는 영화 중 추천을 한다 |
| $\boldsymbol{s(i; u, q, x)}$ | **Context-aware Personalized Search Recommender** | **Google, Bing** |

# 3. Computing s

**User, Search, Context 변수들의 조합으로 만들어지는 Score 개념들을 보았습니다. 이를 적용하여 score를 계산하는 기법을 소개합니다.**

| | |
|---|---|
| **Content Based Filtering** | User Taste Profile |
| **Demographics** | User Demographics + Segmented Preferences |
| **Association Rules** | Context of currently-displayed item |
| **Collaborative Filtering** | User Preferences, Community Preferences |

# 4. Scoring to Recommendation

단순히 점수를 계산한다고 해서 추천이 끝나는 건 아니죠. 순위를 통해 적절한 추천 결과를 제시해야 합니다. Score function을 통해 Score를 계산한 것과 마찬가지로, 우리는 실제 추천을 위한 Ordering Function이라는 것을 다음과 같이 정의할 수 있습니다.

$$O(I; u, q, x)$$

이 때 Score function과 비교해 item $i$가 set of items를 의미하는 $I$로 바뀌었습니다. 순서를 메기기 위해서는 여러 아이템들이 전제되어야 하니 당연합니다.

① 모든 가능한 Item들을 추리고
② User / Query / Context 조건에 따라 Score를 측정합니다.

이를 적용한 Recommendation 방식의 예시들을 한번 봅시다.

# 5. Recommendation

## 1. Basic Top-N Recommendation

가장 기본적인 추천 방식입니다. Score function $s(i;u,q,x)$를 통해 도출된 item score를 순서대로 정렬하여 추천하는 방식입니다.

## 2. Tweaking Top-N Recommendation

추천의 다양성을 확보하기 위한 방식입니다. 만약, 톨킨의 <반지의 제왕>을 재미있게 읽었다고 해서 다음 읽을 책으로 <반지의 제왕 양장본>, <반지의 제왕 E-Book>을 추천해준다면 이는 문제가 있겠죠.

## 3. Extended Recommendation

$O(n,I;u,q,x)$

# 결론

1. 추천시스템은 개념적으로 Search, Context를 함께 고려할 수 있습니다. 앞으로 function notation이 계속 활용될 예정입니다.

2. Score function, Ordering function을 구체화하는 다양한 방식이 있습니다.
   - 알고리즘, ML
   - Score function + Ordering function