[추천시스템] 알고리즘 종류 조사

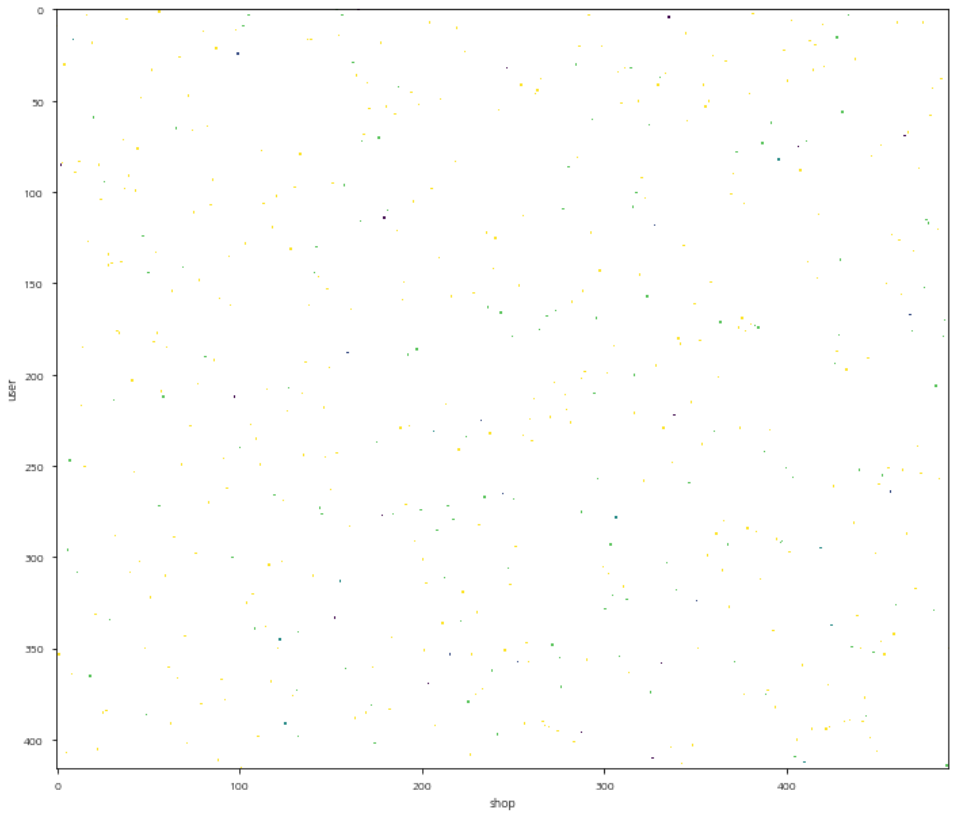
Info

- 현존하는 추천 알고리즘을 조사/스터디
- 팀에 공유하고 후보 알고리즘을 선정
- 목차
  - 비개인화 알고리즘 (Popularity-based)
  - 개인화 알고리즘 (Collaborative Filtering, Contents-based)
  - 모델 성능 평가
  - 알고리즘 후보
  - 주요 이슈
  - 시스템 적용

작성완료

1. 비개인화 알고리즘 (Popularity-based)
   a. 정의
      i. 특정 변수에 가중치를 적용해 스코어를 도출
      ii. 스코어를 통해 랭킹을 만들고 높은 랭킹 위주로 노출
   b. 사례
      i. 큐레이션/이배돼 배라업소 랭킹모델 → [2차] 배라 업소 랭킹 산출 로직 (개선안)
      ii. 배민맛집 랭킹 → [서비스정책] 프론트 노출 정책
      iii. 검색 랭킹모델 → 맛집랭킹
   c. 특징
      i. 계산량이 적고 서비스 적용에 용이 (한번 만들어서 여러번 사용)
      ii. 모델 및 변수에 대한 이해가 쉬움
      iii. 단, 개인의 관심사가 반영이 되지 않음 → 모든 유저에게 동일한 노출
2. ⭐개인화 알고리즘 (Collaborative Filtering, Contents-based)
   a. 정의
      i. 유저 u가 아이템 i를 얼마나 주문할 것인지를 주문수를 예측하는 시스템 → Matrix Completion
      ii. 예시) 유저 by 배라업소 Matrix (수치는 별점)
      iii.

shop_no	7310	202358	396817	419467	439793	491129	514268	532483	548721	565404	570728	581036	583739	584239	587532
mem_no															
141202002872															
141202002942								5							
141202003094															
141202003266				5											4
141202003376															
141202003429						5									
141202003441															
141202003446															
141202003460															
141202003640	5													4	



iv. 예측하고자 하는 실수 예시
    1. 별점 (1 ~ 5)
    2. 좋아요 수
    3. 주문수
    4. 클릭수 등등

b. 모델 종류

i. Baseline Model

1. 평균점수에 각 유저 및 아이템의 평균점수를 합산해서 실수값 예측 → 가장 단순한 형태의 모델
2. 최적화 목표

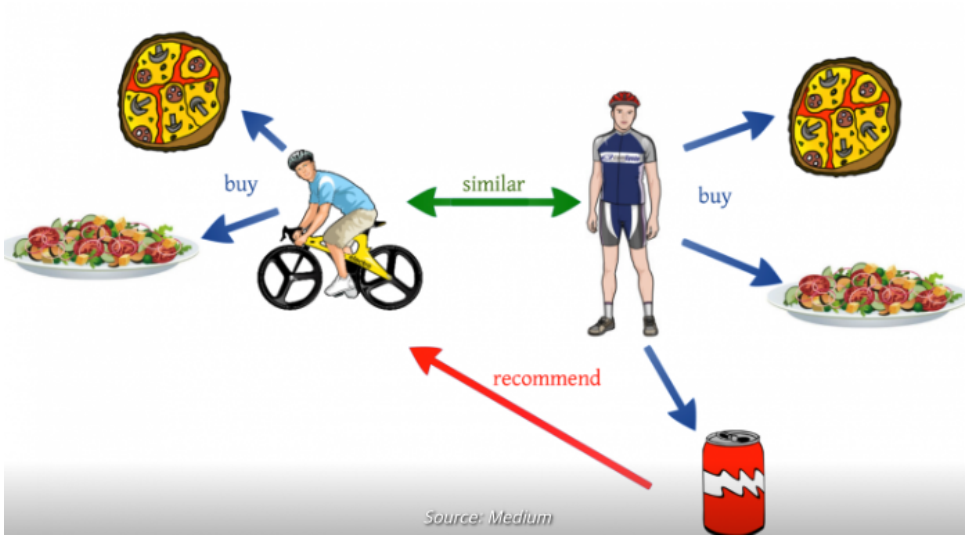
a. $$minimize \sum (r_{ui} - \hat{r}_{ui})^2 + \lambda(\sum_u {b_u}^2 + \sum_i {b_i}^2)$$, where $$\hat{r}_{ui} = \bar{r} + b_u + b_i$$

i. r bar: 전체 평균
ii. bu : 전체 유저 평균 대비 편차
iii. bi: 전체 아이템의 평균 대비 편차

3. 일반적으로, Baseline Predictor with Temporal Models 로 적용할 경우 성능이 높을 수 있음
4. 참고사이트: http://sanghyukchun.github.io/31/

ii. CF - Neighborhood Method

1. 정의: 유저(user-based) 혹은 아이템(item-based)를 특정 유사도를 기준으로 가중치를 적용



a.
b. 참고사이트: https://www.analyticsvidhya.com/blog/2018/06/comprehensive-guide-recommendation-engine-python/

2. 유사도 종류
   a. 코사인 유사도 (Cosine Similarity)

$$\cos\theta = \frac{x \cdot y}{|x||y|}$$

   i.
   b. 평균제곱차이 유사도 (Mean Squared Difference Similarity)

$$\mathrm{msd}(u,v) = \frac{1}{|I_{uv}|} \cdot \sum_{i \in I_{uv}} (r_{ui} - r_{vi})^2$$

   i. ,

$$\mathrm{msd\_sim}(u,v) = \frac{1}{\mathrm{msd}(u,v)+1}$$

   ii. → 역수로 치환 (1은 0을 방지하기 위해 더해줌)
   c. 피어슨 유사도 (Pearson Similarity)

$$\text{pearson\_sim}(u, v) = \frac{\sum_{i \in I_{uv}} (r_{ui} - \mu_u) \cdot (r_{vi} - \mu_v)}{\sqrt{\sum_{i \in I_{uv}} (r_{ui} - \mu_u)^2} \cdot \sqrt{\sum_{i \in I_{uv}} (r_{vi} - \mu_v)^2}}$$

i.

d. 자카드 유사도 (Jaccard Index)

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|}.$$

i.

3. 가중치 반영 with KNN
   a. KNN(K Nearest Neighbors) 기반으로 유사도를 이용해 가중평균을 산출
   b. 점수에 가중평균

$$\hat{r}_{ui} = \frac{\sum_{v \in N_i^k(u)} \text{sim}(u, v) \cdot r_{vi}}{\sum_{v \in N_i^k(u)} \text{sim}(u, v)}$$

i.

c. 평균에 가중치 반영

$$\hat{r}_{ui} = \mu_u + \frac{\sum_{v \in N_i^k(u)} \text{sim}(u, v) \cdot (r_{vi} - \mu_v)}{\sum_{v \in N_i^k(u)} \text{sim}(u, v)}$$

i.

d. 베이스라인모형의 결과에 가중치 반영

$$\hat{r}_{ui} = b_{ui} + \frac{\sum_{v \in N_i^k(u)} \text{sim}(u, v) \cdot (r_{vi} - b_{vi})}{\sum_{v \in N_i^k(u)} \text{sim}(u, v)}$$

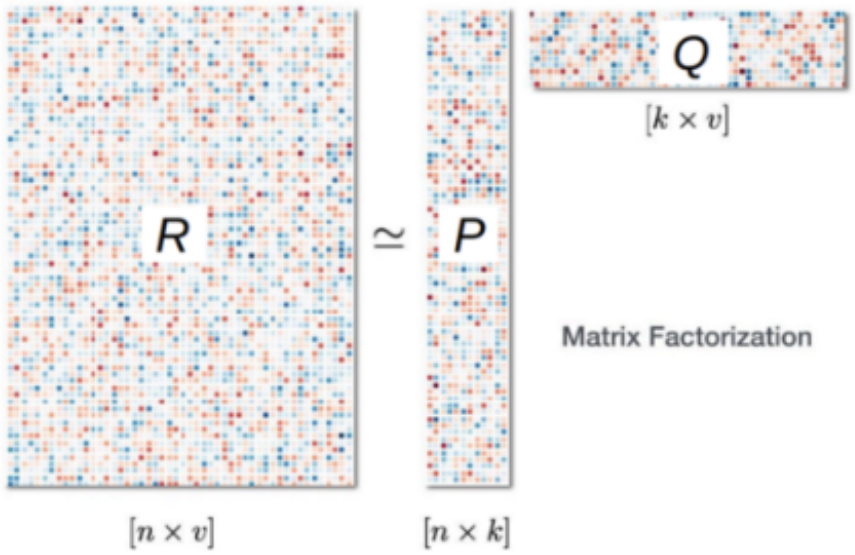
i.

e. 출처: https://datascienceschool.net/view-notebook/fcd3550f11ac4537acec8d18136f2066/

4. 특징
   a. 독자적으로 성능이 높지 못한 편이므로 Baseline 모델과 하이브리드로 이용 가능
   b. 비교적 계산량 높음 → 비효율성 & 부하 발생 가능성

iii. ⭐CF - Matrix Factorization

1. 정의: 잠재요인(Latent Factor) k를 파악해 개별 값과의 내적값을 통해 실수 예측
2. 잠재 요인 k 개만 계산에 이용하므로 neighborhood 모델에 비해 효율적인 계산이 가능

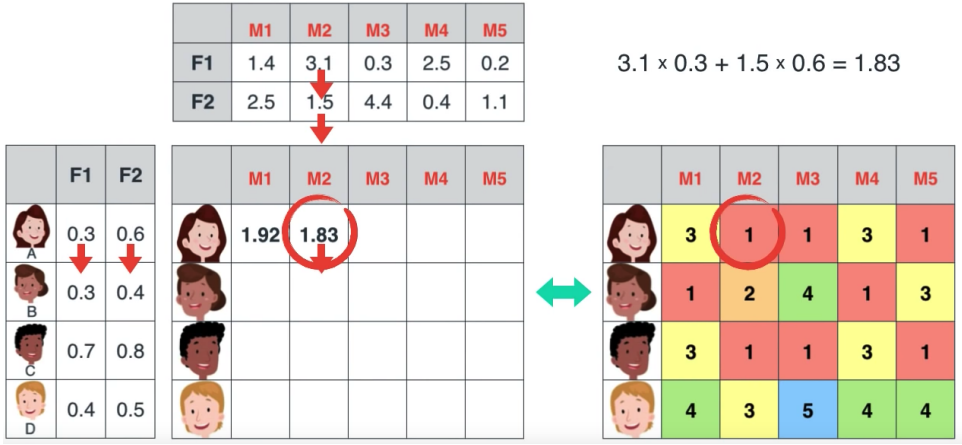


a.

$$R \approx PQ^T$$

$R \in \mathbf{R}^{m \times n}$: m 사용자와 n 상품의 평점 행렬
$P \in \mathbf{R}^{m \times k}$: m 사용자와 k 요인의 관계 행렬
$Q \in \mathbf{R}^{n \times k}$: n 상품의와 k 요인의 관계 행렬

b.



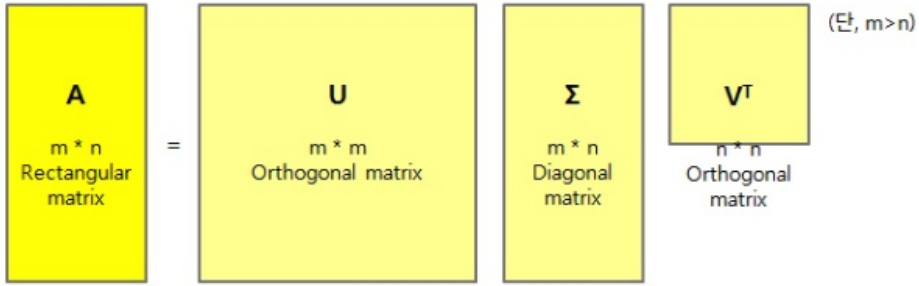
i. 참고사이트 https://www.youtube.com/watch?v=ZspR5PZemcs&t=1762s

c. SVD (Singular Value Decomposition)

i. $$R = U\Sigma V^T$$

U 는 $m \times m$ 크기의 행렬로 역행렬이 대칭 행렬
Σ 는 $m \times n$ 크기의 행렬로 비대각 성분이 0
V 는 $n \times n$ 크기의 행렬로 역행렬이 대칭 행렬

ii.



iii.

iv. 참고

1. 선형독립, 선형종속 및 RANK의 개념: http://rfriend.tistory.com/176?category=606751
2. 차원축소: http://excelsior-cjh.tistory.com/167
3. SVD: http://rfriend.tistory.com/185

3. 최적화 목표

$$L = ||R - P \times Q^T||_2 + \lambda \left(||P||_2 + ||Q||_2\right)$$

$$\hat{r}_{ij} = p_i^T q_j = \sum_k p_{ik} q_{kj}$$

$$argmin_{q,p} \sum_{i,j} (r_{ij} - p_i^T q_j)^2$$

a.

c. 최적화 알고리즘

i. ALS (Alternating Least Square)

1. P 혹은 Q를 고정시키고 나머지에 대해 점진적으로 최적화 작업

Losses:

$$\forall p_i : L(p_i) = ||R_i - P_i \times Q^T||_2 + \lambda \cdot ||p_i||_2$$
$$\forall q_j : L(q_j) = ||R_i - P \times Q_j^T||_2 + \lambda \cdot ||q_j||_2$$

Solutions:

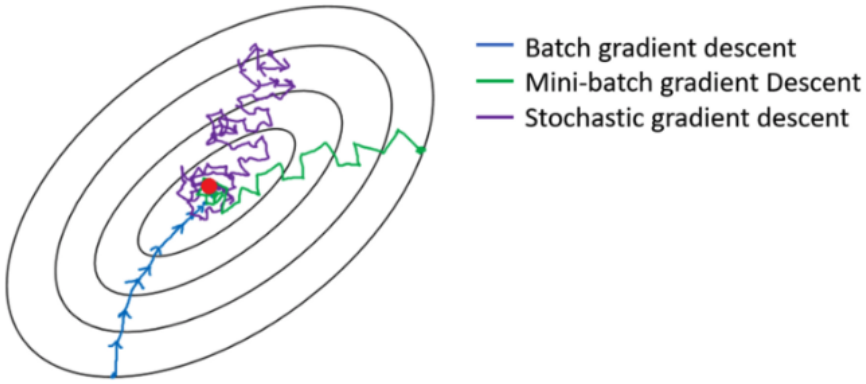
$$p_i = \left(Q^T \times Q + \lambda I\right)^{-1} \times Q^T \times R_i$$
$$q_j = \left(P^T \times P + \lambda I\right)^{-1} \times P^T \times R_j$$

a.

ii. 다른 솔루션으로 SGD(Stochastic Gradient Descent) 존재
   1. Gradient Descent 대비 계산이 빠름, Batch Gradient Descent 에 비해 local minima에 빠질 가능성 낮음

$$w = w - \alpha \nabla_w J(x^i, y^i; w)$$

   a.
   2. 비록 Variance 가 높아 지그재그의 형태로 수렴하나 속도가 빠른 것이 장점



   a.
   b. 참고: http://rasbt.github.io/mlxtend/user_guide/general_concepts/gradient-optimization/
   c. 참고: http://shuuki4.github.io/deep%20learning/2016/05/20/Gradient-Descent-Algorithm-Overview.html

d. Content-Based
   i. 정의: 컨텐츠(예, 업소태그, 메뉴/카테고리, 리뷰 텍스트 등)에 TF-IDF 가중치를 이용해 선호 업소와 유사도가 높은 순서로 노출하는 방식
      1. 예) 배라 업소 태그 정보

	shop_no	rgn2_nm	content
0	575237	강남구	디저트,커피, 디저트·커피, 달콤, 음료판매, 해쨍쨍, 집에서데이트, 비쥬얼, 밀푀...
1	613248	강서구	한식, 비오는날, 밥집, 매콤, 점심, 저녁, 집에서데이트, 내방, 가족식탁용, 진...
2	616996	양천구	한식, 사무실, 비오는날, 밥집, 국밥성애자, 점심, 들깨굴떡국, 저녁, 내방, 가...
3	632641	서초구	디저트·커피, 패밀리, 주말, 달콤, 음료판매, 친구들이랑, 부드러운, 집에서데이트...
4	641984	관악구	고기, 한식, 음료판매, 한우국밥, 회식, 저녁, 부드러운, 나들이, 육즙이흐르는,...

2.

ii. 단어 전처리 (특징 추출)
   1. CountVectorizer
      a. 단순히 단어별 빈도로 처리
   2. TfidfVectorizer
      a. TF-IDF: 단어 빈도와 역문서 빈도를 곱한 값

$$\mathrm{tfidf}(t, d, D) = \mathrm{tf}(t, d) \times \mathrm{idf}(t, D)$$

      b.
      c. 참고: https://blog.breezymind.com/2018/03/02/sklearn-feature_extraction-text-2

iii. 유저별 기준업소와 유사도 높은 업소 기준으로 노출
iv. 다른 모델과 하이브리드로 사용 가능
   1. 예) Popularity-based 모델 기준, 특정 스코어 이상에 해당되는 업소리스트에서 선호업소와 유사도가 높은 순서대로 추천
   2. 신규 유저 혹은 추천해줄 업소 또는 메뉴가 없는 경우

v. 고려사항
   1. implicit feedback 자체가 없는 신규 유저의 경우 여전히 cold-starting 문제 발생
   2. 컨텐츠의 Quality 확보 및 관리 필요

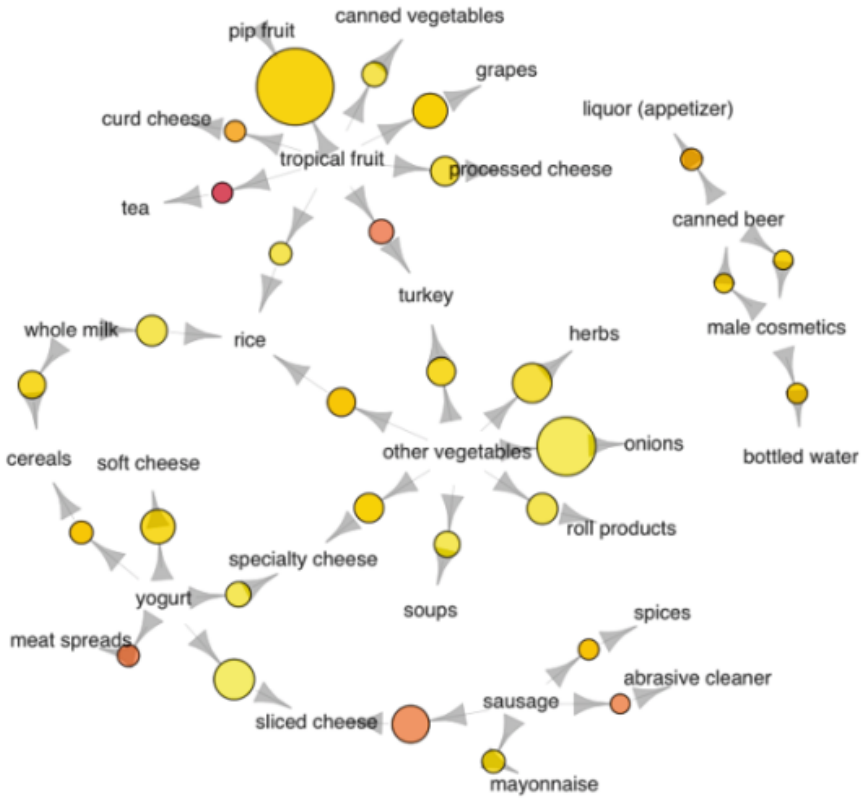
e. Association Rules (a.k.a Basket Analysis)
   i. 아이템 집합의 발생 빈도, 아이템간 신뢰도, 아이템간 통계적 독립여부 등의 지표를 이용해 후발 아이템 예측
   ii. 계산 효율이 높고 예전부터 현재까지 널리 쓰이는 추천 로직
      1.

$$Support = \frac{frq(X,Y)}{N}$$

$$Rule:\ X \Rightarrow Y \longrightarrow Confidence = \frac{frq(X,Y)}{frq(X)}$$

$$Lift = \frac{Support}{Supp(X) \times Supp(Y)}$$

2. 출처: https://www.saedsayad.com/association_rules.htm



Associations between selected items. Visualized using the arulesViz R library.

3.
4. 출처: https://algobeans.com/2016/04/01/association-rules-and-the-apriori-algorithm/

3. 모델 성능 평가
   a. RMSE(Root Mean Squared Error)

$$\text{RMSE} = \sqrt{\frac{1}{|\hat{R}|}\sum_{\hat{r}_{ui}\in\hat{R}}(r_{ui}-\hat{r}_{ui})^2}$$

      i.
   b. MAE(Mean Absolute Error)

$$\mathrm{MAE} = \frac{1}{|\hat{R}|} \sum_{\hat{r}_{ui} \in \hat{R}} |r_{ui} - \hat{r}_{ui}|$$

i.

c. Click Through Rate, Conversion Rate
d. Precision At K
e. NDCG (Normalized Discounted Cumulative Gain)

$$\mathrm{DCG_p} = \sum_{i=1}^{p} \frac{rel_i}{\log_2(i+1)} = rel_1 + \sum_{i=2}^{p} \frac{rel_i}{\log_2(i+1)}$$

i.

f. 설문조사 (정성/정량 만족도)

4. 알고리즘 후보
   a. Matrix Factorization with ALS, SGD
      i. 이유1) 몇 개의 k 요인으로 많은 정보에 대한 표현이 가능 → 효율적 운영 및 빠른 계산이 가능할 것으로 기대
      ii. 이유2) 단일모델로는 성능이 가장 높은 것으로 알려짐
   b. Content-based Model
      i. 이유1) 업소에 대한 컨텐츠(설명, 메뉴, 리뷰 등 텍스트)를 이용해 쉽게 모델 구축 가능
      ii. 이유2) 추후 Baseline 모델과 하이브리드로 구축 고려 가능
   c. Baseline Model
      i. MF 및 content-based 모델의 상대적 성능 비교를 위한 벤치마크
      ii. 구축하기 쉽고 하이브리드 모델로 활용하기 용이
   d. CF - Neighbor model (user-based, item-based)
5. 주요 이슈
   a. Cold-starting 이슈
      i. Content-based 및 베이스라인 모델로 보완
      ii. 클릭, 업소 체류시간 등 implicit feedback을 이용하여 유저 선호/성향 예측
      iii. Baseline 모델 적용 후 짧은 주기로 학습(업데이트)
6. 시스템 적용
   a. 모델 고도화
      i. 업데이트 주기
      ii. A/B Test 통한 모델 선정
   b. 시스템
      i. 연산량 / 속도
      ii. 개발 환경
   c. 프로덕트
      i. 기획/디자인
      ii. 개발 진행