



# Final Project Wrap-up Report

RecSys-2조(R\_AI\_SE) | 김수진, 김창영, 박승아, 전민서, 한대희, 한예본

## 프로젝트 소개

- 주제 : 크롤노티 웹사이트 로그 데이터를 활용한 개인화 추천시스템 개발
- 소개 :



- 본 프로젝트는 '크롤노티' 웹사이트의 로그 데이터를 활용하여 개인화 및 세션 기반 추천 시스템을 개발하는 것을 목표로 함
- '크롤노티'는 쿠팡의 할인 정보를 실시간으로 제공하는 전자기기 특가 알림 사이트로, 사용자에게 최소 10%에서 최대 60% 할인된 가격의 특가 정보를 제공
- 이를 데이터 처리, 모델링, 서빙, 평가 및 테스트 등의 AI production serving 전반적인 사항을 경험하고자 함

## 프로젝트 일정 및 협업 방식

- 프로젝트 일정
  - 프로젝트 개발환경 구축
  - EDA를 통한 데이터 구조 파악
  - 가설 기반 모델링
  - Product Serving
- 협업 방식
  - Github 기반 작업
    - 이슈 기반 작업
    - 데일리 스크럼/피어세션을 활용해 코드 리뷰 후 병합
    - GitHub Kanban Board를 활용한 프로젝트 일정 관리

## 프로젝트 팀 구성 및 역할

- 김수진 : 데이터 분석 및 정제, TopicModeling, EASE, TM2LGCN, Serving API 개발
- 김창영 : 데이터 수집 및 정제, Airflow를 활용한 ELT 파이프라인 설계, Serving API 개발
- 박승아 : 데이터 분석 및 정제, 콘텐츠 기반 추천, (TF-IDF, Word2Vec), IBCF(Item2Vec), Looker Studio
- 전민서 : SASRec, EDA, 파이프라인 설계, 클라우드 기반 인프라 구축, 데이터 정제

한대희 : 데이터 분석 및 정제, 데이터 클리닝, 메모리 기반 협업 필터링(UBCF, IBCF)

한예본 : 데이터 분석 및 정제, EDA(Looker Studio), Sequential Recommendation(SASRec)

## 프로젝트 수행 결과

### Data

- 데이터 소개

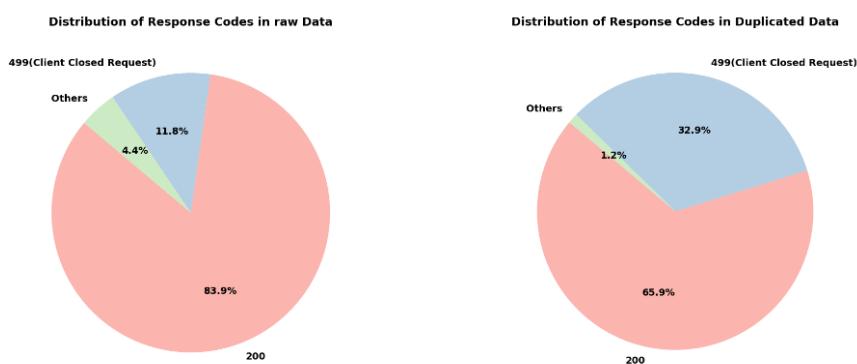
- 메인 데이터 : Nginx에서 정제한 UserIP, Timestamp, Request Destination, user\_device\_endpoint로 구성된 로그 데이터

```
347b3eedsefdbc5b8a6d8664aa703783ce14 [13/Feb/2024:00:00:03 +0900] "GET /products/a5b557d332592ab54ad3b4f2eda40903f54a5baffad1e985d37dbe6d1f/ HTTP/2.0" 200 4152 [] "Mozilla/5.0 (Linux; Android 13; SM-S901N Build/TP1A.220624.014; wv) AppleWebKit/537.36 (KHTML, like Gecko) Version/4.0 Chrome/102.0.5005.189 Whale/1.0.0.0 Crosswalk/27.102.0.20 Mobile Safari/537.36 NAVER(inapp; search; 1010; 11.24.3)" "-"
347b3eedbcgeges5b8a6d8664aa703783ce14 [13/Feb/2024:00:00:03 +0900] "GET /stream/?path=discount HTTP/2.0" 499 0 [] "Mozilla/5.0 (Linux; Android 13; SM-S901N Build/TP1A.220624.014; wv) AppleWebKit/537.36 (KHTML, like Gecko) Version/4.0 Chrome/102.0.5005.189 Whale/1.0.0.0 Crosswalk/27.102.0.20 Mobile Safari/537.36 NAVER(inapp; search; 1010; 11.24.3)" "-"
347b3eedbcssq5b8a6d8664aa703783ce14 [13/Feb/2024:00:00:04 +0900] "GET /comments/a5b557d33259wrqj70ec192ab54ad3b4f2eda40903f54a5baffad1e985d37dbe6d1f/?cidx=0&ridx=20&endpoint=HTTP/2.0" 200 1730 [] "Mozilla/5.0 (Linux; Android 13; SM-S901N Build/TP1A.220624.014; wv) AppleWebKit/537.36 (KHTML, like Gecko) Version/4.0 Chrome/102.0.5005.189 Whale/1.0.0.0 Crosswalk/27.102.0.20 Mobile Safari/537.36 NAVER(inapp; search; 1010; 11.24.3)" "-"
```

- 사이드 정보: 크롤노티 DB에서 API를 통해 수집한 사용자 및 상품 정보

- 데이터 전처리

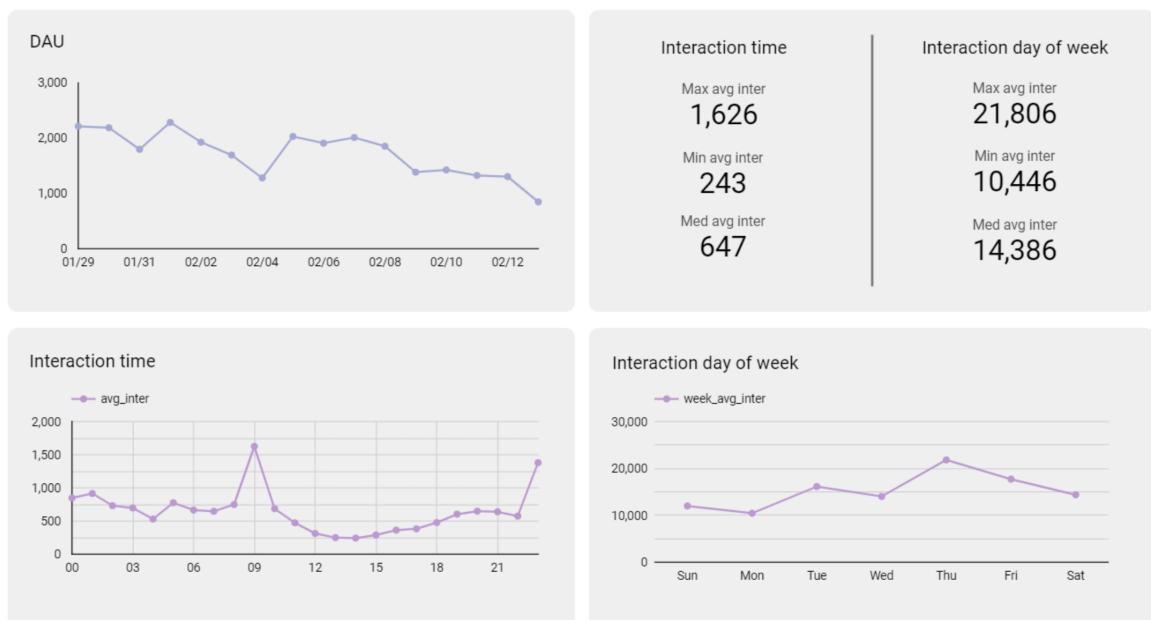
- 비회원 기반 서비스로서, 하나의 유동 IP를 하나의 사용자로 정의하여 session based recommendation을 구현
- User-item interaction에 대한 기준을 크롤노티를 통해 쿠팡 사이트로 바로 들어가는 로그, 크롤노티의 상세페이지로 들어가 확인하는 로그로 정함
- 데이터 클리닝
  - 데이터 분석 과정에서 원본(raw) 로그 데이터와 중복된 로그 데이터의 HTTP 응답 코드 분포를 비교한 결과, 중복된 로그 데이터에서 'Client error response' (499 응답 코드)와 같은 특정 에러 응답들이 원본 로그 데이터에 비해 상대적으로 높은 비율인 것을 확인



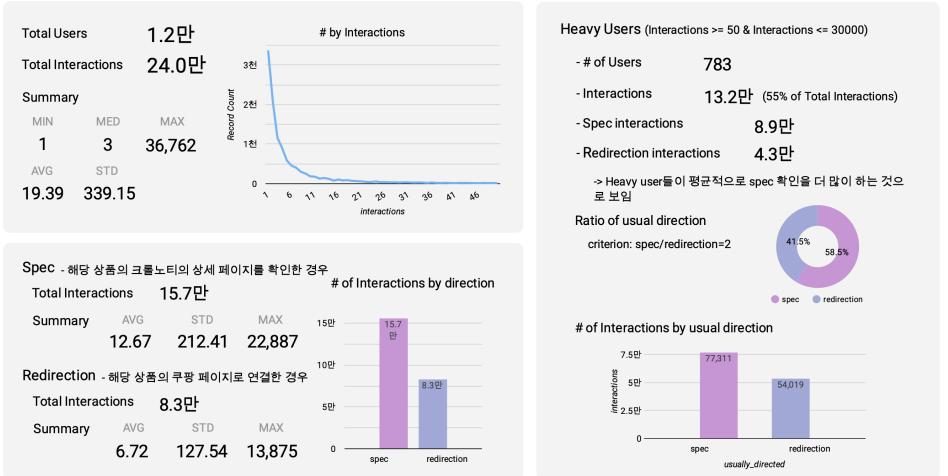
- 이는 중복 되는 로그 데이터가 여러 발생이나 비정상적인 연결 종료와 같은 데이터를 포함할 가능성이 높음을 시사하므로 데이터 클리닝이 필요하다는 결론에 도달함. Client error response 499 와 Redirect response 302 등의 웹 요청 및 응답 프로세스 중 오류, 데이터 수집 시스템의 이중 기록으로 인하여 중복되는 로그 데이터들 제거하였으며, 일정 시간(예: 10초) 내에 3회 이상 연속적으로 상호작용하면 이를 어뷰징 데이터로 간주하고 제거

- EDA

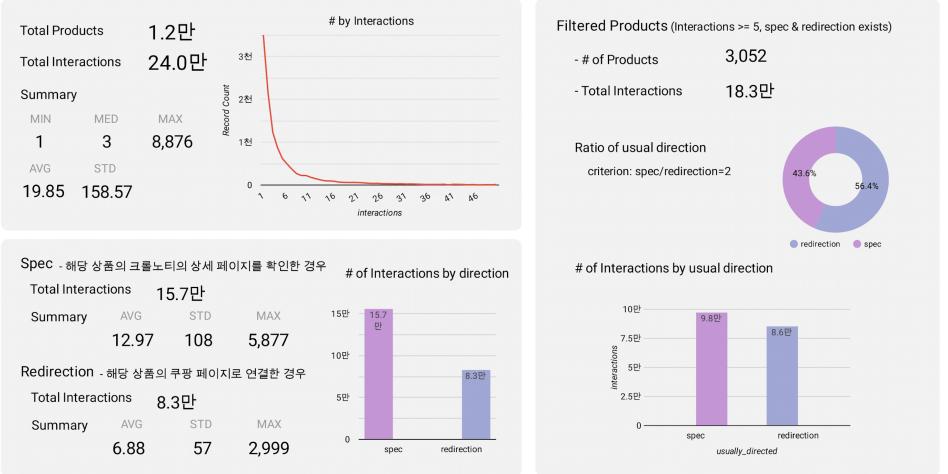
## Time Series (2024.01.29~2024.02.13.)



## Users (2024.01.29~2024.02.13.)



## Products (2024.01.29~2024.02.13.)



## Offline Metric

- 평가 지표 : Hit@10
- 2024.01.29.~ 24.02.13.까지의 데이터를 바탕으로 유저가 마지막으로 조회한 상품을 맞추는 task를 수행
- 모델은 조회 이력이 2회 이상(sequential model의 경우 3회 이상)인 유저에 한해 10개의 상품을 추천하고, 이 추천 결과에 유저가 마지막으로 조회한 상품이 있다면 Hit인 것으로 가정
- Hit@10 베이스라인은 크롤노티에서 현재 운영 중인 추천시스템이 없는 관계로 유저들이 가장 많이 본 10개의 아이템을 동일하게 추천, 0.145의 점수를 기록

## Model 실험

- Content-Based Recommendation
  - TF-IDF
    - TF-IDF : 여러 문서가 있을 때 어떤 단어가 특정 문서 내에서 얼마나 중요한지를 나타내는 통계적 수치
    - task1을 목적으로 개발하였으며 유저 개개인이 조회한 모든 상품들의 타이틀을 하나의 문자열로 만들고, 핫딜 후보 상품의 타이틀을 가져와 이 두 자연어 간의 코사인 유사도를 구함
    - 이 유사도는 유저별 개인 점수로 핫딜 지수에 반영
    - Hit@10 : 0.343
  - Word2Vec
    - Word2Vec : 자연어 처리 분야에서 사용되는 단어 임베딩 기술 중 하나로, 비슷한 맥락에서 사용되는 단어들은 비슷한 벡터를 가짐
    - task2를 목적으로 개발하였으며 모든 상품 타이틀을 학습하도록 하였으며 유저가 조회한 아이템의 타이틀과 코사인 유사도를 구해 유사한 Top K 개의 아이템을 노출하고자 함
    - 새로 등장한 아이템에 대한 벡터를 구할 수 있도록 n-gram을 사용하는 fastText 활용
    - 같은 자연어를 이용하는 TF-IDF에 비해 학습 속도가 느리고 낮은 성능을 가져 사용하지 않음
    - Hit@10 : 0.273
- Collaborative Filtering
  - User-Based CF
    - User-Based CF : 사용자 간의 유사도를 기반으로하여 사용자에게 상품을 추천하는 방식
    - Cosine similarity 를 이용하여 사용자간 유사도 행렬을 생성하고, 입력 사용자와 가장 유사한 상위 3명의 사용자를 찾아냄
    - 입력 사용자의 선호도와 유사한 사용자들의 선호도를 기반으로 다양한 상품을 추천
    - Hit@10 : 기존 사용자의 상호작용 데이터가 없는 새로운 상품을 추천하기 위한 목적으로, 본 실험에서는 평가지표를 사용하지 않음
  - Item-Based CF
    - Item-Based CF : 상품 간의 유사도를 기반으로하여 사용자에게 상품을 추천하는 방식
    - Cosine similarity를 활용하여 아이템 간 유사도 행렬을 생성하고, 입력 아이템과 유사도가 높은 아이템을 상위 n개 추천
    - 입력 아이템과 이미 상호작용한 아이템들중 유사한 상품을 추천함으로써 사용자에게 다양한 상품을 추천
    - Hit@10 : 기존 사용자의 상호작용 데이터가 없는 새로운 상품을 추천하기 위한 목적으로, 본 실험에서는 평가지표를 사용하지 않음
  - Item2Vec
    - Item2Vec : Word2Vec 개념을 추천 시스템에 적용한 모델로, 같은 세션 안에 자주 존재한 아이템들끼리는 벡터상에 가까이 위치하도록 임베딩

- 세션을 문장으로, 세션 안에 속한 아이템을 단어로 하여 특정 아이템의 id를 입력으로 넣을 시, 이와 유사한 아이템을 출력으로 보여줌
- Hit@10 : 학습에 소요되는 자원이 너무 많아 실제 모델 서빙에는 적합하지 않다고 판단하여 성능 지표를 확인하지 않음
- Sequence
  - SASRec
    - SASRec : Transformer 구조를 sequential recommendation에 적용, self-attention 기반으로 global dependency를 학습하고, recurrence 구조를 없애 병렬적 계산이 가능
    - 유저가 조회한 아이템을 바탕으로 유저 임베딩 예측, 시퀀스의 최대 길이는 10으로 사용
    - Hit@10 : 0.292
- Graph
  - SR-GNN
    - LSTM과 GNN을 결합한 모델로, LSTM만을 이용했을 때 아이템간의 상호작용 정보를 반영하기 힘들다는 점을 개량
    - 모델을 학습하는 시간이 다른 모델 대비 너무 오래 걸려서 사용하지 않음
    - Hit@10 : 오프라인 점수가 사용하지 않게 된 주 원인이 아니기에, 측정한 데이터가 존재하지 않음
  - TM2LGCN
    - LightGCN: GCN에서 피처 변환과 비선형 활성화 함수를 제거해 상대적으로 가벼운 모델, 사용자와 아이템 간의 상호 작용을 그래프로 표현하고 이 그래프를 통해 임베딩 학습
    - TM2LGCN은 Topic Modeling으로 생성된 유저 벡터를 사용해 LightGCN 모델 학습, 사용자의 관심사나 행동을 토대로 분류한 정보를 LightGCN 모델의 사용자 임베딩으로 사용함으로써 보다 개인화된 추천 시도
    - 새로운 유저 벡터 LDA model(Topic Modeling)을 통해 생성 가능
    - HIT@10: 0.092
- 모델 별 Hit@10

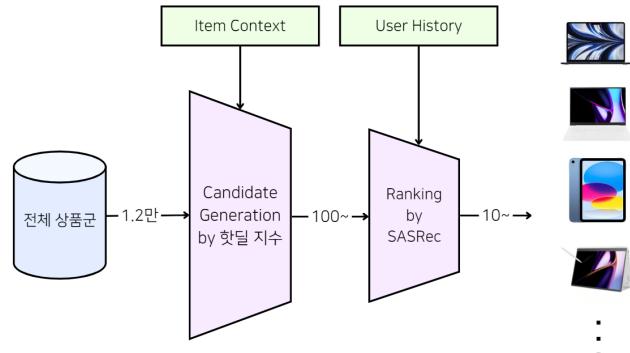
Model	Hit@10
Popularity	0.145
TF-IDF	0.343
Word2Vec	0.273
SASRec	0.292
TM2LGCN	0.092

## Final applied model

### Task1 - 키워드 알림 개인화 추천 모델

- 핫딜 지수를 통해 전체 상품군에서 키워드 알림이 갈만한 상품 후보군을 추림
- 핫딜 후보 상품 1개와 키워드를 등록한 유저의 상품 조회 이력을 SASRec/TFIDF의 input으로 넣어 아이템과 유저 간의 유사도를 구함
- 기존에 존재한 핫딜 지수와 유사도를 8:2 비율로 적용하여 키워드 알림의 발송 여부를 결정함

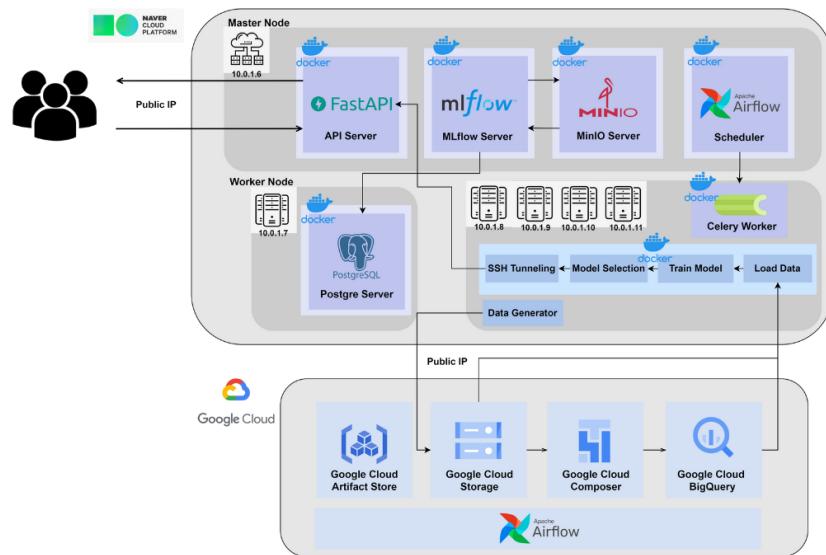
### Task2 - 세션 기반 Top K 추천 최종 모델



- 핫딜 지수를 이용해 전체 상품군에서 유저에게 추천될만한 100개의 item을 추림
- 비로그인 유저가 클릭할 때마다 발생한 세션을 SASRec의 input으로 넣어 유저가 좋아할 만한 아이템 Top K를 reranking 하여 보여줌
- user-free, 세션 기반, 유저가 클릭할 때마다 추천 아이템들을 갱신하여 보여줄 수 있는 모델을 구현

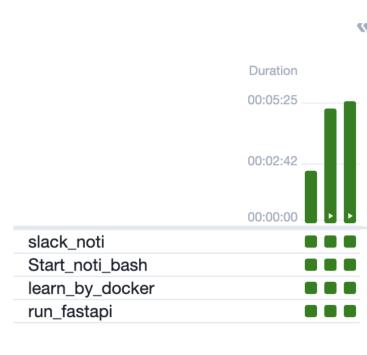
## Product Serving

### Service Architecture



## Airflow

- Slack\_noti
  - 학습 시작시 사용자에게 Slack 으로 알림
- Start\_noti\_bash
  - worker node의 live check
- learn\_by\_docker
  - google cloud artifact store 에서 training용도로 만들어진 docker image 다운로드 및 실행 및 mlflow로 취합, 이후 학습된 모델을 Google Cloud storage로 업로드
- run\_fastapi
  - google cloud artifact store 에서 training용도로 만들어진 docker image 다운로드 및 api 서버로 접근하여 실행



## Service API

- SASRec

The screenshot shows the Postman interface for the SASRec API. The request URL is `http://127.0.0.1:8000/predict/sasrec`. The request body contains JSON data: `{"user_id": "1", "product_ids": ["101", "102", "103", "104", "105"]}`. The response code is 200, and the response body is a JSON object with fields like `user_id`, `product_ids`, and `predicted_products`.

- TF-IDF

The screenshot shows the Postman interface for the TF-IDF API. The request URL is `http://127.0.0.1:8000/predict/tfidf`. The request body contains JSON data: `{"text": "Hello World! This is a test document."}`. The response code is 200, and the response body is a JSON object with fields like `text` and `tfidf_scores`.

## 후속 개발 및 연구

### Two-stage 모델 구현

- 서로 다른 특징을 학습할 수 있는 모델을 앙상블이 아닌 Two-stage로 구현하고자 함

### 도커 이미지 경량화

- 한정된 자원을 보다 효율적으로 활용할 수 있도록 도커 이미지 경량화를 시도하고자 함

### 쿠버네티스 활용

- 컨테이너를 쉽고 빠르게 배포 및 확장하고, 관리를 자동화할 수 있도록 쿠버네티스로의 전환을 도모

## 자체 평가 의견

### 잘 했던 점

- 데이터 웨어하우스를 구축해서 프로젝트 진행했다.
- 현실의 데이터를 활용한 프로젝트였다. 이에 데이터 정제 및 가공부터 product serving까지를 모두 경험하였다.
- 실제 웹사이트를 운영하는 개발자와 함께 협력하는 방법을 배웠다.

### 아쉬웠던 점

- 데이터 정제하는데 많은 시간을 쏟아서 다양한 실험하는데 충분한 시간이 부족했다.
- 더 좋은 모델이 있었지만, 속도를 위해 포기해야 할 부분이 존재하였다.
- Artifact store에 Docker Image를 업로드 하는 부분을 자동화 하고 싶었지만 조건이 달라서 어려움이 있다.

### 시도했으나 잘 되지 않았던 것들

- 키워드를 등록한 유저를 확인할 수 있는 로그가 한정되어 있어 그들만을 대상으로 정확한 EDA를 할 수 없었다.

### 프로젝트를 통해 배운 점 또한 시사점

- 실제 데이터는 학습용 데이터셋과 다르다.
- 클라우드 환경을 써보니 매우 편했다. 다만 비용이 많이 비싸서 도입시 충분히 고민해야 할 것 같다.
- 엔지니어와 리서쳐의 관점 차이를 이해할 수 있게 되었다.

## 1. 학습 목표 달성 방법

- **팀과 개인의 학습 목표:** 실제 웹 로그 데이터로 태스크에 맞는 논리적인 모델 설계, 서빙/평가 및 테스트 등 AI production 서빙의 전반적인 사항 경험, github 활용 등 협업 및 소통 능력 향상
- **개인 학습 측면의 접근:** 정제되지 않은 데이터를 분석하고 인사이트를 도출해 이에 맞는 모델 설계하기, 모델 예측값 서빙 API를 구현해 서버 상의 동작 이해하기
- **공동 학습의 중요성:** 분업 후 정기 회의에서 맡은 부분 리뷰, 모델 일지 작성, 회의록/협의사항 작성 등 정해진 규칙으로 원활한 협업

## 2. 모델 개선 방법

- **데이터 정제:** 실제 웹 로그 데이터에서 학습에 필요한 의미 있는 데이터를 추출하기 위한 정제 과정 수행
- **모델 선택과 학습:** 카카오 추천 팀 및 쿠팡에서 사용하는 Topic Modeling, Filtering+Boosting 기법을 적용하고자 시도했으나 데이터가 충분하지 않았음, 사용자의 상호작용 데이터를 중심으로 모델 학습 진행, Topic Modeling으로 생성된 user vector를 사용해 LightGCN 모델을 학습

## 3. 달성한 결과와 깨달음

- 실제 로그 데이터에서 의미 있는 정보를 추출하는 과정의 중요성, 이는 모델의 성능에 직접적인 영향을 끼침, 세션 기반 추천을 진행할 때 실제 로그 데이터에서 세션을 정의하는 기준을 정하기가 어려웠음
- 태스크에 맞는 모델 선정의 중요성, 실질적으로 어떻게 구현할 것인가를 구상하는 과정이 중요함, 비용적 측면/성능 등 실제 서비스에 적용 가능한 모델을 구상하는 과정은 대회 프로젝트와 관점이 다름을 느낌, 현업의 관점으로 어떻게 효율적으로 진행할 것인가 고려하게 됨, 서빙 API를 구현하고 모델 예측값을 서빙하는 과정에서 AI 모델이 적용되는 과정을 이해함
- 협업을 진행할 때 팀원 모두 각자 맡은 부분을 책임감 있게 수행하고 정기 회의에서 각자 리뷰를 진행해 수월하게 진행할 수 있었음, 협업과 소통의 중요성을 깨달음

## 4. 마주한 한계와 아쉬웠던 점

- 데이터를 정제하는 시간이 오래 걸려 다양한 모델을 실험하는 시간이 부족했음, 프로젝트에서 사용할 수 있는 데이터의 양이 제한적이어서 일부 모델을 적용하지 못한 아쉬움이 있음
- 모델의 속도와 비용적 측면을 고려하니 시도할 수 있는 모델이 제한적이었음

## 5. 미래 프로젝트를 위한 새로운 시도

- 정제된 충분한 데이터를 활용해 다양한 모델에 여러 기법을 적용해 최적의 조합 모색
- 모델 학습에 데이터가 들어가는 과정을 분석해 체화하고 싶음

# 김창영 T6042

## 학습 목표

- 데이터 엔지니어링 해보기
- ELT 데이터 파이프라인 설계해보기
- MLOps 및 FastAPI를 활용한 모델 서빙 API 개발해보기

## 프로젝트에서 시도한 것들

이전 프로젝트와 동일하게 깃허브를 활용해서 협업을 진행했다. 바뀐점이 있다면 기존 깃 사용 컨벤션을 팀원들이

### 1. 데이터 엔지니어링

- Nginx 로그에서 product\_id와 pids(쿠팡 user-item-vender 정보)를 추출했으며, 해당 데이터와 크롤노티에서 제공받은 상품 정보 조회 API를 활용해서 데이터를 수집했다.
- Nginx 로그에서 user\_device를 추출했으며, 유저정보조회API를 통해 유저 정보 데이터를 수집했다.

### 2. ELT 파이프라인 설계

- GCP Composer의 Airflow로 dag를 설계해서 ELT파이프라인을 설계해보았다.
- GCS에 업로드된 데이터들을 주기적으로 불러와 데이터 웨어하우스(BigQuery)에 적재하고 필요한 부분 혹은 통계치 요약 같이 데이터마트를 만들수 있도록 설계하였다.

### 3. MLOps 및 모델 서빙 API 개발

- 마키나락스에서 배포한 MLOps for MLE 참고해서 MLOps 파이프라인을 설계할 것을 제안했다.
- MLflow tracking server의 DB로 PostgreSQL 서버를 도커로 띄웠으며, MLflow 서버와 송신할 수 있도록 해주었다.
- 팀원들이 만든 모델 모듈들을 활용하여 FastAPI를 사용하여 모델 서빙 API를 개발했다.

## 느낌 점

크롤노티 측 DB에 직접 접근할 수 있었으면, 좀 더 수월하게 데이터 웨어하우스를 구축할 수 있었을 것 같은데 API를 통해 수집하다보니 번거로움이 존재했다. 또한 Nginx로그에서 데이터를 추출하고 분석하는데 팀 전체가 시간을 많이 할애한 것 같다. 이번 프로젝트때는 가급적 해본 적 없는 데이터파이프라인을 만들어보고 싶었는데 해당 부분을 구현해볼 수 있어서 재미있었다. 또한 네이버 클라우드와 구글 클라우드 환경을 사용해 봄으로써 클라우드 환경에 익숙해질 수 있는 계기가 된것 같다.

## 한계/교훈을 바탕으로 다음 프로젝트에서 시도해 볼 내용

유저 디바이스 정보가 등록된 유저들을 위한 개인화 추천시스템을 NCF를 실험해보았고 결과 또한 좋게 나왔으나, 다른 모델들과 비교실험이 필요하다고 판단했다. 다만 모델링보다 모델 서빙 API 개발이 팀 차원에서 우선이라 모델 개발보다는 API 개발에 초점을 맞췄다. 기회가 된다면 해당 부분을 이어서 진행해보면 좋을 것 같다.

## 나는 내 학습 목표를 달성하기 위해 무엇을 어떻게 했는가?

저번 Movie Recommendation 프로젝트와는 달리, 이번 최종 프로젝트에서의 목표는 현실의 데이터를 가공해 활용하는 것, 우리가 정의한 task에 적합한 모델링 하는 것이었다.

이를 위해 nginx 로그를 정규식으로 표현해보면서 우리가 원하는 csv 파일의 형태로 만드는 과정을 거쳤다. '키워드 알림 개인화'를 구현할 때, 개인적으로 가장 중요시 여긴 점은 유저의 상품 조회 기록과 핫딜 후보 상품이 novelty 등의 다양성이 없이 완전히 유사해야 한다는 부분이었다. 키워드에 속한 아이템 중 serendipity를 준답시고 유저의 조회 이력과는 많이 동떨어진 카테고리의 상품을 추천하는 것은 잘못된 추천이라고 생각했기 때문이다. 이를 위해 콘텐츠 기반 추천 중 하나인 자연어 처리 모델 TF-IDF와 Word2Vec을 구현했다. 이와는 반대로, '세션 기반 Top K 추천'에서는 유저가 그동안 조회해온 상품들과 완전히 유사한 아이템들을 추천해서는 안 된다고 생각하였다. Top K 추천의 목적은 유저가 추천된 상품들에 관심을 가지면서 서비스에 체류하는 시간을 늘리는 것이라고 생각하였기 때문에 다양성이 있는 목록을 제시하는 것이 중요하다고 판단하였다. 이에 같은 세션 안에 자주 존재한 아이템들끼리는 벡터상에서 가까이 위치하도록 하는 Item2Vec이라는 협업 필터링 모델을 구현했다.

## 전과 비교해서, 내가 새롭게 시도한 변화는 무엇이고, 어떤 효과가 있었는가?

이번 프로젝트에서는 데이터의 기간을 여러 경우로 나누어 실험하였다. 이전에 진행해온 book rating, movie recommendation 등의 대회에서는 모든 기간의 데이터를 다 사용하였다. 그러나 크롤노티의 경우, 취급하는 상품들이 빠르게 바뀌고, 신제품일수록 관심을 많이 받는 특징이 있었다. 이에 2023년 10월부터의 데이터를 모두 쓸 것이 아니라 일부 기간만을 모델에 사용하는 것이 더 좋은 성능을 보일 것이라는 가설을 세웠고 가장 최근의 데이터로부터 1주, 2주, 3주, 한 달 간격의 데이터를 가져와 실험하였다. TF-IDF로 모델을 고정하여 진행한 결과, Hit@10이 1주(0.620), 2주(0.607), 3주(0.599), 한 달(0.587) 순으로 높게 나왔다. 이를 통해 크롤노티의 사례에서는 데이터를 최신의 것을 활용하는 것이 훨씬 좋은 성능을 보임을 알게 되었다. 또한, 분석, 모델링하는 대상이 변화 할 때마다 실험을 통해 적절한 데이터의 기간을 정해야 함을 깨닫게 되었다.

## 마주한 한계는 무엇이며, 아쉬웠던 점은 무엇인가?

TF-IDF를 활용할 때, 사용할 수 있는 정보가 한정되어 있어 상품명만을 활용하였다. 따라서 다양성을 줄 수 없었기에 '세션 기반 Top K 추천'에서는 해당 모델을 사용할 수 없었다. 만일 대분류, 중분류 등의 값이나 상품 설명과 같은 내용이 더 있다면 TF-IDF를 활용했을 때 다양성이 포함된 결과를 줄 수 있었을 것 같다. 또한, 시간 상의 이유로 Item2Vec 모델에 있어 경량화 등의 개선을 하지 못한 점이 아쉽다. 그리고 FastAPI 등 serving 단계의 지식이 완전히 체화되지 않아 이번 프로젝트에서 다뤄보지 못해 아쉽다.

## 한계/교훈을 바탕으로 다음 프로젝트에서 시도해보고 싶은 점은 무엇인가?

포스트세션 기간 동안 Item2Vec을 새로운 아이템에 대해 대응할 수 있도록 모델을 경량화 하여 빠른 속도로 학습할 수 있도록 모델을 수정해볼 계획이며 inference 및 성능지표 확인까지 구현해보고자 한다. 또한, 이를 FastAPI로 구현, docker image로 생성해 현재 만든 프로젝트에 적용할 수 있도록 만들어볼 예정이다. 그리고 Item2Vec에서 파생된 벡터를 이용한 다른 모델을 구상하고 구현해보고 싶다.

## 전민서 T6151

### 【프로젝트 과정 속의 나의 학습 목표】

- 실제 로그 데이터를 이용하여 사용할 수 있는 형태의 완성된 결과물을 만들어보기
- 문제를 정의하는 방법을 배우기
- 배운 프로젝트 서빙 지식을 바탕으로 배치 서빙 아키텍처 만들기
- 엔지니어와 리서쳐의 관점을 둘 다 느껴보기

### 【학습 목표를 이루기 위해 노력한 점들】

- 주어진 로그 데이터를 분석하여 사용 가능한 정보와, 사용 불가능한 정보를 분류하기

사용한 로그 데이터는 Nginx에 저장된 유저의 접속 기록을 바탕으로 만들어진 데이터였다. 이 로그 데이터에는 유저가 서버에 요청한 **get**, **post** 데이터, **ip**기록, 사용한 브라우저 등이 남아있었는데 이중 유저의 **get**정보에서 아이템id를 분리했고, 유저의 **ip**를 이용하여 유저를 특정하였으며 특정 유저가 같은 아이템을 지속적으로 조회한 경우 해당 유저의 로그를 사용하지 않는 등 전처리에 대한 처리를 진행하였다.

- 초기 성능을 확인한 후 가능성이 없는 모델은 빠르게 사용하지 않기

처음에 키워드 추천과, 세션기반 추천에 대해서 어떤 방식으로 문제를 접근할지에 대한 고민이 많았다. 유저-아이템 상호작용 데이터를 시계열 데이터로 확인하고 처리하는 방식이 존재하고, 혹은 **MF** 방식으로 처리하는 방식도 존재한다. 이전 경험으로 보아 시계열 데이터가 아닌 **MF**방식도 충분히 좋은 성능을 가졌던 기억이 있다. 또한 그래프를 이용한 방법등도 존재하지만, 그래프를 이용하는 방식은 유저 정보가 없다면 사용할 수 없는 문제점이 있어서 사용하지 않았다. 최종적으로는 **TF-IDF**와 **SASRec** 모델을 선택하였는데 이는 **SASRec**은 유저의 **item** 조회기록만을 바탕으로 추론할 수 있어 유저 임베딩을 사전에 학습 시키지 않더라도 즉각적으로 사용할 수 있기 때문이다. 또한 모델들의 학습 시간을 사용가능성에 크게 반영하였다.

- 확장 가능하며, 유연한 파이프라인 구축

구글 클라우드를 적극적으로 사용하여 추후 확장성을 고려하였다.

지금 환경에서는 전부 네이버 클라우드 플랫폼에서 동작하지만 이후 API의 실행은 외부에 존재하는 서버에서 동작하고 학습만 클라우드 시스템에서 동작하는등 변화가 존재할 수 있다고 예측하였다. 그래서 **GPU** 클러스터 내부에서 만들어진 가중치 파일과 데이터 파일등을 전부 구글 스토리지에 적재하여 필요한 곳에서 사용할 수 있도록 구성하였으며, **Airflow**의 각 단계마다 **Docker**를 이용해 격리하여 새로운 모델을 사용할 경우에도, 해당 모델에 맞는 도커 이미지만 생성하여 구글 아티팩트 스토어에 업로드 해 준다면 바로 사용할 수 있도록 구성하였다.

### 【프로젝트를 통해 배운 점과 변화한 점】

- 가지고 있는 정보에 따라서 적용할 수 있는 모델의 한계가 존재한다.

- 오프라인 테스트를 통해 모델을 평가하는 것은 신뢰도에 문제가 있다.

- 확장성을 고려하고 파이프라인을 설계하는 것은 어려운 작업이다.

### 【프로젝트 중 마주친 한계】

처음부터 추천시스템을 고려하거나, 2차가공을 고려한 데이터가 아니기 때문에 로그를 정제하는 단계도 힘들었지만, 유저 데이터의 부재, 아이템 데이터의 부족등 여러가지 문제점이 있었다.

그렇기에 적용할 수 있는 모델의 한계도 존재하였다. 또한 유저가 봤다는 사실 하나로 긍정이라고 판단하였기 때문에 좋은 모델인지에 대한 의문도 남는다.

### 【다음 프로젝트가 있다면?】

지금 만들어진 모델과 파이프라인을 바탕으로 A/B 테스트를 진행 해 보고 싶다.

오프라인 테스트를 이용해 만들어진 점수는 신뢰하기에 조금 무리가 있는 것 같다.

**kubernetes**위에 **airflow**를 사용하는 방식으로 변경하고 싶다.

## Final Wrap Up Report

한대희\_T6179

### 나는 내 학습목표 달성을 위해 무엇을 어떻게 했는가?

- 원본 로그 파일을 처리하기 위해 파일 입출력과 정규 표현식을 활용하여 필요한 데이터를 추출하고 DataFrame 을 생성하였습니다. 이후 url 명세표와 데이터의 분포 및 특성을 파악하여 각 변수들 간의 관계를 분석하였습니다. 이를 통해 데이터의 형식이나 누락된 값 등의 오류를 발견하고 처리하는 과정에서 로그 데이터 분석 및 처리에 대한 실전 경험을 쌓았습니다.
- 크롤노티 측에서 제공한 상품 정보 API 를 활용하여 상품 아이템에 대한 데이터를 수집하고, 이를 기반으로 상품의 side information 을 생성하였습니다. 이 과정에서 API 호출 및 데이터 파싱 기술을 연마하고, 상품 정보 데이터를 정제하고 가공하는 방법을 습득하였습니다.

### 나는 어떤 방식으로 모델을 개선했는가?

- 데이터 분석 과정에서 원본 로그 데이터와 중복된 로그 데이터의 HTTP 응답 코드 분포를 비교한 결과, 중복된 로그 데이터에서 'Client error response' (499 응답 코드)와 같은 특정 에러 응답들이 원본 로그 데이터에 비해 상대적으로 높은 비율로 나타난다는 것을 발견하였습니다.
- 이를 해결하기 위해 웹 요청 및 응답 프로세스 중 오류, 데이터 수집 시스템의 이중 기록으로 인하여 중복되는 로그 데이터들을 제거하였습니다. 또한, 일정 시간(예: 10 초) 내에 3 회 이상 연속적으로 상호작용하는 경우를 어뷰징 데이터로 간주하고 제거함으로써, 상품 페이지 링크에 짧은 시간 동안 비정상적으로 반복해 접근하는 행위를 방지하였습니다. 이를 통해 과적합 방지 및 일반화 성능 향상 뿐만 아니라 학습 시간 단축 및 실제 사용자 행동을 더욱 정확하게 반영할 수 있었습니다.
- Cosine similarity 를 이용하여 사용자 간 유사도 행렬을 생성하고, 입력 사용자와 가장 유사한 상위 3 명의 사용자를 찾아내었습니다. 이후 유사 사용자들의 인터랙션이 높은 아이템들 중 입력 사용자가 아직 상호작용하지 않은 아이템들을 추천하는 User-based Collaborative Filtering 모델을 개발하였습니다. 이를 통해 입력 사용자의 선호도와 유사한 사용자들의 선호도를 기반으로 다양한 상품을 추천할 수 있었습니다.
- 또한, Cosine similarity 를 활용하여 아이템 간 유사도 행렬을 생성하고, 입력 아이템과 유사도가 높은 아이템을 상위 n 개 추천하는 Item-based Collaborative Filtering 모델을 개발하였습니다. 이를 통해 입력 아이템과 이미 상호작용한 아이템들 중 유사한 상품을 추천함으로써, 유저에게 다양한 상품을 추천할 수 있었습니다.

### 마주한 한계 혹은 아쉬웠던 점은 무엇이며, 한계/교훈을 바탕으로 다음 프로젝트에서 시도해볼 것은 무엇인가?

- 크롤노티와의 협업을 통해 개선된 모델을 개발하여 온라인 서비스에 적용하고, 온라인 A/B 테스트 및 비즈니스적 평가를 진행하고자 합니다.

### [프로젝트 과정 속의 목표]

- AI Production과 Serving의 전반을 경험해보자.
- 데이터 분석 결과를 바탕으로 대시보드를 만들어보자.
- 크롤노티 로그 데이터를 바탕으로 task를 정의하고, 이를 달성하기 위한 모델링을 해보자.

### [목표를 이루기 위해 노력한 점들]

#### - 데이터 분석을 통한 task 정의

크롤노티 측에서 받은 로그 데이터를 정제/분석하고 이를 바탕으로 크롤노티 서비스에 맞는 추천 시스템을 개발하는 task를 정의하였다. Raw data를 정제한 뒤, 유저와 아이템 간의 interaction이라고 볼 수 있는 로그만을 정제하여 해당 내용을 분석하고, 모두가 확인할 수 있도록 Looker Studio를 활용하여 시각화해 보았다. 또한 크롤노티 서비스의 특성을 고려하여 “키워드 알림의 개인화”, “세션 기반 Top K 추천”이라는 두 task를 정의하고 각각에 맞는 모델을 개발하였다.

#### - SASRec 모델 구현 및 MLflow 실험

세션 기반 Top K 추천은, 사용자의 아이템 상호작용 이력을 바탕으로 사용자가 보고 싶어할 아이템을 추천하여 추천 배너를 띄우는 task이었다. 사용자의 실시간 선호도를 반영하기 위해서는 data를 sequential하게 볼 필요가 있다고 판단하였고, 이에 sequential recommendation model인 SASRec 모델을 만들어 사용자의 아이템 조회 이력 10개 이하를 입력하였을 때, 후보 아이템들의 선호도를 0에서 1 사이의 값으로 나타내고, 이를 바탕으로 Top K 추천을 하도록 하였다. 또한, SASRec 모델을 학습할 때 매 epoch 마다의 loss 값, 그리고 evaluation 데이터에 대한 Hit@10의 값 및 모델을 저장하여 실험을 관리할 수 있도록 MLflow를 연결하였다.

#### - 깃허브 이슈 기반 협업

팀원들과 깃허브 활용에 대한 규칙을 정하고, 이를 지키면서 깃 사용에 익숙해질 수 있었다. 특히, 이슈를 생성하고, 칸반 보드를 활용하여 자신이 진행 상황과 앞으로의 계획을 공유하면서 더욱 효율적으로 프로젝트를 진행할 수 있었다.

### [프로젝트를 통해 배운 점과 변화한 점]

#### - AI production의 과정에 대한 이해

지난 세 개의 대회형 프로젝트와는 다르게, 이번에는 데이터를 구하고 이로부터 task를 정의하여 모델을 만들고 API 서빙을 하는 데까지 AI production 전반을 배울 수 있는 기회였다. 이전의 프로젝트에서는 모델의 성능 향상이라는 뚜렷한 목표를 가지고 있었으나, 최종 프로젝트에서는 데이터를 구하는 데에서부터의 모든 과정을 직접 수행하고, 이를 실제 서비스에 도입하기 위한 고려를 해봄으로서 많은 것을 배울 수 있었다. 또한 그러한 과정에서 Google Cloud 기반 Big Query와 Storage, FastAPI, Docker 등 다양한 도구를 활용해 보면서 많은 것을 배우고, 또 앞으로 어떤 것을 공부해 나가야 할지 생각해보는 시간이 되었다.

#### - 협업을 통해 일의 효율과 동시의 나의 능력을 키울 수 있다는 점

깃허브를 통한 협업을 통해 프로젝트 진행 속도를 높인 것뿐만 아니라, 팀원들과 함께 데이터를 분석하고, task를 수행하면서 실전에서의 AI 활용에 대한 간접적인 체험을 해볼 수 있었다.

### [프로젝트 중 마주친 한계]

- 크롤노티로부터 제공받은 로그 데이터의 한계로 CTR 등의 지표를 확인할 수 없어 오프라인 성능을 Hit@10, ndcg@10 등으로 정의할 수밖에 없던 점이 아쉬웠다.
- 4주라는 짧은 시간 안에 프로젝트가 완성되어야 했는데, 데이터 정제와 분석에 시간이 많이 들어 모델 관련하여 다양한 실험을 해보지 못한 점이 아쉬웠다.

### [앞으로 개선할 점]

- 서빙 관련해서도 공부를 더 많이 할 필요를 느꼈다.
- 데이터 분석을 위한 SQL 문법 공부를 더 해보자.