

## Project #2: DB mining & Automated Recommendation System

본 프로젝트는 추가로 제공된 데이터를 바탕으로 프로젝트 #1의 방법으로 DB를 구축하고 이를 이용하여 DB mining 및 Automated Recommendation System 구현을 목적으로 한다. 본 프로젝트는 크게 세 부분으로 나뉘며 Python과 MySQL을 사용하여 구현하여야 한다.

### PART I. 의사결정나무

### PART II. 연관분석

### PART III. 추천시스템

프로젝트 #2 수행에 앞서 이를 위한 데이터베이스 구축을 할 때 각 table의 column들의 순서, 자료형, 제약은 첨부한 예시 코드를 따르도록 한다. 데이터는 프로젝트 #2에서 새로 첨부한 데이터를 insert하도록 한다.

### PART I. 의사결정나무

PART I는 사이트 A에서 BEST item으로 선정한 아이템 기준에 대한 의사결정나무를 만드는 것을 목표로한다. R1-1과 R1-2에서는 의사결정나무를 구성할 데이터를 만들기 위해 mysql-connector-python을 사용해서 query를 실행하는 것을, R1-3에서는 python의 sklearn 라이브러리를 사용해서 의사결정나무를 만들고 graphviz 라이브러리를 사용해서 결과를 시각화하는 것을 수행한다. R1-4에서는 R1-3에서 생성한 의사결정나무와는 다른 BEST item 선정 여부에 대한 의사결정나무를 만들어보는 것을 목표로 한다.

#### (R1-1)

사이트 A는 사이트 유지 및 활성화에 기여하는 아이템들을 BEST item으로 선정한다. 이 사이트의 BEST item 선정 기준을 파악해보고자 한다. 먼저 주어진 best\_item\_list.txt를 반영하기 위해 DMA\_project2의 item 테이블에 best\_item이라는 새로운 column을 추가한다. 이 때, best\_item의 데이터형은 TINYINT(1)이며 Default 값은 0이다. 만약 아이템의 id가 best\_item\_list.txt에 포함된다면 해당 아이템의 best\_item의 값에 1을 저장한다.

#### (R1-2) 결과물: DMA\_project2\_team##\_part1.csv

다음과 같은 column이 포함되는 결과를 반환하는 쿼리를 작성하라. 단, Nested query를 이용하여 하나의 SQL 문장으로 작성하여야 한다. 결과는 DMA\_project2\_team##\_part1.csv에 저장되어야 한다.

- id: 아이템의 id
- best\_item: 아이템의 BEST item 선정 여부
- ratings: 아이템이 사용자들에게 받은 평가 점수 ( [-4, +4] 범위의 값, 0은 중립 )
- num\_of\_specs: 아이템이 가진 스펙의 수
- num\_of\_tags: 아이템에 붙어 있는 태그의 수
- num\_of\_users: 아이템을 이용한 이력이 있는 사용자의 수
- avg\_usage\_time: 아이템을 이용한 사용자들의 인당 평균 이용 시간 ( 사용자 한 명의 이용 시간은 가입 이래 총 이용 시간 )
- num\_of\_reviews: 아이템에 작성된 리뷰의 수
- sum\_of\_recommend: 아이템에 작성된 리뷰의 recommend 총합 ( -1도 포함 )
- avg\_review\_len: 아이템에 작성된 리뷰의 평균 본문 길이

(R1-3) 결과물: DMA\_project2\_team##\_part1\_gini.pdf, DMA\_project2\_team##\_part1\_entropy.pdf

R1-2에서 반환 받은 결과로부터 BEST 아이템 선정 기준에 대한 의사결정나무를 생성하고자 한다. Node impurity 측정 방식을 gini와 entropy 두 가지로 하여 각각 의사결정 나무를 생성하며, 그 결과는 graphviz를 통해 저장되어야 한다. 본 문제에서 만드는 의사결정나무의 속성은 아래와 같다. 보고서에는 이를 바탕으로 만들어진 의사결정나무를 간단히 분석하는 내용이 포함되어야 한다.

- 사용 라이브러리: sklearn.tree.DecisionTreeClassifier
- Node impurity criterion: gini / entropy
- 결과 파일명: node impurity criterion에 따라 DMA\_project2\_team##\_part1\_gini.pdf, DMA\_project2\_team##\_part1\_entropy.pdf 로 구분함.
- 분석 목표: BEST 아이템 선정 기준
- min\_samples\_leaf: 10
- max\_depth: 5
- feature names: ratings, num\_of\_specs, num\_of\_tags, num\_of\_users, avg\_usage\_time, num\_of\_reviews, sum\_of\_recommend, avg\_review\_len
- class names: normal, BEST

(R1-4)

R1-3에서 생성한 의사결정나무와는 다른 BEST 아이템 선정 기준에 대한 의사결정나무를 만들고 R1-3에서 생성한 의사결정나무와 비교하라. R1-3 외에 다른 input feature를 추가로 사용하거나 제거할 수도 있고, 의사결정나무의 다른 속성을 변경할 수도 있다. 보고서에는 다음과 같은 내용이 포함되어야 한다.

- 의사결정나무에 사용된 input features
- Node impurity criterion
- sklearn.tree.DecisionTreeClassifier에 입력한 속성 (ex. max\_depth=3)

## PART II. 연관분석

PART II 연관분석에서는 사이트 A의 bundle들간의 연관 분석을 목표로 한다. R2-1과 R2-2에서는 연관분석과 추천시스템을 위한 view 생성을 위해 mysql-connector-python을 사용하여 query를 실행하는 것을, R2-3에서는 R2-2의 view에서 연관분석을 위한 horizontal table의 결과를 반환하는 것을 목표로 하며, R2-4에서는 python의 mlxtend 라이브러리를 사용하여 연관분석을 수행하고 결과를 출력하는 것을 목표로 한다.

(R2-1) 결과물: DMA\_project2\_team##\_part2\_bundle.csv

본 연관분석에서는 포함된 아이템의 수, 관련된 장르의 수, 포함된 아이템을 이용한 사용자의 수가 많은 상위 30개의 bundle에 대해 진행하고자 한다. 분석을 위해 하나의 SQL 문장으로 아래의 column들이 포함되는 bundle\_score라는 이름을 가지는 view를 생성하라.

- bundle\_id: 번들의 id
- bundle\_name: 번들의 name
- num\_item: 해당 번들에 포함된 아이템 수\*100
- num\_genre: 해당 번들과 관련된 장르 수\*100
- num\_user: 해당 번들에 포함된 아이템을 이용한 이력이 있는 사용자 수
- score: num\_item+num\_genre+num\_user 의 값

이 때, score를 기준으로 내림차순 정렬되어야 하며 이를 기준으로 상위 30개만을 view에 저장해야 한다. 그리고 해당 view를 DMA\_project2\_team##\_part2\_bundle.csv에 저장하라.

(R2-2) 결과물: DMA\_project2\_team##\_part2\_UBR.csv

연관분석과 추천시스템을 위해 각 user가 bundle에 대해 가지는 관심 정도를 rating으로 정의하고자 한다. 이는 아래와 같이 정의된다. 단, 사용자가 추천/리뷰/이용 활동을 전혀 하지 않은 아이템만이 포함된 bundle의 경우 0점이 아니라 rating을 하지 않은 것으로 한다. 여기서 '추천'은 사용자가 아이템에 남긴 review에서 recommend 정보가 1인 것을 의미한다.

### 〈Rating Equation〉

$$\text{rating}(\text{user}, \text{bundle}) = 5 * (\text{user가 해당 bundle에 포함된 아이템 중 추천한 아이템 개수}) \\ + \min(\text{user가 이용한 이력이 있는 item이 해당 bundle에 포함된 개수}, 5)$$

예를 들어, a라는 사용자가 bundle\_b에 포함된 item\_c를 추천했고 bundle\_b에 포함된 item 10개에 대해 이용 이력이 존재한다면  $\text{rating}(\text{user\_a}, \text{bundle\_b}) = 5 * 1 + \min(10, 5)$  로 해당 bundle에 10이라는 관심 정도를 가진 것이다. 또 d라는 사용자가 bundle\_b에 포함된 item에 추천을 남긴 리뷰는 없으나 이에 포함되는 4개의 item에 대해 이용 이력이 존재한다면  $\text{rating}(\text{user\_d}, \text{bundle\_b}) = 5 * 0 + \min(4, 5)$  로 해당 bundle에 4라는 관심 정도를 가진 것이다.

R2-1에서 정의한 총 30개의 bundle에 대해 user들의 rating 정보를 user\_bundle\_rating 이라는 이름의 view로 생성하여야 한다. 하나의 SQL 문장을 이용하고 이 때 bundle은 R2-1의 총 30개에 대해서만 저장되어야 하며, rating 정보가 있는 user-bundle set에 대해서만 저장해야 한다.

아래의 column들을 포함하는 **user\_bundle\_rating** 이라는 이름을 가지는 view를 생성하라.

- user: user\_id
- bundle: 번들 이름 ( 상위 30개 중 하나 )
- rating: 위에서 정의한 user가 bundle에 가지는 관심 정도 ( >= 1 )

위의 view를 만든 후, 이 view 안에서 20개 이상의 rating 정보를 가진 user들에 대한 정보만을 남긴 **partial\_user\_bundle\_rating** 이라는 이름을 가진 view를 생성하라. partial\_user\_bundle\_rating이 가지는 column은 user\_bundle\_rating과 동일해야 한다. 이 partial\_user\_bundle\_rating view를 DMA\_project2\_team##\_part2\_UBR.csv에 저장하라.

#### (R2-3) 결과물: DMA\_project2\_team##\_part2\_horizontal.pkl

연관분석을 위해 vertical table 형태의 partial\_user\_bundle\_rating을 horizontal table로 만든 결과를 pandas의 DataFrame으로 저장하라. DataFrame은 user id를 index로 가져야 하며, bundle name들을 column 명으로 가져야 한다. partial\_user\_bundle\_rating에 해당 user의 해당 bundle에 대한 rating 정보가 있다면 1, 없다면 0을 저장해야 한다. 저장된 DataFrame의 각 user는 연관분석의 transaction 역할을, 각 bundle은 연관분석의 item 역할을 하게 된다.

이 때 horizontal table로 만들기 위해 sql query문을 사용하여도 되고, pandas 라이브러리를 사용하여도 된다. 해당 horizontal table을 DMA\_project2\_team##\_part2\_horizontal.pkl에 저장하라.

#### (R2-4) 결과물: DMA\_project2\_team##\_part2\_association.pkl

R2-3에서 만든 DataFrame을 사용해서 다음의 조건을 만족하는 frequent itemset을 만들고 연관분석을 수행하라. 그리고 이 결과에 대한 간략한 정성적, 정량적 평가를 수행하라.

- Frequent itemset의 최소 support: 0.35
- 연관분석 metric: lift ( lift >= 2 인 것들을 출력 )
- 결과 파일명: DMA\_project2\_team##\_part2\_association.pkl

## PART Ⅲ. 추천시스템

PART Ⅲ 추천시스템에서는 사용자에게 bundle을 추천하는 추천시스템 구현을 목적으로 한다. R3-1에서는 점수 예측 결과를 반환하는 함수 작성을, R3-2부터 R3-4까지는 recommendation system을 구현하도록 한다. 이 때 라이브러리로는 surprise를 이용한다.

### (R3-1)

점수 예측 결과 top-n개 결과를 반환하는 get\_top\_n 함수를 작성하라. 세부 내용은 때대 코드를 참고하여 작성하도록 한다.

### (R3-2) User-based Recommendation 결과물: 3-2-1.txt, 3-2-2.txt

['8051826169', '8027368512', '7998746368', '8054453794', '8030770479'] 의 총 5명 user에 대하여 다음의 알고리즘과 유사도 함수를 사용한 추천 결과 top-5 bundle을 텍스트 파일로 출력하라.

- 알고리즘 : KNNBasic      유사도: cosine      파일명: 3-2-1.txt
- 알고리즘 : KNNWithMeans      유사도: pearson      파일명: 3-2-2.txt

또한, User-based recommendation에서 다양한 알고리즘과 유사도 함수를 적용해보고 cross validation(k=5, random\_state=0)을 기준으로 가장 좋은 성능을 보이는 모델을 제출하라.

### (R3-3) Item-based Recommendation 결과물: 3-3-1.txt, 3-3-2.txt

['World of Magicka Bundle', 'Borderlands Triple Pack', 'Tripwire Complete Bundle', 'Grand Theft Auto V & Great White Shark Cash Card', 'Killing Floor 1 Complete Your Set!'] 의 총 5개 bundle에 대해 다음의 알고리즘과 유사도 함수를 사용한 추천 결과 top-10 user를 텍스트 파일로 출력하라.

- 알고리즘 : KNNBasic      유사도: cosine      파일명: 3-3-1.txt
- 알고리즘 : KNNWithMeans      유사도: pearson      파일명: 3-3-2.txt

또한, Item-based recommendation에서 다양한 알고리즘과 유사도 함수를 적용해보고 cross validation(k=5, random\_state=0)을 기준으로 가장 좋은 성능을 보이는 모델을 제출하라.

### (R3-4) Matrix-based Recommendation 결과물: 3-4-1.txt, 3-4-2.txt, 3-4-3.txt, 3-4-4.txt

['8051826169', '8027368512', '7998746368', '8054453794', '8030770479'] 의 총 5명 user에 대하여 다음의 알고리즘을 사용한 추천 결과 top-5 bundle을 텍스트 파일로 출력하라.

- SVD(n\_factors=100, n\_epoch=50, biased=False)      파일명: 3-4-1.txt
- SVD(n\_factors=200, n\_epoch=100, biased=True)      파일명: 3-4-2.txt
- SVD++(n\_factors=100, n\_epoch=50)      파일명: 3-4-3.txt
- SVD++(n\_factors=100, n\_epoch=100)      파일명: 3-4-4.txt

또한, Matrix-based recommendation에서 다양한 알고리즘과 유사도 함수를 적용해보고 cross validation(k=5, random\_state=0)을 기준으로 가장 좋은 성능을 보이는 모델을 제출하라.

### 채점 기준(절대평가)

- PART I (30%): 각각 5%, 10%, 10%, 5%
- PART II (30%): 각각 5%, 10%, 5%, 10%
- PART III (30%): 각 추천 결과 파일 2%, 각 cv 기준 best model 3%
- 보고서 품질(5%), 발표(5%)

결과물들을 'DMA\_project2\_team##\_zip'파일로 압축하여 발표일 전날인 5월 10일 23:59까지 ETL에 업로드해야 한다. ETL 상에 문제가 생겼을 경우 [osa8361@snu.ac.kr](mailto:osa8361@snu.ac.kr) 로 오류 증명 파일과 함께 해당 일시까지 보내야 한다. 제출해야 할 결과물과 파일명, 파일 확장자는 다음과 같다.

- 보고서 (20페이지 이내)

- 파일명: DMA\_project2\_team##\_보고서.pdf
- R1-4에 대해 수행한 결과는 python 코드에는 포함되지 않아도 되지만, 보고서에는 그에 대한 내용이 있어야 한다.

- 발표 자료 및 발표 동영상 (5분 이내)

- 발표 자료 파일명: DMA\_project2\_team##\_발표자료.pdf
- 발표 동영상 파일명: DMA\_project2\_team##\_발표동영상.mp4
- 발표 동영상은 팀 당 5분 이내로 제작되어야 하며 powerpoint의 녹화기능을 사용한다.

- Python 프로그램 코드

- Python 코드 파일명: DMA\_project2\_team##\_py
- 함수들의 입력 값들의 의미는 다음과 같다.

host, user, password: MySQL에 접근하기 위한 계정 정보

schema: schema 이름

- 뼈대 코드의 주석에 작성된 TODO들에 따라 팀의 번호, MySQL 계정 정보 등을 바꿔야 한다.
- R1-2, R2-1, R2-2의 SQL query 문장은 Nested query를 이용하여 하나의 SQL 문장으로 작성하여야 한다.
- 사용 라이브러리는 mysql-connector-python, pandas, numpy, collections, sklearn, mlxtend, surprise, graphviz를 기반으로 한다. 필요하다면 다른 라이브러리를 사용할 수 있으나 이에 대해 [osa8361@snu.ac.kr](mailto:osa8361@snu.ac.kr)로 사전에 메일을 보내야 한다.

- 그 외 결과물

- DMA\_project2\_team##\_part1.csv
- DMA\_project2\_team##\_part1\_gini.pdf
- DMA\_project2\_team##\_part1\_entropy.pdf
- DMA\_project2\_team##\_part2\_category.csv
- DMA\_project2\_team##\_part2\_UBR.csv

- DMA\_project2\_team##\_part2\_horizontal.pkl
- DMA\_project2\_team##\_part2\_association.pkl
- 3-2-1.txt, 3-2-2.txt, 3-3-1.txt, 3-3-2.txt, 3-4-1.txt, 3-4-2.txt, 3-4-3.txt, 3-4-4.txt
- csv 파일은 column들 사이에 분리 기호는 콤마(,)여야 하며 row들 사이에는 줄 넘김 (\\n)으로 구분되어야 한다. Query 실행 결과 또는 view에 대한 csv를 저장할 땐 workbench 상에서 저장하는 것이 아니라 python 코드를 통해 저장을 수행하여야 한다.
- Pandas의 DataFrame을 pickle로 저장할 때는 to\_pickle 함수를 이용하여 저장한다.