Daeun Sarah Kim

2017.12.07

<Ⅰ> 프로젝트 개관

1. 초록

This project aims for creating a Market-Basket-Model-based food recommender using Fine Food Reviews offered by Kaggle. Market Basket Model is the expression of data sets adopting the concepts of Market, Basket and Item. A relationship discovered from each set of items can be applied to relationships founded from other ones. In the case of data from Kaggle, most of users had written only one review and most of products have only one review, which means there are small number of meaningful baskets. It means number of recommendations would be small. So it is expected to have low recall. Most reviews are positively biased so it seems that once recommending food, the value of precision would be high. Among 568454 number of reviews, outliers having too small or too big baskets are excluded and others are distributed into training data and test data with a probability of 0.90. 0.97 of test data compose baskets and others are judged whether they would be recommended or not. The Market-Basket-Model-based test using the processed data sets made a result of precision of 0.976 and recall of 0.621. It supports the previous assumption that a recall is influenced by the average size of baskets and a precision is affected by a bias of data. This will help select data to be used and improve efficiency and accuracy of a recommendation system.

본 프로젝트에서는 Kaggle에서 제공되는 Food Review 데이터를 활용하여 Market Basket Model 기반의 Food Recommender를 만드는 것을 목표로 한다. Market Basket Model은 data set을 현실에 있는 Market과 Basket, 그리고 Item으로 표현한 것이다. 각 Item 집합에서 연관 관계를 발견하고 그 관계를 통해 다른 Item과의 관계를 연관 지을 수 있다. Kaggle의 Food Review는 각 User 또는 각 Product에 대해 해당하는 Review가 대부분 1개로 학습에 의미 있는 Basket을 만들기 어려워 Recommendation의 횟수가 적으며 따라서 Recall이 낮을 것이라 예상할 수 있었다. 또한 대부분의 Review가 긍정적인 평가에 치우쳐져 있어 일단 Recommend할 경우 Precision 수준은 높을 것이라 예상할 수 있었다. 총 568454개의 데이터 중 basket의 크기가 지나치게 크거나 작은 outlier를 제외시키고 0.90의 확률로 data를 training 및 testing 용으로 나누었다. 또한 testing 데이터에서도 0.97의 확률로 basket을 구성할 데이터와 recommendation 여부를 판단할 data로 나누었다. 위와 같은 규칙으로 가공된 데이터를 Market Basket Model을 통해 학습 및 테스트하여 Precision은 0.976, Recall은 0.621의 결과를 얻었다. 그리고 그 결과는 basket의 평균적인 크기에 recall이 영향을 받을 것이며, Review의 편향에 Precision이 영향을 받을 것이라는 추측을 지지한다. 그리고 이러한 자료는 Recommendation을 위한 시스템 개발 시, 사용할 데이터를 선별하고 시스템의 효율성 및 정확성을 높이는 데 사용될 수 있을 것이다.

<Ⅱ>서론

1. 프로그램 순서도

프로그램 순서도는 다음 <그림 1>과 같다.

1. ‘Amazon fine food Reviews’ data가 들어있는 Reviews.csv 파일을 데이터 변환 프로그램인 Data\_Parser에 넣는다.
2. Data\_Parser를 통해 Reviews.csv파일을 trainig.csv과 testing.csv로 변환한다.
3. 변한된 파일과 market-basket모델링 프로그램인 Recommender[[1]](#footnote-1)를 통해 결론을 도출한다.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Reviews.csv | Data\_Parser  (2) | training.csv  testing.csv | Recommender  (3) | Result |
| (1) |  |  |

<그림 1>

2. Recommender

1) Market-Basket Model

(1) Definition

Market은 다양한 Item을 가지고 있으며 Basket은 Market의 부분집합을 가지고 있다. 각 Basket에는 한 Item i가 나타난다면 필수적으로 나타나는 아이템 j가 있는 경우가 있다. 이러한 형태의 Item과 Basket, Market집합을 Association Rule[[2]](#footnote-2)과 A-priori Algorithm[[3]](#footnote-3)으로 data를 분석한다.

(2) 활용

다음 <그림 2>은 가장 큰 집합인 Market 속, 가장 작은 원으로 표현된 Item과 그 Item을 둘러싼 원으로 표현된 Basket 집합 사이의 연관 관계를 표현하고 있다.



<그림 2>

현재 프로그램에서는 Hash Set을 통해 Basket을 구현하였고 각 Basket은 다시 Tree Map을 통해 Baskets라는 집합으로 묶이게 된다.

Recommender는 이러한 Itemset을 A-priori Algorithm으로 분석하여 집합을 분류한다. 이때 Pass단계는 3단계이다.

2) Confusion Matrix

(1) Definition

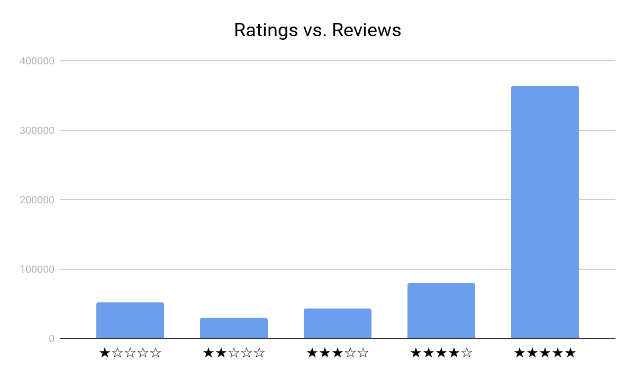
1, 0으로 표현되는 True, False를 통해 실제와 예측이 어느 정도 비율로 일치하는지 표현한다. Precision은 1이라고 예측한 것 중 실제 값이 1인 것의 비율을 나타낸다. Recall은 실제 값이 1인 것 중 1이라고 예측한 것의 비율을 나타낸다. Accuracy는 전체중에서 올바르게 예측한 것의 비율을 나타낸다. 각 값은 1에 가까울수록 긍정적이다.

(2) 활용

Recommender는 training data를 통해 학습한 이후 testing data속 c라는 정보를 가진 아이템을 통해 q라는 속성을 가진 아이템의 추천 여부를 결정한다. 이 결정은 confusion matrix를 통해 Precision과 Recall, Accuracy로 표현된다. Recommender는 학습을 완료하고 추천 여부를 결정한 이후 결과를 출력한다.

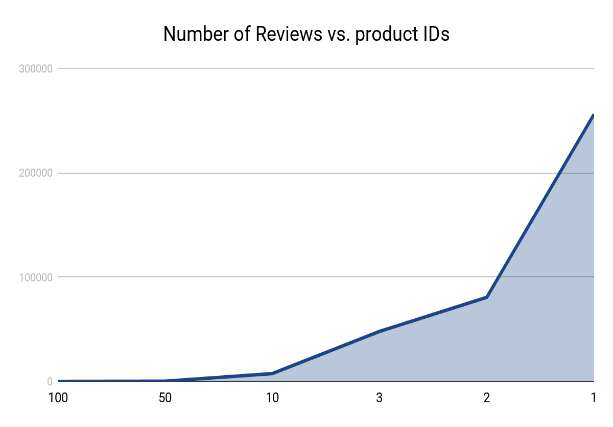
<Ⅲ> 본론

1. Data 분석 및 분석 결과 유추

1. Distribution of Scores

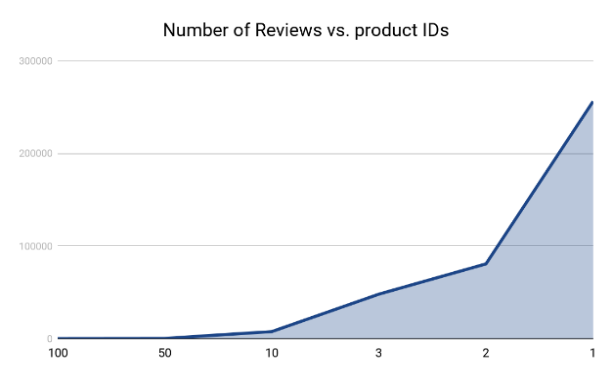
Product Ratings와 그에 따른 Score 빈도에 대해 분류로, Score 5점이 가장 많은 빈도를 가진다. 다음 <그림 3>에서는 세로축은 Product Ratings, 가로축이 Score를 표시하며 ★이 1, ☆을 표현한다. 대부분의 Product에 대한 Score가 좋기 때문에, 분별력이 떨어질 수 있다는 것을 예상할 수 있다.

<그림 3>

1. Number of Reviews by Product IDs

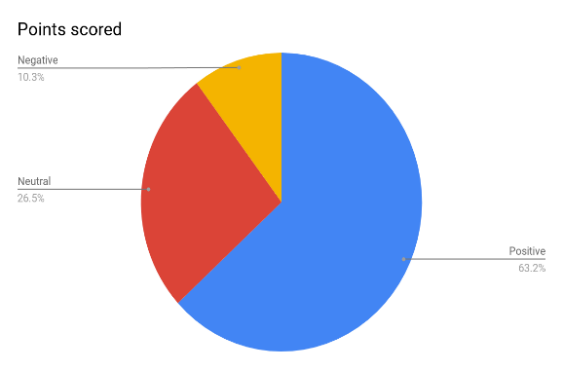
Product(food)당 가지고 있는 Review의 개수로, 대부분 1개의 리뷰를 가짐. <그림 4> 참조해보면, 가로축은 제품 ID의 개수이고 세로축은 리뷰의 개수이다. 대부분 리뷰 수가 하나이기 때문에, Recommand 수가 적게 나올 것을 예상할 수 있다.

<그림 4>

1. Number of Reviews by User IDs

ID당 쓴 Review의 개수로, 대부분 한 ID당 1개의 리뷰를 남김. <그림 5>참조해보면, 가로축은 User의 아이디 수고, 세로축은 리뷰의 개수이다. 의미 있는 Basket일 확률이 낮을 수 있다는 것을 예측할 수 있다. (User의 신뢰성 대해 파악하기 어려움)

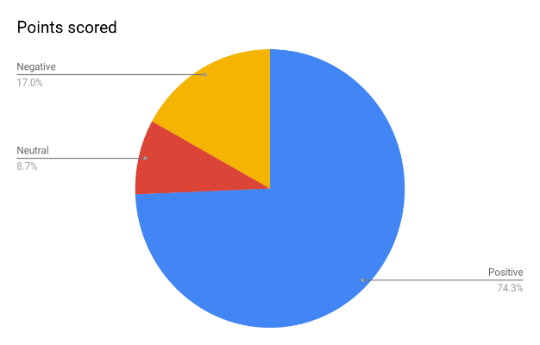
<그림 5>

1. Bias of Ratings of Product IDs

Product(food)가 받은 평가의 평균이다. <그림 6>참조해보면, 63.2%가 Positive로 음식 대부분이 좋은 평가를 받았음을 알 수 있다. 그러므로 Recommend를 정할 때, 더 Split한 기준이 필요함을 예상할 수 있다.

<그림 6>

1. Bias of Ratings by User ID

User가 작성한 평가의 평균이다. <그림 7>참조해보면, 74.3%가 Positive로 대부분의 User들이 긍정적으로 평가받고 있다는 것을 알 수 있다. 그러므로 Recommend를 정할 때, 더 Split한 기준이 필요함을 예상할 수 있다.

<그림 7>

2. Data Parsing

1) 데이터 변환

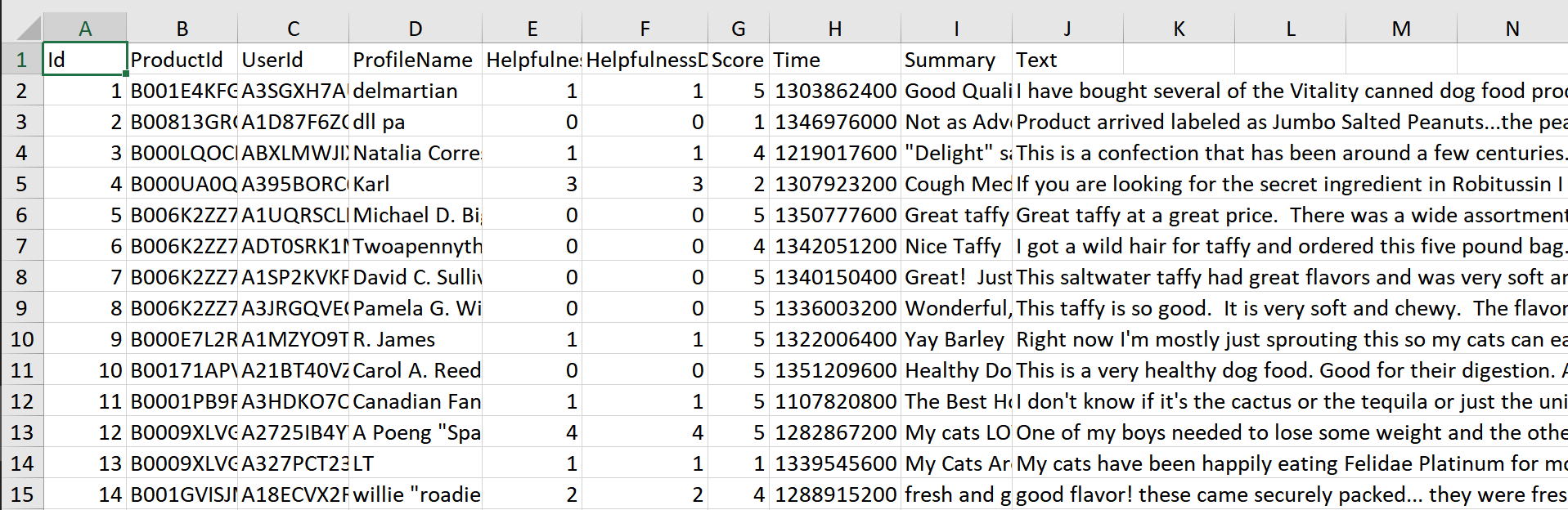
Kaggle에서 제공되는 Food Review data는 다음 <그림 8>에서 보이는 것처럼, 순서대로 ‘Id’, ‘ProductId’, ‘UserId’, ‘ProfileName’, ‘HelpfulnessNumerator’, ‘HelpfulnessDeniminator’, ‘Score’, ‘Time’, ‘Summary’, ‘Text’로 명명된 column을 가진다. 이 중 Market Basket 이론을 사용하기 위해 필요한 ProductId, UserId, Score를 재조합하여 Training과 Testing을 위한 data를 만든다. ProductId와 UserId는 동일한 값을 int value로 대응시켰으며 Score의 경우 Raw data에서의 값을 그대로 사용한다.

각 Basket을 0.90의 확률로 Training data로 할당하며 나머지는 Testing data로 할당한다. Testing data에서는 각 Basket에 속한 data를 0.97의 확률로 test 시에 Basket을 구성하는 data로 할당하고 나머지를 추천 여부를 판단하는 data로 사용한다.

학습에서의 편향을 줄이고 학습 시간을 단축시키고자 Review의 개수가 5보다 작거나 100보다 큰 basket은 Training data 및 Testing data에서 제외시켰다. 동일한 UserId가 같은 ProductId에 대해 여러 개의 Review를 작성하였을 경우 가장 첫번째 Review의 score를 학습에 반영하였다. 다음 <그림 9>와 <그림 10>은 각각 Training data와 Testing data에 해당한다. Testing data의 ‘c’와 ‘q’의 비율은 97:3이다.

2) 데이터 형태

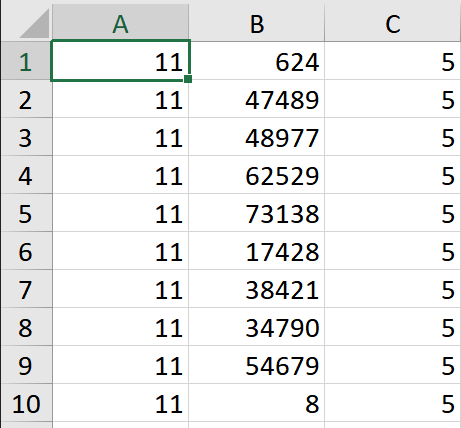
(1) 변환 전(Reviews.csv)



<그림 8>

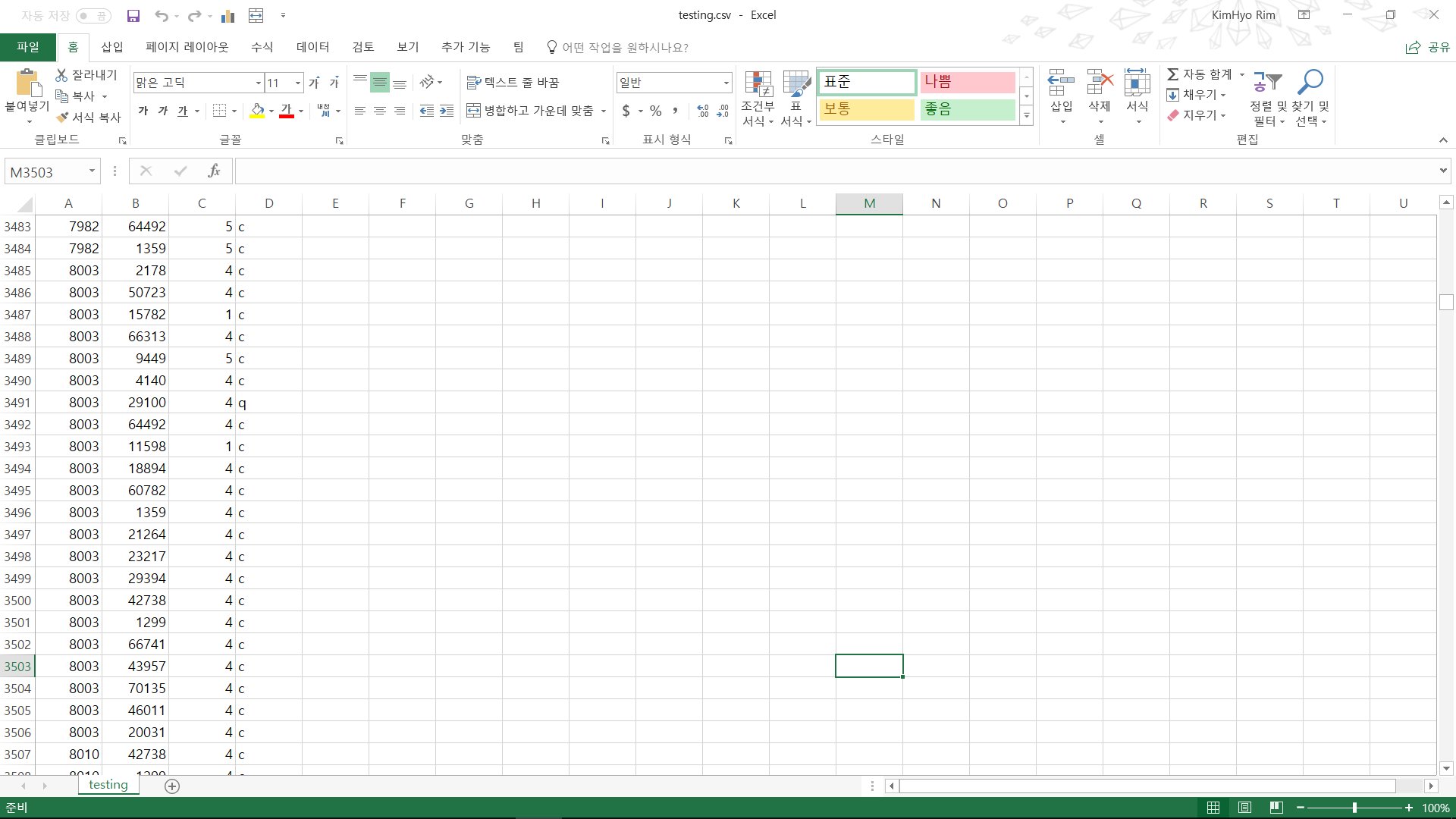
(2) 변환 후

-Training.csv

0

<그림 9>

-Testing.csv



<그림 10>

3. 결과

1) config.properties

Recommender의 data.outlier\_threshold, data.like\_threshold을 중점적으로 변형시키며 Food data에 가장 최적화 된 값을 찾았다. 각 변수 이외의 값은 특별한 언급이 없다면 다음<표 1>의 고정 값을 사용하였다.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Data.like\_threshold | data.outlier\_threshold | training.min\_supports | prediction.threshold\_2 | prediction.threshold\_3 | prediction.min\_evidence\_3 |
| 5.0 | 5 | 5 | 0.005 | 0.006 | 3 |

<표 1>

(1) data.outlier\_threshold

Food data의 특성상 대다수의 Product가 1개의 Review를 가진다. 그래서 처음에는 추천결과의 질이 낮을 것으로 예상하였으나 Score와 Review개수를 보았을 때 정말 맛있는 음식을 발견하였을 때만 Review를 쓴다고 가설을 수정하였다. 그리하여 Outlier의 수를 30에서부터 10씩 줄였을 때 큰 폭으로 Recall이 개선되었다. Outlier의 수가 5에 진입하면서부터 2까지 큰 변화가 없으므로 Outlier의 값은 5로 최적화하였다. 각 항목 변화 값은 <표 2>과 같다.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Outlier | 30 | 20 | 10 | **5** | 2 |
| Precision | 1.000 | 0.971 | 0.982 | **0.976** | 0.976 |
| Recall | 0.111 | 0.255 | 0.540 | **0.621** | 0.621 |
| Accuracy | 0.458 | 0.542 | 0.714 | **0.760** | 0.760 |
| Ratings [0:0, 0:1] | [254, 0] | [251, 3] | [250, 4] | **[248, 6]** | [248, 6] |
| Ratings [1:0, 1:1] | [352, 44] | [295, 101] | [182, 214] | **[150, 246]** | [150, 246] |

<표 2>

(2) data.like\_threshold

앞선 Outlier에서 볼 수 있듯이 Review의 개수는 작지만 좋은 질을 가질 것으로 예상하고 값을 3.0부터 5.0까지 수정하였다. 실제로도 threshold를 5로 두었을 때 전체 Accuracy에서 가장 높은 비율을 보였다. Recall의 경우 4.0으로 설정했을 때와 달리 값이 5.0일 때 Recall이 더 향상된 모습을 보였다. Precision의 경우 5.0보다 4.0이 소폭 상승하였으나 전체 Accuracy는 5.0에서 가장 높은 비율을 보이므로 like\_threshold값은 5.0으로 최적화하였다. 각 항목 변화 값은 <표 3>과 같다.

|  |  |  |  |
| --- | --- | --- | --- |
| like | 3.0 | 4.0 | **5.0** |
| Precision | 0.987 | 0.943 | **0.976** |
| Recall | 0.664 | 0.636 | **0.621** |
| Accuracy | 0.702 | 0.694 | **0.760** |
| Ratings [0:0, 0:1] | [82, 5] | [137, 19] | **[248, 6]** |
| Ratings [1:0, 1:1] | [189, 374] | [180, 314] | **[150, 246]** |

<표 3>

<Ⅳ>결론

본 프로젝트에서는 Market Basket Model 기반의 recommendation 시스템을 개발하기 위한 big data의 물성 분석과 최적화된 threshold를 찾기 위한 실험을 수행하였다. 실험 결과는 data의 score가 편향되어 있고, 의미 있는 basket의 수가 적기 때문에 recommendation의 횟수 및 recall 수준이 낮을 것이며 일단 recommend된 경우 높은 precision을 가질 것이라는 추측을 지지한다. outlier threshold를 변화시키며 수행한 첫 번째 실험에서는 outlier를 선정하는 기준을 완화함으로써 한 번 밖에 등장하지 않는 의미 없는 data를 효과적으로 제거할 수 있다는 것을 알 수 있었다. like threshold를 변화시키며 수행한 두 번째 실험에서는 score가 편향에 따라 의미 있는 선호와 불호의 기준이 다르다는 것을 알 수 있었다.

프로젝트를 통해 도출된 결론은 big data를 활용한 추천 시스템 개발 시 유효도가 낮은 data를 효과적으로 선별하여 시간과 메모리 사용량 등에서의 효율성을 높이는 데 사용될 수 있을 것이다. 또한 data의 특성 별 의미 있는 threshold 값을 결정하는 데에 도움을 주어 추천 시스템의 정확도를 높일 수 있을 것이다.

1. Recommender는 입력 받은 데이터를 Market-basket 모델을 활용하여 분석한다. 또한, config.property파일을 입력받아 outlier와 threshold 처리에 필요한 값들을 얻는다. Recommender는 다음 4가지 단계를 거치며 결과를 도출해 낸다. [↑](#footnote-ref-1)
2. 각 Basket속의 Item의 연관 관계는 한 Item i가 선택된 이후 거의 반드시 선택되는 Item j의 관계인 confidence of rule, 혹은 모든 아이템 집합에 나타나는 아이템 j의 관계인 interest of rule으로 표현 할 수 있다. [↑](#footnote-ref-2)
3. ‘I’라는 item을 선택한다면 ‘j’라는 item을 선택할 확률이 높다’는 결론을 가진 알고리즘이다. 만약 Itemset I가 support(threshold값) s보다 많이 존재한다면 모든 subset도 마찬가지이며 그 반대도 성립한다. 알고리즘 상에서는 1부터 k(N: 1, 2, … , n)번째 단계까지 각 basket의 itemset을 고유한 정수로 변환하고 각 itemset에 대해 빈번하게 발생하는 count가 더 큰 itemset만 k+1 단계로 전달한다. [↑](#footnote-ref-3)