Computer Convergence Application
Homework 3: Frequent pattern mining

컴퓨터공학부
2017-18538
황선영

1. finding candidate motif sequence

Drug-induced liver injury(DILI)는 가장 예측하기 어려운 항생제에 대한 부작용 중 하나이며 승인된 의약품이 시판되고도 철회하게 하는 주요 원인이다. FDA는 약 1000여개에 대한 약물에 대해 DILI severity와 potential 에 따라 분류하여 dataset을 구축하였다. Dataset에 있는 약물에 대해 높은 확률로 DILI를 유발하는 약물과 DILI를 유발하지 않는 약물 각각에 대해 pattern minining algorithm을 이용하여 candidate motif sequence를 찾음으로써 어떤 motif sequence가 DILI 유발 여부에 관여하는지 분석한다.

vDILIConcern이 vMost_DILI-Concern인 compound, 즉 거의 drug-induced liver injury를 causing하는 약물을 positive로 정한 다음 frequent하게 나타나는 motif sequence를 찾는다. 이 때 negative에 있는 motif sequence는 제외하기 때문에 positive에만 있는 motif sequence만 pure_positive_result로 나타난다. vNo-DILI-Concern인 compound, 즉 drug-induced liver injury를 causing하지 않는 약물을 negative로 정한 다음 frequent하게 나타나는 motif sequence를 찾는다. 이 때 positive에 있는 motif sequence는 제외하기 때문에 negative에만 있는 motif sequence만 pure_negative_result로 나타난다. share_positive_result와 share_negative_result는 positive와 negative에 동시에 나타나는 candidate motif sequence를 의미한다. 두 결과가 같으므로 share_positive_result만 첨부하였다. 그리고 data의 양이 많아 index 0-4에 해당하는 요소 5개만 나타냈다.

2. pattern mining algorithm: Apriori vs FP growth

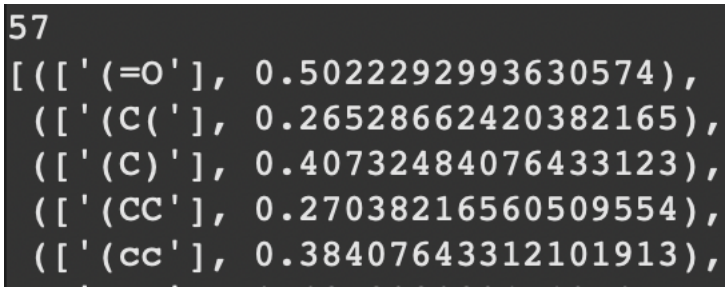
Apriori algorithm은 apriori property와 join과 prune property을 이용하는 pattern mining algorithm이다.

```
126
[(['2cc'], 0.259554140127388 53),
 (['O)c'], 0.299044585987261 14),
 (['cc2'], 0.329617834394904 44),
 (['c(c', '(=O'], 0.27643312101910 83),
 (['c1c', '(=O'], 0.32006369426751 59),
```

[Figure1-1] Apriori algorithm: pure_positive_result

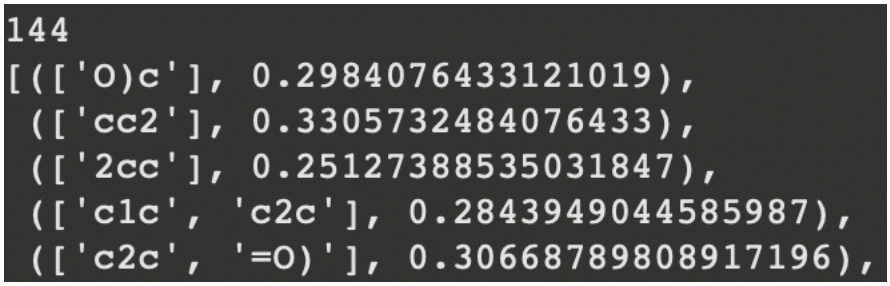
```
29
[(['(O)'], 0.35940860215053766),
 ([')O)'], 0.2615591397849462),
 (['1)C'], 0.258870967741935 46),
 (['CC('], 0.3293010752688172),
 (['CCC'], 0.3682795698924731),
```

[Figure1-2] Apriori algorithm: pure_negative_result

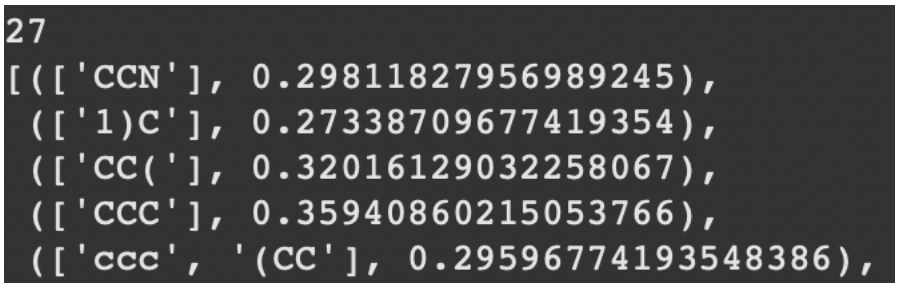


[Figure1-3] Apriori algorithm: share_positive_result

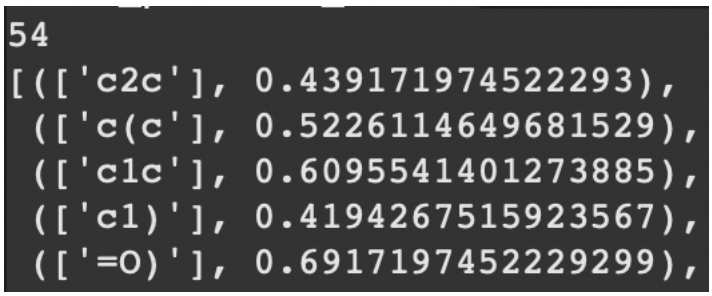
FP-growth algorithm은 minimum support를 만족하는 database로부터 conditional frequent pattern tree와 conditional pattern을 construct하는 pattern mining algorithm이다. Scan 횟수가 apriori algorithm에 비해 적어 execution time이 짧다. Execution time이 중요하다면 FP-growth algorithm을 선택하면 좋을 것이다.



[Figure2-1] FP-growth algorithm: pure_positive_result



[Figure2-2] FP-growth algorithm: pure_negative_result

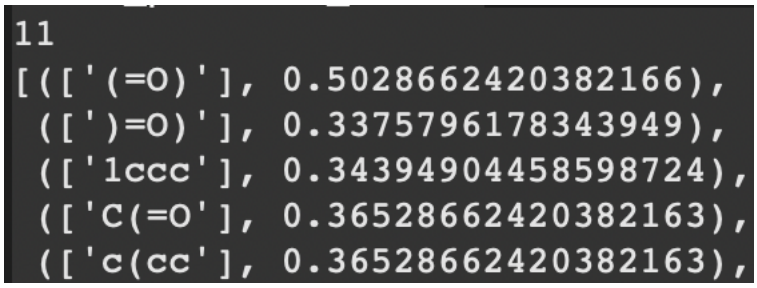


[Figure2-3] FP-growth algorithm: share_positive_result

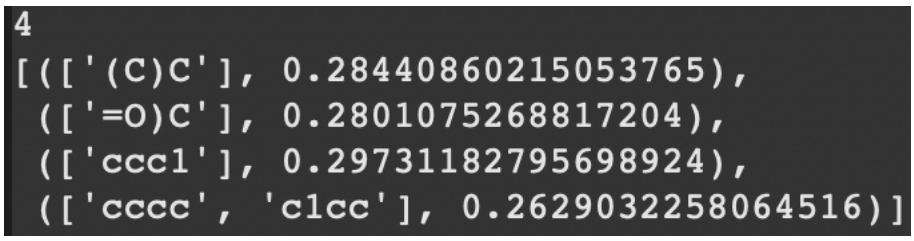
3. length of patterns(n-gram)

n-gram은 연속적인 n개의 token(word, character)로 구성된 것이다. 문장 분석의 경우 n-gram이 커질수록 맥락을 잘 내포하는 경향이 있다. N-gram이 3일 때의 결과는 Figure1과 같다. N-gram이 2인 경우 pure_positive의 길이가 5000이 넘을 만큼 많아서 결과가 유효하지 않다고 판단해 첨부하지 않았다. N-gram이 4일 때와 5일 때를 확인해보니 candidate motif sequence의 구조가 더 명확해질 수 있겠으나 그 수가 굉장히 적은 것을 알 수 있었다. N-gram이 3일 때 적절한 수의

candidate motif sequence가 도출되므로 n-gram=3이 가장 적절하다고 판단된다.



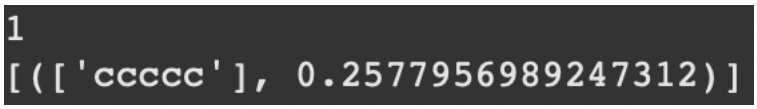
[Figure3-1] n-gram=4: pure_positive_result



[Figure3-2] n-gram=4: pure_negative_result

[Figure3-3] n-gram=4: share_positive_result

[Figure4-1] n-gram=5: pure_positive_result

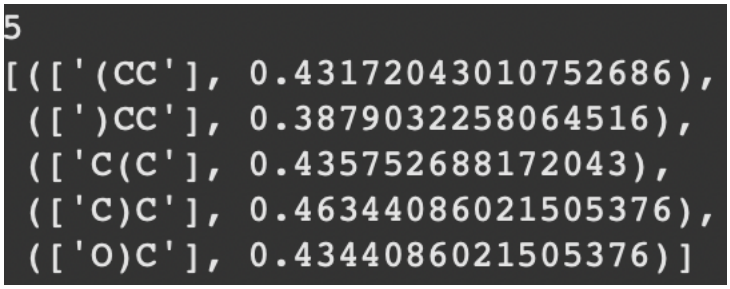


[Figure4-2] n-gram=5: pure_negative_result

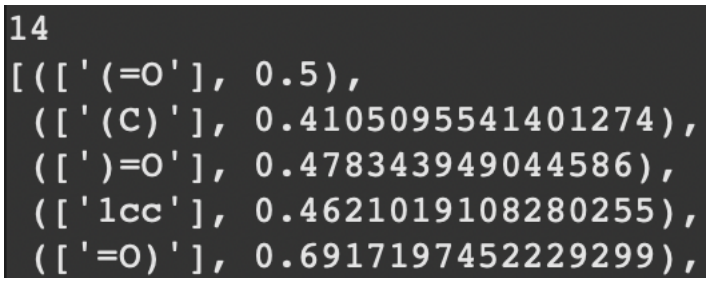
[Figure4-3] n-gram=5: share_positive_result

4. minimum support

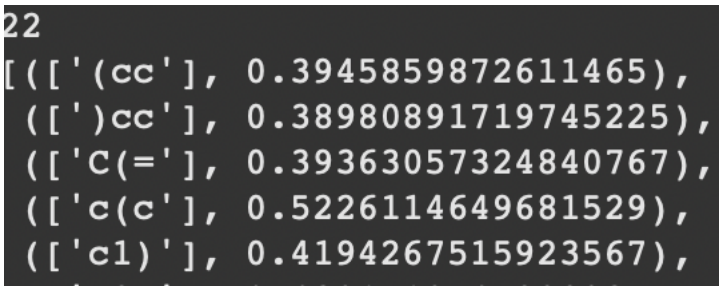
Minimum support가 커질 수록 support가 높아지는 대신 candidate motif sequence의 수가 급격하게 적어지는 것을 알 수 있다. 하지만 minimum support가 너무 낮다면 support가 낮은 candidate motif sequence가 pure_positive_result, pure_negative_result에 나타나게 되는데 support가 낮으므로 그 집단의 대표성을 띄기 힘들다. Minimum support가 0.25일 때의 결과는 Figure1과 같다. Minimum support가 0.25-0.375사이의 범위에 있어야 support도 적당히 높고 대신 candidate motif sequence의 수도 적당한 결과가 도출될 것이다.



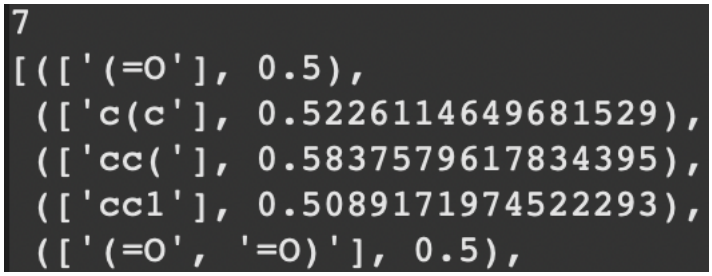
[Figure5-1] minimum_sup=0.375: pure_positive_result



[Figure5-2] minimum_sup=0.375: pure_negative_result

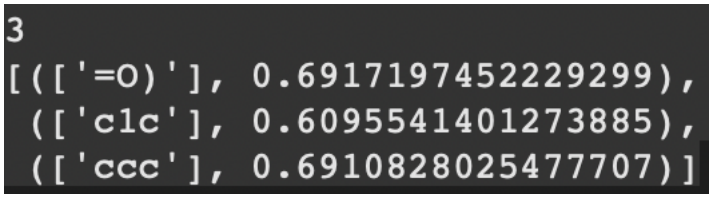


[Figure5-3] minimum_sup=0.375: share_positive_result



[Figure5-1] minimum_sup=0.5: pure_positive_result

[Figure5-2] minimum_sup=0.5: pure_negative_result



[Figure5-3] minimum_sup=0.5: share_positive_result

5. augmentation & tokenization

Augmentation은 data에 인위적인 변화를 가해 새로운 training data를 대량 확보하는 기법이다. 현실

세계에서도 실제로 존재할 법한 데이터를 생성함으로써 좀 더 일반화된 model을 얻는 것을 목표로 한다. Augmentation이 20일 때의 결과는 Figure1과 같다. Augmentation은 기존의 data의 정보량은 보존한 상태로 noise를 주는 방식이기 때문에 정보량 자체는 변하지 않는다. 정보량에 약간의 변화를 주는 것이기 때문에 deep learning으로 분석된 data의 표현되는 강력한 특징을 느슨하게 만드는 것이다. 이는 overfitting을 막아주고 예측 범위를 약간 넓혀줄 수는 있으나 매우 큰 효과를 가져오진 못한다.

```
65
[(['(=O'], 0.404838709677419 36),
 (['(C('], 0.33172043010752 69),
 (['(C)'], 0.47096774193548385),
 (['(CC'], 0.42258064516129035),
 (['(cc'], 0.28440860215053765),
```

[Figure6-1] Augmentation=10: pure_positive_result

```
21
[(['(O)'], 0.3639784946236559),
 ([')O)'], 0.2564516129032258),
 (['CC('], 0.321505376344086 05),
 (['CCC'], 0.3548387096774194),
 (['CCN'], 0.30053763440860215),
```

[Figure6-2] Augmentation=10: pure_negative_result

```
157
[(['O)c'], 0.290445859872611 46),
 (['cc2'], 0.321656050955414),
 (['(=O', '1cc'], 0.2681528662420382),
 (['c(c', '(=O'], 0.297452229299363 05),
 (['c1c', '(=O'], 0.345222929936305 7),
```

[Figure6-3] Augmentation=10: share_positive_result