Association Analysis

연관성 분석

7/16. 이승한

목차

1. Association Analysis 란?

- 흥미로운,관계‘발견하기!

  (also called ,장바구니분석‘)

- MKT에서 활용(ex) 맥주& 기저귀)

2. 용어 소개

(1) Transaction : { 빵, 초콜렛, 수세미}

(2) Association Rules : { 빵, 초콜렛, 수세미} -> { 바나나}

(3) Support (지지도): 특정사건이 얼마나자주 발생하는가? support(A) = A가 발생한비율

support(A,B) = A,B가 둘 다 발생할 확률

(4) Confidence (신뢰도): 특정조건하에, 다른사건이 얼마나자주 발생? confidence(A->B) : A를 산 사람이B도 샀을 확률

(5) Lift(향상도): 특정조건하에, 다른사건의 발생이 어떻게 변화? lift(A->B) : confidence(A->B) / support(B)

2. 용어 소개

- 지지도(support) = $P(X \cap Y)$
- 신뢰도(confidence) = $P(Y|X) = \dfrac{P(X \cap Y)}{P(X)} = \dfrac{support}{P(X)}$
- 향상도(lift) = $\dfrac{P(Y|X)}{P(Y)} = \dfrac{P(X \cap Y)}{P(X)P(Y)} = \dfrac{confidence}{P(Y)}$

lift 쉽게 생각해!

상품Y를 구매한비율에 비해, ,X를 구매한 고객이'Y를 구매한 비율이 몇 배?

2. 용어 소개

Transaction #	Purchase
1	딸기,당근,수박
2	딸기,메론,레몬,호박
3	딸기,당근,호박
4	메론,레몬,수박
5	딸기,당근,수박

Example

Support (지지도)

{딸기,당근}: 3/5(=0.6)

{호박}: 2/5(=0.4)

{딸기}: 4/5(=0.8)

{당근}: 3/5(=0.6)

Confidence (신뢰도)

{딸기}->{당근}: 0.6 / 0.8

{당근}->{딸기}= 0.6 / 0.6

Lift (향상도)

{딸기}->{당근}: (0.6/0.8) / 0.6

{당근}->{딸기}: (0.6/0.6) / 0.8

3. Apriori Algorithm

이 알고리즘이 필요한 이유는? (뭐가 문제길래?)

'Item 수 증가' -> 따져봐야 할 경우의 수 너무 많아!

ex) k개의 item : 2^k의 경우의 수

(이중 의미 없는 것들도 많이 섞여 있을 것!)

ex. {휘발유, 립스틱, 메로나맛 우유}

좀 덜 중요한 건 빼도 되지 않을까??

해결책: Ignore the Rare Combination!

3. Apriori Algorithm

{A상품,B상품} 조합이 많이 등장한다면,

당연히 A상품 / B상품 각각 많이 팔리는 것!

다르게 말하면, A상품 자체가 많이 등장하지 않는다면,

굳이 {A,B}든 {A,F,G} 조합이든 따져볼 필요가 있을까? NO!

진행 단계

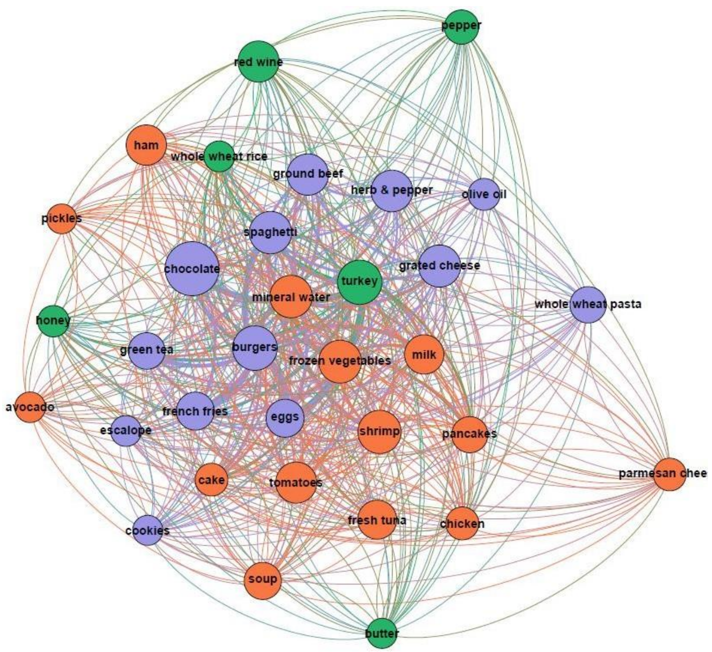
1) 모든 품목을 대상으로, minimum threshold 넘는 것만 선별!
2) 선택된 품목들만 대상으로 rule 만들기!

4. 코드 실습

파일 열어주세요

필요 package : 1) mlxtend / 2) apyori

5. 시각화(Gephi)



네트워크 분석 Visualization

중 요 개 념

1) **Centrality**

2) **Modularity**

[1.Centrality]

1. Degree Centrality(정도중심성)

- 가장 간단! 다른노드와많이연결이되어있는가?
- 많이 연결되어 있으면, D.C 값이큼

➡ EigenvectorCentrality (고유벡터중심성)

Degree centrality + '노드의 중요도' 고려

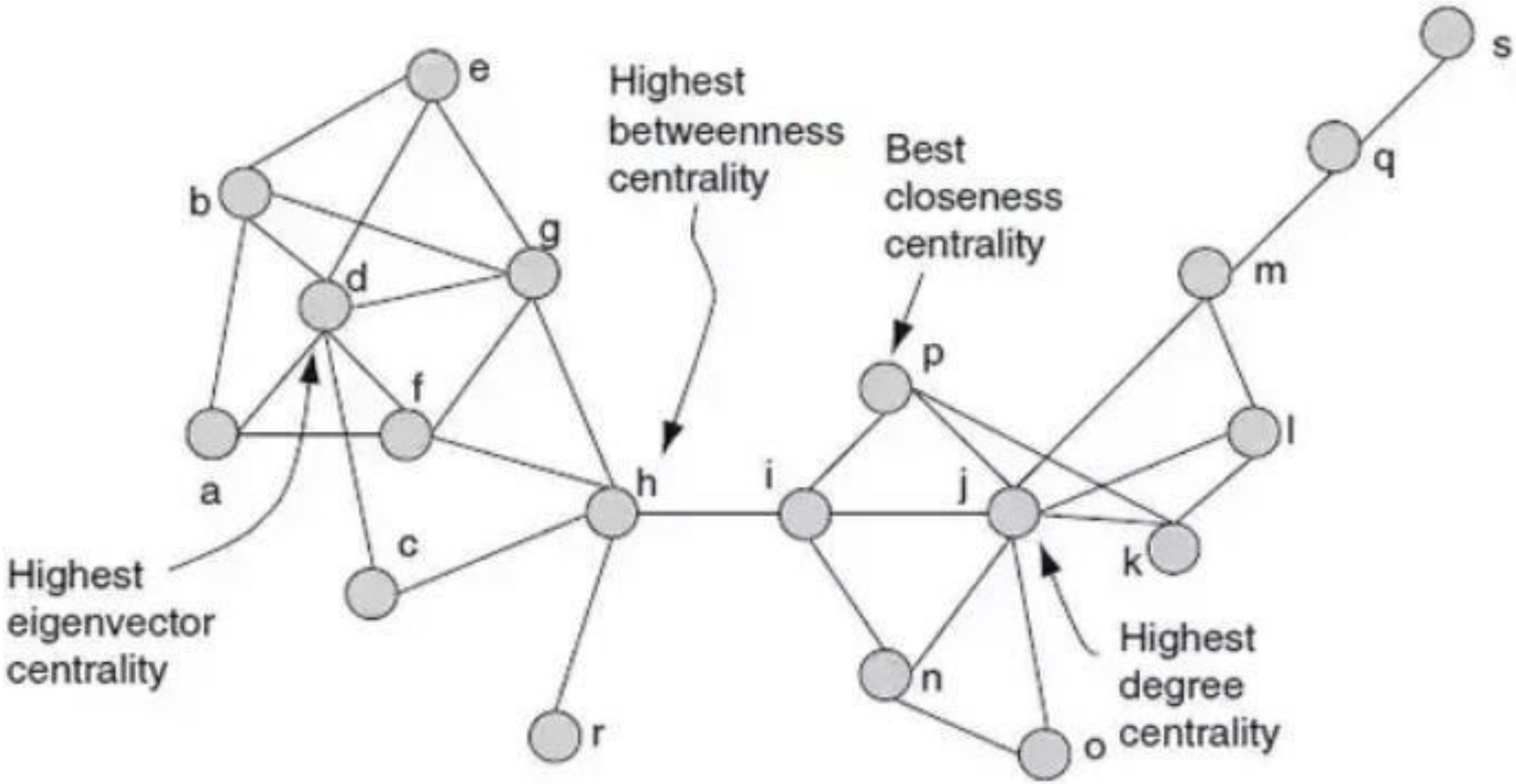
2. Betweenness Centrality(사이중심성)

- 노드들 각의 최단 경로를가지고 계산
- 한 노드에서 다른 노드로 가는 '최단경로'에 A노드가 얼마나 많이 포함되어 있는지
- 최단 경로에 많이 포함되어 있으면, B.C값이 큼

3. Closeness Centrality(근접중심점)

- 가정 : 중요한 노드일수록 다른노드까지도달하는경로가짧을것!
- A노드가, 다른 모든 노드(B,C...Z)로까지 가는 거리의 평균 -> 이 값을 역수 취함
- 다른 노드들까지 도달하는 경로가 짧으면, C.C값이 큼

[1.Centrality]



[2.Modularity]

measure the strength of division of a network into modules

https://en.wikipedia.org/wiki/Modularity_(networks)

Modularity [edit]

Hence, the difference between the actual number of edges between node v and w and the expected number of edges between them is

$$A_{vw} - \frac{k_v k_w}{2m}$$

Summing over all node pairs gives the equation for modularity, Q.[1]

$$Q = \frac{1}{2m} \sum_{vw} \left[A_{vw} - \frac{k_v k_w}{2m} \right] \frac{s_v s_w + 1}{2} \qquad (3)$$

It is important to note that **Eq. 3** holds good for partitioning into two communities only. Hierarchical partitioning (i.e. partitioning into two communities, then the two sub-communities further partitioned into two smaller sub communities only to maximize Q) is a possible approach to identify multiple communities in a network. Additionally, (3) can be generalized for partitioning a network into c communities.[5]

$$Q = \frac{1}{(2m)} \sum_{vw} \left[A_{vw} - \frac{k_v k_w}{(2m)} \right] \delta(c_v, c_w) = \sum_{i=1}^{c} (e_{ii} - a_i^2) \qquad (4)$$

where e_{ij} is the fraction of edges with one end vertices in community i and the other in community j:

$$e_{ij} = \sum_{vw} \frac{A_{vw}}{2m} 1_{v \in c_i} 1_{w \in c_j}$$

and a_i is the fraction of ends of edges that are attached to vertices in community i:

$$a_i = \frac{k_i}{2m} = \sum_j e_{ij}$$

[2.Modularity]

,그룹 내에서는Dense Connection, (close between groups)

,그룹 갂에는Sparse Connection` (far between groups)

-1 ~ 1 사이값

Calculation : (A) –(B)

(A) Connection of edges within modules

(B) Random distribution of links between all nodes (regardless of modules)

[2.Modularity]