AI기반 데이터 분석 및 AI Agent 개발 과정

『1과목 : 』
AI기반 데이터 분석

2025.09.22-10.02(9일, 62시간)

Prepared by DaeKyeong K

Ph.D.

삼성전자 구미사업부 G.제조팀

HS Global
HARDWARE SOFTWARE

목차

『1-11』
기타 데이터 마이닝

Association Rule Analysis

Syallabus

학습목표

- 이 워크샵에서는 Association Rule Analysis 확인할 수 있다

눈높이 체크

- Association Rule Analysis 을 알고 계시나요?

1. 연관성 분석(장바구니 분석)

연관성 분석(장바구니 분석) 란?

- 연관성 분석 기법은 방대한 데이터 세트에서 객체나 아이템 간의 연관관계를 찾아내는 분석기법이다. Y값(종속변수, 목표변수)이 없는 상태에서 데이터 속에 숨겨져 있는 패턴, 규칙을 찾아내는 비지도학습(unsupervised learning)의 하나인 '연관규칙분석(Association Rule Analysis)', 혹은 유통업계에서 사용하는 용어로 '장바구니분석(Market Basket Analysis) ' 이라고도 한다.

- 상품 추천에 사용하는 분석기법

1. 연관규칙분석, 장바구니분석 (Association Rule Analysis, Market Basket Analysis) : 고객의 대규모 거래데이터로부터 함께 구매가 발생하는 규칙(예: A à 동시에 B)을 도출하여, 고객이 특정 상품 구매 시 이와 연관성 높은 상품을 추천
2. 순차분석 (Sequence Analysis) : 고객의 시간의 흐름에 따른 구매 패턴(A à 일정 시간 후 B)을 도출하여, 고객이 특정 상품 구매 시 일정 시간 후 적시에 상품 추천
3. Collaborative Filtering : 모든 고객의 상품 구매 이력을 수치화하고, 추천 대상이 되는 고객A와 다른 고객B에 대해 상관계수를 비교해서, 서로 높은 상관이 인정되는 경우 고객B가 구입 완료한 상품 중에 고객A가 미구입한 상품을 고객A에게 추천

1. 연관성 분석(장바구니 분석)

연관성 분석(장바구니 분석) 란?

4. Contents-based recommendation : 고객이 과거에 구매했던 상품들의 속성과 유사한 다른 상품 아이템 중 미구매 상품을 추천 (↔ Collaborative Filtering은 유사 고객을 찾는 것과 비교됨)
5. Who-Which modeling : 특정 상품(군)을 추천하는 모형을 개발 (예: 신형 G5 핸드폰 추천 스코어모형)하여 구매 가능성 높은(예: 스코어 High) 고객(군) 대상 상품 추천

- 넥플릭스에서 상품추천에 이용하는 알고리즘
- 넥플릭스는 이용자들이 동영상에 매긴 별점과 위치정보, 기기정보, 플레이버튼 클릭 수, 평일과 주말에 따른 선호 프로그램, 소셜 미디어 내에서 언급된 횟수 등을 분석해 알고리즘을 개발했다.
- 출처 : 넷플릭스의 빅데이터, 인문학적 상상력과의 접점, 조영신, KISDI 동향 Focus
- 규칙(rule)이란 "if condition then result" (if A --> B) 의 형식으로 표현을 합니다.
- 연관규칙(association rule)은 특정 사건이 발생하였을 때 함께 (빈번하게) 발생하는 또 다른 사건의 규칙을 말합니다.

1. 연관성 분석(장바구니 분석)

연관성 분석(장바구니 분석) 란?

- 대량의 트랜잭션 데이터에서의 규칙성이나 패턴화 도출의 목적이 주어진 경우, 이의 문제 해결을 위해 다양한 연관 규칙 알고리즘을 비교하고 적용할 수 있다.
- 연관성 분석 기법은 방대한 데이터 세트에서 객체나 아이템 간의 연관관계를 찾아내는 분석기법이다. 연관관계는 빈발 아이템이나 연관규칙의 형태로 표현되는데 빈발 아이템은 {X,Y}처럼 표현하고, 연관 규칙은 {X}→{Y}처럼 특정 아이템 'X'가 발생하면, 'Y'가 함께 발생한다는 형태로 표현한다. 예를 들어 {순대, 족발}→{보쌈}처럼 순대와 족발을 구매하면 보쌈도 함께 구매한다는 규칙을 찾아내는 것이다. 여기서 {순대, 족발}은 해당 규칙에서의 '조건'(antecedent)에 해당하고, {보쌈}은 '결과'(consequence)에 해당하게 된다. 이러한 연관성 분석은 사전에 목표변수(Y)가 주어지지 않으며, 각 트랜잭션(혹은 거래데이터)에서 객체나 아이템 간의 패턴을 찾아내는 기법이므로 자율학습(비지도 학습) 기법에 속하게 된다.

1. 연관성 분석(장바구니 분석)

연관성 분석 주요 측도

- 연관성 분석에서 가장 핵심적인 개념은 각 아이템 간의 연관성을 파악하는 주요 3개 측도인 지지도(Support), 신뢰도(Confidence), 향상도(Lift)이다.

• 지지도(Support)

– 지지도는 전체 데이터 세트에서 해당 아이템 집합이 포함된 비율을 말하며 아래의 S(X) 혹은 S(X,Y)와 같이 표현된다.

$$S(X) = \frac{count(X)}{N} \qquad (4.11)$$

$$S(X,Y) = \frac{count(X,Y)}{N} = P(X \cap Y)$$

– 즉, 지지도는 빈도적 관점에서 확률을 정의할 때, 전체 데이터 세트 중 아이템 집합 {X,Y}가 발생할 확률과 같다.

1. 연관성 분석(장바구니 분석)

연관성 분석 주요 측도

- 신뢰도(Confidence)

- 신뢰도는 연관규칙 {X}→{Y}에서 '조건' X를 포함한 아이템 세트 중에서 X, Y 둘 다 포함된 아이템 세트가 발생한 비율을 말하는데, 규칙의 왼쪽에 있는 '조건 X'가 발생했다는 조건하에 규칙의 오른쪽에 있는 '결과 Y'가 발생할 확률을 의미한다. 신뢰도는 특정 연관 규칙의 예측력이나 정확도에 대한 측정이다. 사실 이 신뢰도는 조건부 확률 P(Y|X) 와 동일한 의미이다.

$$Conf(X \Rightarrow Y) = \frac{S(X,Y)}{S(X)} = \frac{\frac{count(X,Y)}{N}}{\frac{count(X)}{N}} = \frac{count(X,Y)}{count(X)} \qquad (4.12)$$

$$= \frac{P(X \cap Y)}{P(X)} = P(Y|X)$$

- 신뢰도 (X→Y)와 신뢰도 (Y→X)는 서로 같지 않다. 즉, 두 신뢰도 모두 X, Y 가 모두 포함된 지지도(X,Y)가 분자에 포함되어 있지만 신뢰도 (X→Y)는 지지도(X)가 분모에 들어가게 되고, 신뢰도 (Y→X)는 지지도(Y)가 분모에 들어가게 되는 점이 다르다. 결국 이는 조건부 확률 P(Y|X)와 P(X|Y)가 서로 다른 결과를 가져오는 것과 동일한 의미라고 할 수 있다.

1. 연관성 분석(장바구니 분석)

연관성 분석 주요 측도

- 향상도(Lift)
  - 지지도와 신뢰도는 연관규칙 생성과 탐사에 있어서 매우 중요한 핵심개념임에는 틀림없지만, 지지도(X,Y)와 신뢰도(X→Y)가 높았다는 것만으로 유의미한 규칙이라고 결론 내리기는 어렵다. 그 이유는 만일 전체 데이터 세트에서 원래부터 아이템 세트 {Y}가 포함된 경우의 수가 많았다면, 아이템 세트 {Y}와 함께 아이템 세트 {X}가 포함되어 있을 가능성도 그만큼 커지고, 지지도(X,Y)와 신뢰도(X→Y)는 높게 나타날 수밖에 없기 때문이다.
  - 즉, 연관규칙 {X}→{Y}가 탐색 되었을 때 정말로 조건 {X}가 발생했을 때 결과 {Y}가 함께 나타나는 경우의 수가 많아서 그런 것인지, 아니면 원래부터 {Y}가 많이 포함되어 있어 연관규칙 {X}→{Y}가 탐색 된 것인지에 대한 보다 명확한 측정 지표가 필요하게 된다. 이럴 때 이를 측정하는 지표가 바로 향상도(Lift)이다.
  - 향상도(Lift)가 의미하는 바는 조건 {X}가 주어지지 않았을 때의 결과 {Y}가 발생할 확률 대비, 조건 {X}가 주어졌을 때의 결과 {Y}의 발생 확률의 증가 비율을 의미한다. 즉, 아이템 집합 {Y}가 원래 발생된 경우의 수보다 연관규칙 {X}→{Y}가 탐색 되었을 때 조건에 해당하는 아이템 집합 {X}가 주어졌다는 "정보"가 결과 아이템 집합 {Y}가 발생하게 되는 경우의 수를 예상하는 데 얼마나 유용하느냐를 나타낸다고 할 수 있다.

1. 연관성 분석(장바구니 분석)

연관성 분석 주요 측도

- 향상도(Lift)
- 향상도(Lift)는 다음과 같이 측정된다.

$$Lift(X \Rightarrow Y) = \frac{Conf(X \Rightarrow Y)}{S(Y)} \tag{4.13}$$

$$= \frac{\frac{S(X,Y)}{S(X)}}{S(Y)} = \frac{S(X,Y)}{S(X)S(Y)} = \frac{\frac{count(X,Y)}{N}}{\frac{count(X)}{N}\frac{count(Y)}{N}}$$

$$= \frac{P(Y|X)}{P(Y)} = \frac{\frac{P(X \cap Y)}{P(X)}}{P(Y)} = \frac{P(X \cap Y)}{P(X)P(Y)}$$

- 따라서 향상도가 1을 넘어가면 조건과 결과 아이템 집합 간에 서로 양의 상관관계가 있으며, 1보다 작으면 서로 음의 상관관계 만일 향상도가 1로 나타나면 조건과 결과 아이템 집합은 서로 독립적인 관계라고 할 수 있다.

1. 연관성 분석(장바구니 분석)

연관성 분석 주요 측도

- 예

지지도(Support), 신뢰도(Confidence), 향상도(Lift) 예시

Customer ID	Transaction ID	Items
1131	1번	계란, 우유
2094	2번	계란, 기저귀, 맥주, 사과
4122	3번	우유, 기저귀, 맥주, 콜라
4811	4번	계란, 우유, 맥주, 기저귀
8091	5번	계란, 우유, 맥주, 콜라

N = 5 (전체 transaction 개 수)

s(Y) = n(Y)/ N
= n{2번, 3번, 4번}/N = 3/5 = 0.6

연관규칙 {계란, 맥주} → {기저귀} 에 대해
X Y

- **지지도(Support)**

s(X→Y) = n(X∪Y)/N
= n{2번, 4번}/ N
= 2/5 = 0.4

- **신뢰도(Confidence)**

c(X→Y) = n(X∪Y)/ n(X)
= n{2번, 4번}/ n{2번, 4번, 5번}
= 2/3 = 0.667

- **향상도(Lift)**

Lift(X→Y) = c(X→Y)/ s(Y)
= 0.667 / 0.6 = 1.111

1. 연관성 분석(장바구니 분석)

연관성 분석 주요 측도

- **연관성 분석 기법의 주요 활용 분야**

• 연관성 분석은 소매업 혹은 쇼핑몰 등의 거래데이터에서 물품 간에 연관관계를 파악하는 '장바구니 분석'에 가장 많이 사용되어 왔기 때문에 마케팅 영역에서는 연관성 분석 = 장바구니 분석으로 거의 동일한 용어로 사용되기도 한다. 그러나 연관성 분석이 비단 장바구니 분석에만 활용되는 것은 아니며, 객체(혹은 문서) 및 아이템 간의 연관 관계성 혹은 빈출 패턴을 파악할 필요가 있는 다양한 영역에서 활용되고 있다.

1. 쇼핑/유통 분야에서 구매된 물품 간의 장바구니 분석
2. 금융상품이나 서비스 간의 가입 패턴의 연관성 분석
3. 인터넷/모바일 서비스 상품 추천 시스템의 구매 항목 간 연관성 알고리즘으로 사용
4. 웹 페이지 간의 링크 관계성 분석 (웹 사용자의 행동 추적 및 패턴 예측)
5. 가맹점 간의 방문 연관성 및 순서 패턴 도출 통한 쿠폰 마케팅 등 활용
6. 생물 정보학 분야에서의 단백질과 유전자 패턴 분석
7. 신용카드 사기 및 보험청구 사기 등 부정거래 패턴 발견

• 즉, 방대한 데이터베이스나 인터넷 로그 데이터 등에서 아이템, 객체들 간의 발생 연관성을 파악하거나 발생 순서 간 패턴을 파악하려는 거의 모든 분야에서 활용 가능한 현실 세계의 실무 활용성이 높은 기법이라고 할 수 있다.

1. 연관성 분석(장바구니 분석)

연관성 분석 주요 측도

- 연관성 분석 알고리즘
- 연관성 분석의 대표적인 알고리즘에는 아프리오리(Apriori) 알고리즘과 이를 개선한 빈출 패턴 성장(FP-Growth) 알고리즘이 있다.

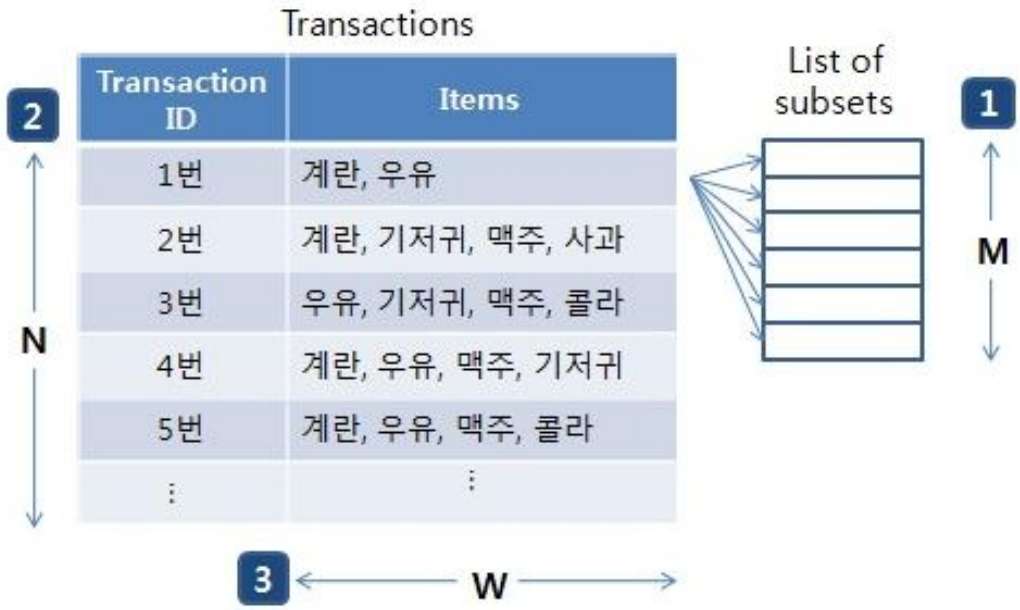
알고리즘	알고리즘 설명
아프리오리(Apriori) 알고리즘	빈출 아이템 집합을 효과적으로 계산하기 위하여 검토해야 할 대상 집합 Pool을 효과적으로 줄여주는 기법. 즉, 특정 집합 {1,2}가 빈발하지 않는다면, 이들을 포 함한 {0,1,2}, {1,2,3}, {1,2,4}, {0,1,2,3,4} 등도 빈발하지 않은 것이 되어, 지지도 계 산 검토 대상에서 제외할 수 있다. 이런 식으로 검토대상의 아이템 데이터 집합 을 효과적으로 줄여서 연관규칙을 계산해내는 기법. 데이터가 커질 경우 계산량이나 속도 면에서 비효율적이라는 단점이 있음
빈출패턴 성장 (FP-Growth)	아프리오리 알고리즘의 빈발 아이템을 찾는 방법을 개선한 알고리즘으로서 데이 터베이스를 모든 중요한 정보를 가진 FP-Tree라는 구조로 압축한 뒤, 높은 빈도 의 세트에 관련한 조건부 데이터 세트로 분할하여 규칙을 만들어 낸다. 전체 데 이터베이스를 검색하는 작업이 아프리오리 알고리즘보다 현저하게 줄어들어서 높은 성능과 처리속도를 보이는 알고리즘임. 반면, 아프리오리 알고리즘보다 구 현이 어렵다는 단점이 있음.
DHP algorithm	Transaction 개수(N)을 줄이는 전략

2. 연관성 분석 알고리즘

연관성 분석 주요 측도

- 연관성 분석 알고리즘

Association Rule : strategy and algorithm



1 모든 가능한 항목집합 개수(M)를 줄이는 방식 ➡ Apriori algorithm

2 Transaction 개수(N)를 줄이는 방식 ➡ DHP Algorithm

3 비교하는 수(W)를 줄이는 방식 ➡ FP-growth Algorithm

2. 연관성 분석 알고리즘

Apriori algorithm

- 최소지지도 이상을 갖는 항목집합을 빈발항목집합(frequent item set)이라고 합니다. 모든 항목집합에 대한 지지도를 계산하는 대신에 최소 지지도 이상의 빈발항목집합만을 찾아내서 연관규칙을 계산하는 것이 Apriori algorithm의 주요 내용입니다.
- 빈발항목집합 추출의 Apriori Principle

1. 한 항목집합이 빈발(frequent)하다면 이 항목집합의 모든 부분집합은 역시 빈발항목집합이다.(frequent item sets -> next step)
2. 한 항목집합이 비비발(infrequent)하다면 이 항목집합을 포함하는 모든 집합은 비빈발항목집합이다. (superset -> pruning)

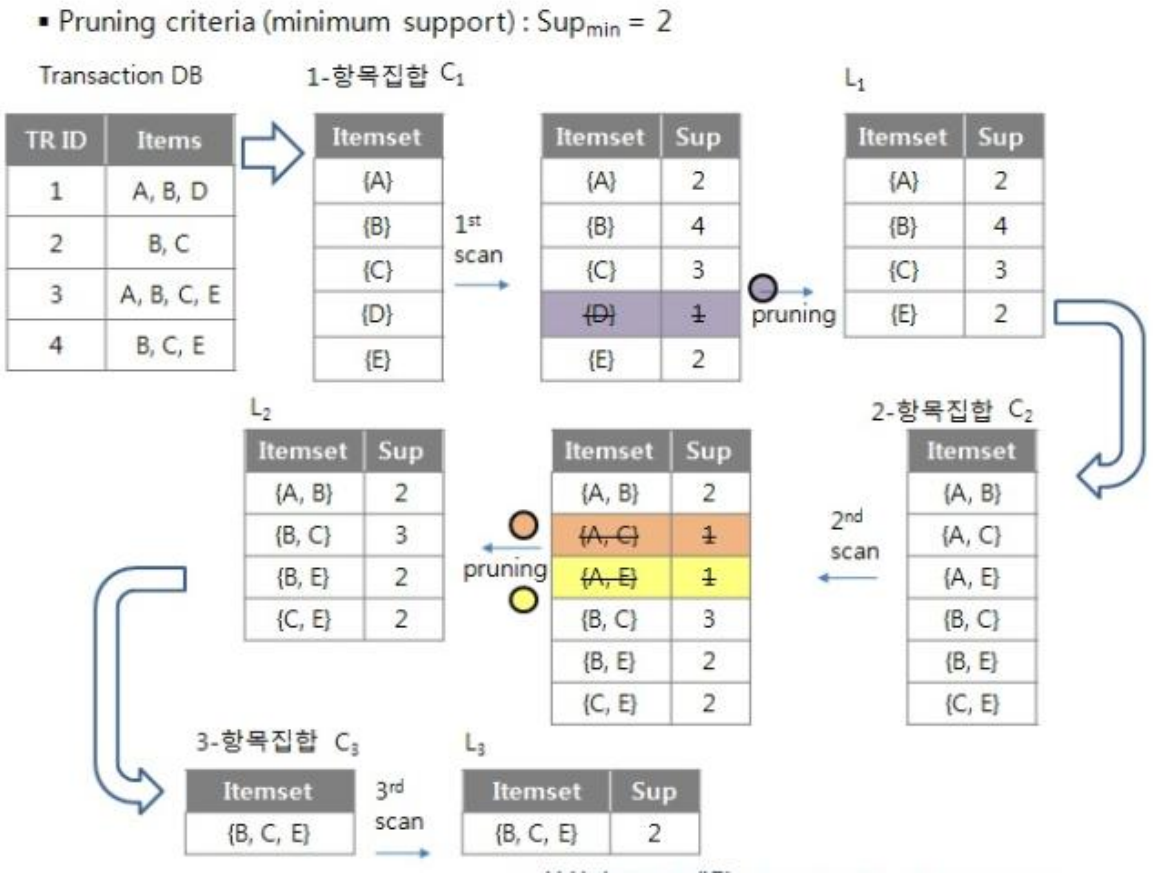
- Apriori Pruning Principle

If there is any itemset which is infrequent, its superset should not be generated/tested!
(Agrawal & Srikant @VLDB'94, Mannila, et al. @ KDD' 94)

2. 연관성 분석 알고리즘

Apriori algorithm

- {A, B, C, D, E}의 5개 원소 항목을 가지는 4건의 transaction에서 minimum support 2건 (=2건/총4건=0.5) 기준으로 pruning 하는 예



2. 연관성 분석 알고리즘

Apriori algorithm

- 빈발항목집합 추출 Pseudo Aprior algorithm

1) $L_1 = \{\text{large 1-itemsets}\};$
2) for $(k = 2; L_{k-1} \neq 0; k++)$ do begin
3) $C_k = apriori-gen(L_{k-1})$; // New candidates
4) forall transactions $t \in T$ do begin
5) $C_t = subset(C_k, t)$; // Candidates contained in t
6) forall candidates $c \in C_t$ do
7) c.count ++;
8) end
9) $L_k = \{c \in C_k \mid c.count \geq \text{min sup}\}$
10) end
11) Answer $= \bigcup_k L_k$;

k-itemset	[Notation] An itemset having k items
L_k	Set of large k-itemsets (those with minimum support) Each member of this set has two fields: i) itemset and ii) support count.
C_k	Set of candidate k-itemsets (potentially large itemsets) Each member of this set has two fields: i) itemset and ii) support count
$\overline{C}_k$	Set of candidate k-itemsets when the TIDs of the generating transactions are kept associated with the candidates

[Reference] "Fast Algorithms for Mining Association Rules", Rakesh Agrawal, Ramakrishman Srikant, 1994

2. 연관성 분석 알고리즘

Apriori 예

1. mlxtend 패키지 설치

```
pip install mlxtend
```

2. 패키지 로드

```
# 패키지 로드
import pandas as pd
from mlxtend.preprocessing import TransactionEncoder
from mlxtend.frequent_patterns import apriori
```

2. 연관성 분석 알고리즘

Apriori 예

3. 데이터셋 생성

```
# 데이터셋 생성
dataset=[['사과','치즈','생수'],
['생수','호두','치즈','고등어'],
['수박','사과','생수'],
['생수','호두','치즈','옥수수']]
dataset

>>
```

```
[['사과', '치즈', '생수'],
 ['생수', '호두', '치즈', '고등어'],
 ['수박', '사과', '생수'],
 ['생수', '호두', '치즈', '옥수수']]
```

2. 연관성 분석 알고리즘

Apriori 예

4. 가나다 순으로 Column값을 생성해서(고등어,사과,생수,수박,옥수수,치즈,호두) 첫번째 (사과,치즈.생수) 데이터에 고등어가 표시되어이있으면 True값으로 없으면 False값으로 표시한다. 다음 사과도 첫번째 데이터에 사과가 있으면 1값으로 없으면 0값으로 표시한다. 이러한 것을 반복하면 4X7 행렬이 생성된다

```
te = TransactionEncoder()
te_ary = te.fit(dataset).transform(dataset)
df = pd.DataFrame(te_ary, columns=te.columns_) #위에서 나온걸 보기 좋게 데이터프레임으로 변경
df
>>
```

	고등어	사과	생수	수박	옥수수	치즈	호두
0	False	True	True	False	False	True	False
1	True	False	True	False	False	True	True
2	False	True	True	True	False	False	False
3	False	False	True	False	True	True	True

2. 연관성 분석 알고리즘

Apriori 예

5. 지지도를 0.5로 놓고 apriori를 돌려보면 아래와 같이 결과가 출력된다. 사과를 살 확률 0.5, 치즈 등을 함께 살 확률 0.75

```
frequent_itemsets = apriori(df, min_support=0.5, use_colnames=True)
frequent_itemsets
```

```
>>
```

	support	itemsets
0	0.50	(사과)
1	1.00	(생수)
2	0.75	(치즈)
3	0.50	(호두)
4	0.50	(생수, 사과)
5	0.75	(치즈, 생수)
6	0.50	(호두, 생수)
7	0.50	(치즈, 호두)
8	0.50	(치즈, 호두, 생수)

2. 연관성 분석 알고리즘

Apriori 예

6. 신뢰도가 0.3인 항목만 보기

```
from mlxtend.frequent_patterns import association_rules
association_rules(frequent_itemsets, metric="confidence", min_threshold=0.3
```

>>

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction	zhangs_metric
0	(생수)	(사과)	1.00	0.50	0.500000	1.000000	1.0	0.000	inf	0.0
1	(사과)	(생수)	0.50	1.00	0.500000	1.000000	1.0	0.000	inf	0.0
2	(치즈)	(생수)	0.75	1.00	0.750000	1.000000	1.0	0.000	inf	0.0
3	(생수)	(치즈)	1.00	0.75	0.750000	1.000000	1.0	0.000	inf	0.0
4	(호두)	(생수)	0.50	1.00	0.500000	1.000000	1.0	0.000	inf	0.0
5	(생수)	(호두)	1.00	0.50	0.500000	1.000000	1.0	0.000	inf	0.0
6	(치즈)	(호두)	0.75	0.50	0.666667	1.333333	1.5	0.125	1.0	0.5
7	(호두)	(치즈)	0.50	0.75	0.500000	1.000000	1.3	0.125	inf	0.5
8	(치즈, 호두)	(생수)	0.50	1.00	0.500000	1.000000	1.0	0.000	inf	0.0
9	(치즈, 생수)	(호두)	0.75	0.50	0.666667	1.333333	1.5	0.125	1.0	0.5

2. 연관성 분석 알고리즘

Apriori 예

결론
신뢰도는 위에 표처럼 해석 할 수 있는데 예를 들면 10번째 열을 보면 (생수,호두)를 구매한 고객이 (치즈)를 구매할 확률이 높다고 판단 할 수 있다

향상도(lift)	의미
1 보다 작은 경우	우연적 기회보다 높은 확률
1 인 경우	독립
1 보다 큰 경우	우연적 기회보다 낮은 확률

2. 연관성 분석 알고리즘

FP-Growth* Algorithm (Frequent-Pattern Growth)

- FP-growth 알고리즘을 적용시키기 위해서 가장 먼저 해야 할 일은 FP-tree를 만드는 것이다. 이 FP-tree를 만듦으로써 우리는 기존에 데이터셋을 모두 찾아보면서 찾아야했던 frequent itemset들을 FP-tree 하나에서만 찾을 수 있게 된다.
- 우선, 우리에게 다음과 같은 데이터셋이 있다고 하자.

$$D = \left[\{초장\}^{9}, \{과메기\}^{40}, \{과메기, 김\}^{50}, \{과메기, 김, 초장\}^{1} \right]$$

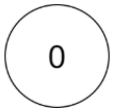
- 예시 데이터셋. 초장만 산 사람이 9명, 과메기만 산 사람이 40명, 과메기와 김을 함께 산 사람이 50명, 과메기, 김, 초장을 모두 함께 산 사람이 1명이라는 뜻이다.

1. 모든 각각의 item들에 대해 전체 데이터셋에서 나온 빈도를 계산한다.
   - ✓ 데이터셋의 모든 item들에 대해 빈도를 계산한다. 우리의 예시에서는 초장이 9+1=10으로 10번, 과메기가 40+50+1=91번, 김이 50+1=51번 나타났다. 이를 빈도가 높은 순서대로 표현하면 과메기:91, 김:51, 초장:10이다.

2. 연관성 분석 알고리즘

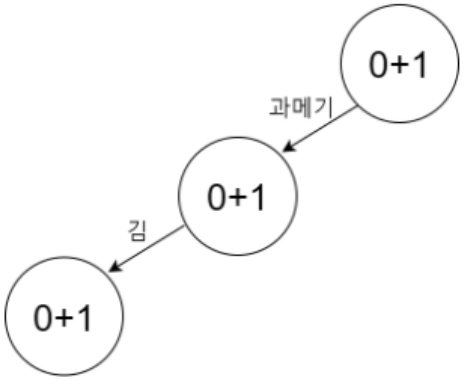
FP-Growth* Algorithm (Frequent-Pattern Growth)

2. itemset 하나씩 tree에 더함으로써 tree를 만들어준다.
   - ✓ 이제 tree를 만들 차례이다. 우선, 맨 처음에 0 node를 하나 만든다.



시작 node

   - ✓ 그리고 각 instance를 하나씩 더해주면서 tree를 만들어간다. 이 때, 빈도가 높은 item이 더 0에 가까운 node가 된다. 예를 들어, 우선 {과메기, 김} itemset으로 tree를 만든다고 하자. 그러면 다음과 같은 형태가 된다.

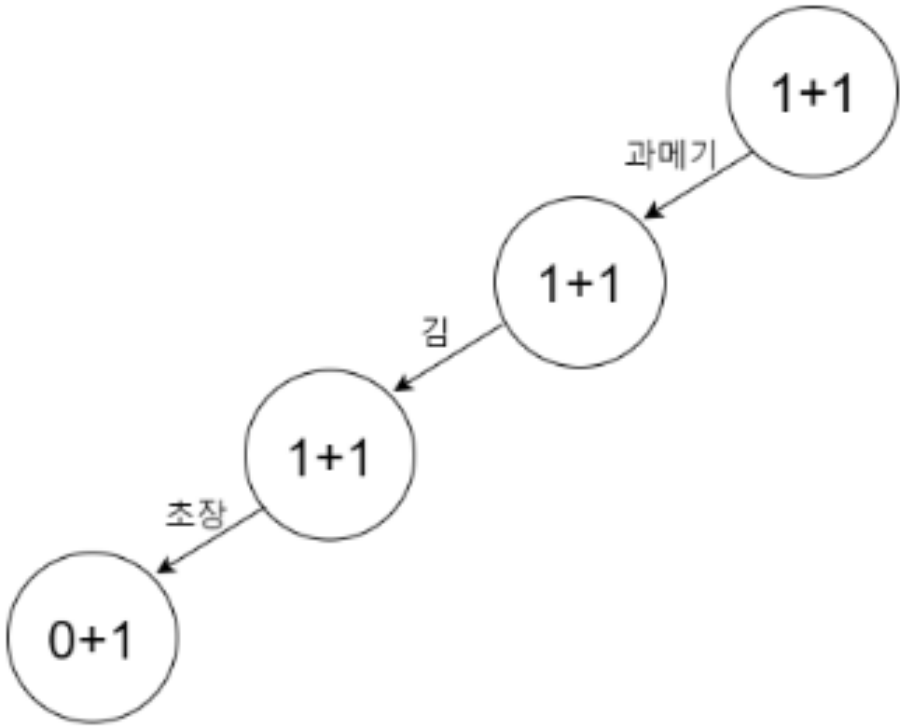


{과메기, 김}이 추가된 tree

2. 연관성 분석 알고리즘

FP-Growth* Algorithm (Frequent-Pattern Growth)

- ✓ 빈도가 더 높은 과메기가 더 가까운 곳에 위치하는 tree가 만들어졌다. 이 단계에서 우리가 {과메기, 김, 초장} itemset을 추가한다고 하자. 그러면 다음과 같은 형태가 될 것이다.

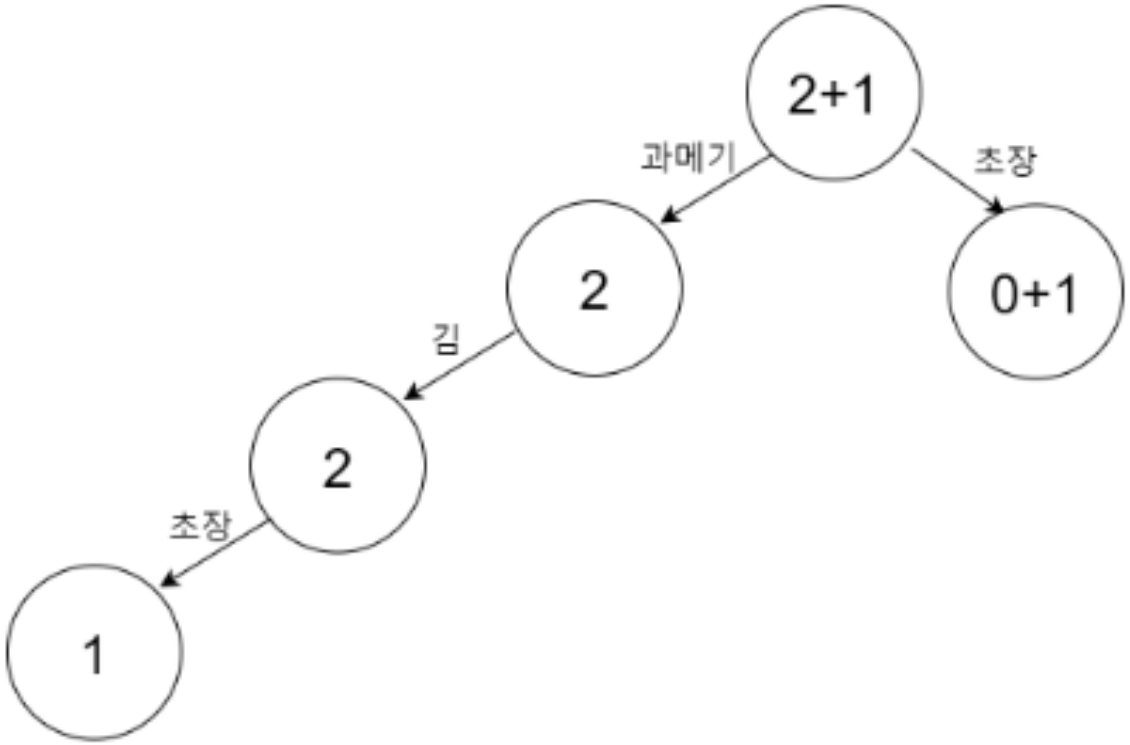


{과메기, 김, 초장}이 추가된 tree

2. 연관성 분석 알고리즘

FP-Growth* Algorithm (Frequent-Pattern Growth)

✓ 여기서 {초장} itemset을 추가한다고 하자. 그러면 다음과 같은 형태가 될 것이다.

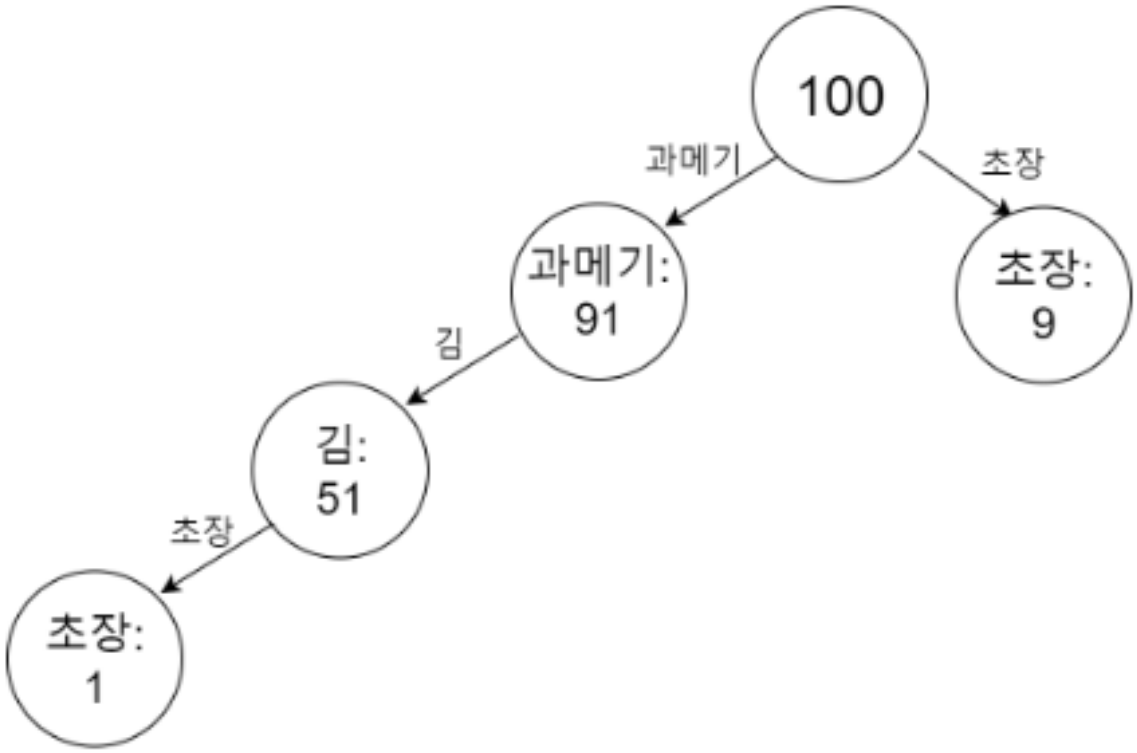


{초장}이 추가된 tree

2. 연관성 분석 알고리즘

FP-Growth* Algorithm (Frequent-Pattern Growth)

✓ 이런 식으로 dataset 안의 모든 itemset에 대해 tree를 만들 수 있다. 그러면 최종적으로 다음과 같은 FP-tree를 얻을 수 있다.



최종 FP-tree

2. 연관성 분석 알고리즘

FP-Growth* Algorithm (Frequent-Pattern Growth)

- ✓ 이렇게 dataset 안의 모든 itemset의 정보를 반영하는 FP-tree를 만들어낼 수 있다. 이 FP-tree를 이용하면 아주 쉽게 support 값을 구할 수 있다. 예를 들어, 우리의 tree에서 {김, 과메기}의 support를 구하고 싶다고 하자. 김은 무조건 과메기와 함께 나타나기 때문에 김의 frequency만 고려하면 되고, 이는 51이므로 {김, 과메기}의 support는 51이 된다. 이러한 방법으로 우리의 dataset의 support를 모두 구하면 다음과 같다.

item sets	support
{과메기}	91
{김}, {김, 과메기}	50
{초장}	10
{과메기, 김, 초장}	1

2. 연관성 분석 알고리즘

FP-Growth* Algorithm (Frequent-Pattern Growth)

pip install mlxtend –upgrade

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Requirement already satisfied: mlxtend in /usr/local/lib/python3.7/dist-packages (0.14.0)
Collecting mlxtend
  Downloading mlxtend-0.20.0-py2.py3-none-any.whl (1.3 MB)
     |████████████████████████████████| 1.3 MB 4.8 MB/s
Requirement already satisfied: setuptools in /usr/local/lib/python3.7/dist-packages (from mlxtend) (57.4.0)
Requirement already satisfied: scikit-learn>=1.0.2 in /usr/local/lib/python3.7/dist-packages (from mlxtend) (1.0.2)
Requirement already satisfied: matplotlib>=3.0.0 in /usr/local/lib/python3.7/dist-packages (from mlxtend) (3.2.2)
Requirement already satisfied: numpy>=1.16.2 in /usr/local/lib/python3.7/dist-packages (from mlxtend) (1.21.6)
Requirement already satisfied: scipy>=1.2.1 in /usr/local/lib/python3.7/dist-packages (from mlxtend) (1.4.1)
Requirement already satisfied: pandas>=0.24.2 in /usr/local/lib/python3.7/dist-packages (from mlxtend) (1.3.5)
Requirement already satisfied: joblib>=0.13.2 in /usr/local/lib/python3.7/dist-packages (from mlxtend) (1.1.0)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.7/dist-packages (from matplotlib>=3.0.0->mlxtend) (0.11.0)
Requirement already satisfied: python-dateutil>=2.1 in /usr/local/lib/python3.7/dist-packages (from matplotlib>=3.0.0->mlxtend) (2.8.2)
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.1 in /usr/local/lib/python3.7/dist-packages (from matplotlib>=3.0.0-
>mlxtend) (3.0.9)
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.7/dist-packages (from matplotlib>=3.0.0->mlxtend) (1.4.2)
Requirement already satisfied: typing-extensions in /usr/local/lib/python3.7/dist-packages (from kiwisolver>=1.0.1->matplotlib>=3.0.0->mlxtend)
(4.2.0)
Requirement already satisfied: pytz>=2017.3 in /usr/local/lib/python3.7/dist-packages (from pandas>=0.24.2->mlxtend) (2022.1)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.7/dist-packages (from python-dateutil>=2.1->matplotlib>=3.0.0->mlxtend)
(1.15.0)
Requirement already satisfied: threadpoolctl>=2.0.0 in /usr/local/lib/python3.7/dist-packages (from scikit-learn>=1.0.2->mlxtend) (3.1.0)
Installing collected packages: mlxtend
  Attempting uninstall: mlxtend
    Found existing installation: mlxtend 0.14.0
    Uninstalling mlxtend-0.14.0:
      Successfully uninstalled mlxtend-0.14.0
Successfully installed mlxtend-0.20.0
WARNING: The following packages were previously imported in this runtime:
  [mlxtend]
You must restart the runtime in order to use newly installed versions.
```

2. 연관성 분석 알고리즘

FP-Growth* Algorithm (Frequent-Pattern Growth)

```
pip install mlxtend –upgrade

from sklearn.preprocessing import MultiLabelBinarizer

data = [
    ['휴지', '물티슈', '샴푸'],
    ['수세미', '물티슈', '비누'],
    ['휴지', '수세미', '물티슈', '비누'],
    ['수세미', '비누']
]

mlb = MultiLabelBinarizer()
encoded_data = mlb.fit_transform(data)
df_encoded = pd.DataFrame(encoded_data, columns=mlb.classes_)

print(df_encoded)

>>
   물티슈  비누  샴푸  수세미  휴지
0    1   0   1    0   1
1    1   1   0    1   0
2    1   1   0    1   1
3    0   1   0    1   0
```

2. 연관성 분석 알고리즘

FP-Growth* Algorithm (Frequent-Pattern Growth)

```
from mlxtend.frequent_patterns import fpgrowth
fpgrowth(df_encoded, min_support=0.5, use_colnames=True)
>>
```

	support	itemsets
0	0.75	(물티슈)
1	0.50	(휴지)
2	0.75	(수세미)
3	0.75	(비누)
4	0.50	(비누, 물티슈)
5	0.50	(물티슈, 수세미)
6	0.50	(비누, 물티슈, 수세미)
7	0.50	(휴지, 물티슈)
8	0.75	(비누, 수세미)

THANK YOU.

앞으로의 엔지니어는 단순한 '코더'나 '기계 조작자'가 아니라 뇌-기계 인터페이스를 통해 지식과 능력을 즉각 확장하는 존재(뉴로-인터페이스: Neuro Interface)가 될 수 있습니다.

🎯 목표 달성을 위한 여정이 시작됩니다.
✨ 궁금한 점이 있으시면 언제든 문의해주세요!
🚀 함께 코더와 프롬프트 전문가로 성장해 나갑시다!