연관규칙

- 연관규칙분석(association rule analysis)
  - 대용량 데이터베이스에서 변수들 간의 흥미로운 관계를 찾도록 고안된 방법
  - 마케팅과 웹마이닝 등 여러 분야에서 사용
- 자료에서 존재하는 항목(item) 간의 if-then 형식의 연관규칙을 찾는 방법(비지도 학습)
  - 상품의 구매, 서비스 등 일련의 거래 또는 사건들 간의 연관성에 대한 규칙을 발견하기 위해 적용
  - 마케팅에서 손님의 장바구니에 들어있는 품목간의 관계를 알아본다는 의미에서 장바구니 분석(market basket analysis)

- 연관규칙분석은 특정한 상품을 구입한 고객이 어떤 부류에 속하는지, 그들이 왜 그런 구매를 했는지 알아보기 위해서 고객들이 구매한 상품에 대한 자료를 분석하는 것
  - 이 분석결과를 통해 효율적인 매장진열, 패키지 상품의 개발, 교차판매전략 구사, 기획상품의 결정 등에 응용
- 적용분야
  - 호텔 및 백화점에서 고객이 원하는 서비스 파악
  - 금융서비스 내역으로 특정 서비스를 받을 가능성 파악
  - 인터넷 쇼핑몰에서의 추천상품 파악

연관규칙

- 연관규칙분석은 자료로부터 항목간의 연관규칙을 찾아주는데 찾은 모든 규칙이 유용한 정보를 가지고 있는 것은 아님
  - 연관규칙은 유용한 규칙, 자명한 규칙, 설명이 불가능한 규칙으로 구분
- 분석 결과 수많은 규칙들을 얻게 되며 이 규칙들 중에서 유용한 규칙을 발견하는 것은 분석자의 몫

- **편의점의 거래내역**

고객번호	품목
1	오렌지쥬스, 사이다
2	우유, 오렌지쥬스, 식기세처제
3	오렌지쥬스, 세제
4	오렌지쥬스, 세제, 사이다
5	식기 세척제, 사이다

- **동시 구매표 작성**

	오렌지 쥬스	식기 세척제	우유	사이다	세제
오렌지쥬스	4	1	1	2	2
식기세척제	1	2	1	1	0
우유	1	1	1	0	0
사이다	2	1	0	3	1
세제	2	0	0	1	2

- **동시구매표로부터 "오렌지쥬스 구입 고객은 사이다를 구입한다" 와 같은 간단한 규칙을 생성**
  - **"If X, then Y"와 같은 형식으로 표현되는 규칙을 연관규칙**
- **모든 규칙이 유용한 것은 아님**
  - **유용한 규칙이 되기 위한 필요조건은 두 품목 X와 Y를 동시에 구매한 경우의 수가 일정 수준 이상**
  - **품목 X를 포함하는 거래 중 품목 Y를 구입하는 경우의 수도 일정수준 이상**

연관규칙분석의 측도

- **연관규칙이 유용한 규칙일 필요조건에 대한 측도**
  - **지지도(support), 신뢰도(confidence), 향상도(lift)**
- **지지도**
  - **두 품목 X와 Y의 지지도는 전체 거래들 중 품목 X와 품목 Y를 동시에 포함하는 거래의 비율**

$$\text{지지도} = P(X \cap Y) = \frac{\text{X와 Y가 동시에 포함된 거래수}}{\text{전체 거래수}}$$

- **신뢰도 P(Y|X)**

$$\text{신뢰도} = \frac{P(X \cap Y)}{P(X)} = \frac{\text{X와 Y가 동시에 포함된 거래수}}{\text{품목 X를 포함하는 거래수}}$$

	오렌지 쥬스	식기 세척제	우유	사이다	세제
오렌지쥬스	4	1	1	2	2
식기세척제	1	2	1	1	0
우유	1	1	1	0	0
사이다	2	1	0	3	1
세제	2	0	0	1	2

- **위 동시구매표에서 "오렌지쥬스 구입 고객은 사이다를 구입한다" 지지도와 신뢰도**

$$\text{지지도} = P(X \cap Y) = \frac{2}{5} \quad \text{신뢰도} = \frac{P(X \cap Y)}{P(X)} = \frac{2}{4}$$

예제 3.1

- **다음은 피자의 거래내역에 대해서 연관규칙들의 지지도와 향상도를 구하시오.**
  - **페퍼로니, 올리브, 버섯은 추가하는 토핑**

항목	거래수
페퍼로니	100
올리브	150
버섯	200
{페퍼로니, 올리브}	400
{페퍼로니, 버섯}	300
{올리브, 버섯}	200
{페퍼로니, 올리브, 버섯}	100
추가안함	550
전체 거래수	2000

- **각 품목에 대한 거래수와 전체 거래수에 대한 상대도수**

항목	품목이 포함된 거래수	확률
페퍼로니	900	0.450
올리브	850	0.425
버섯	800	0.400
{페퍼로니, 올리브}	500	0.250
{페퍼로니, 버섯}	400	0.200
{올리브, 버섯}	300	0.150
{페퍼로니, 올리브, 버섯}	100	0.050

$$\text{상대도수} = \frac{\text{품목이 포함된 거래수}}{\text{전체 거래수}}$$

- **지지도와 신뢰도**

항목 X⇒Y	지지도 P(X∩Y)	상대도수 P(X)	신뢰도 P(Y|X)
페퍼로니⇒올리브	0.250	0.450	0.556
올리브⇒페퍼로니	0.250	0.425	0.588
버섯⇒올리브	0.150	0.400	0.375
올리브⇒버섯	0.150	0.425	0.353
페퍼로니⇒버섯	0.200	0.450	0.444
버섯⇒페퍼로니	0.200	0.400	0.500
{페퍼로니, 올리브}⇒버섯	0.050	0.250	0.200
{페퍼로니, 버섯}⇒올리브	0.050	0.150	0.333
{올리브, 버섯}⇒페퍼로니	0.050	0.200	0.250

$$\text{지지도} = P(X \cap Y) \qquad \text{신뢰도} = \frac{P(X \cap Y)}{P(X)}$$

- **세 가지 토핑을 모두 포함하는 규칙 중에서 신뢰도가 가장 높은 규칙**
  - **{페퍼로니, 버섯} ⇒올리브 , 신뢰도=0.333**
  - **그러나 전체 거래에서 토핑 {페퍼로니, 버섯}의 거래가 일어날 가능성은 0.15로 매우 낮음**
  - **따라서 이 연관규칙은 유용한 규칙이 아닐 수도 있음**

- 앞선 예제에서 보듯이 지지도와 신뢰도만으로 유용한 규칙인지 판단하기 어려울 경우가 발생
  - 따라서 향상도라는 측도를 고려
- 연관규칙 X⇒Y에 대한 향상도

$$\text{향상도} = \frac{P(X \cap Y)}{P(X)P(Y)} = \frac{\text{신뢰도}}{P(Y)}$$

$$= \frac{\text{품목 X와 Y를 포함하는 거래수}}{(\text{품목 X를 포함하는 거래수}) \times (\text{품목 Y를 포함하는 거래수})}$$

  - 품목 X가 주어지지 않았을 때의 품목 Y의 확률 대비 품목 X가 주어졌을 때의 품목 Y의 확률의 증가 비율
  - 이 값이 클수록 품목 X의 구매여부가 품목 Y의 구매여부에 큰 영향

- 연관규칙 X⇒Y은 품목 X와 품목 Y의 구매가 상호 관련이 없는 경우에는 P(Y|X)=P(Y)이므로 향상도가 1
  - 향상도가 1보다 크면 이 규칙은 결과를 예측하는데 있어서 우연적 기회보다 우수
  - 향상도가 1보다 작으면 이 규칙은 결과를 예측하는데 있어서 우연적 기회보다 나쁘다는 의미
  - 향상도=1이면 두 품목이 서로 독립적인 관계
    향상도<1이면 두 품목이 음의 상관 관계
    향상도>1이면 두 품목이 양의 상관 관계

연관규칙분석 절차

- k개의 품목이 있는 경우에는 2^k개의 가능한 품목 집합이 있고, k가 아주 큰 경우에 이 모든 집합 중에 지지도가 높은 집합을 찾는 것은 현실적으로 불가능
- Apriori 알고리즘
  - 최소 지지도를 갖는 연관규칙을 찾는 대표적인 방법
  - 최소 지지도 보다 큰 집합만을 대상으로 높은 지지도를 갖는 품목 집합을 찾는 것

- Apriori 알고리즘 단계
  - 최소 지지도를 넘는 모든 빈발품목집합을 생성
  - 빈발품목집합에서 최소 신뢰도를 넘는 모든 규칙을 생성
- 최소 지지도 이상의 빈발품목집합 생성 방법
  - 개별 품목 중에서 최소 지지도를 넘는 모든 품목을 발견
  - 앞에서 찾은 개별 품목만을 이용해서 최소 지지도를 넘는 2가지 품목 집합 발견
  - 앞선 두 단계에서 찾은 품목 집합을 결합하여 최소 지지도를 넘는 3가지 품목집합을 발견
  - 이런 방법을 반복적으로 사용하여 최소지지도가 넘는 빈발품목집합들을 발견

- 연관규칙의 생성
  - 빈발품목집합에 대하여 연관규칙을 생성하기 위해 공집합을 제외한 빈발품목집합의 모든 부분집합을 고려
  - 모든 부분집합에 대하여 연관규칙을 생성하고 신뢰도를 계산하여 주어진 최소 신뢰도를 넘는 연관규칙을 발견

예제 3.2

- 다음 거래내역에 대하여 최소 지지도가 30%인 경우에 Apriori 알고리즘을 이용하여 빈발품목집합을 생성하시오.

거래	품목
1	F, K, N
2	E, F
3	E, S
4	E, F, N
5	C, E, F, K, N
6	C, K, N
7	C, K, N

- 각 품목의 빈도

C	E	F	K	N	S
3	4	4	4	5	1

  - S를 제외한 모든 품목의 지지도가 0.3 이상
- 2-후보품목집합

CE	CF	CK	CN	EF	EK	EN	FK	FN	KN
1	1	3	3	3	1	2	2	3	4

  - {(C,K), (C,N), (E,F), (F,N), (K,N)}
- 3-후보품목집합
  - (E,N)집합이 없으므로 (EFN)은 포함되지 않음

CKN
3

연관규칙분석의 고려사항

- 적절한 품목 선택
  - 어떤 품목을 선택할 것인가는 전적으로 분석의 목적에 따라 선택
  - 탐색해야 할 규칙의 수는 고려되는 품목의 수에 따라 지수적으로 증가하므로 분석목적에 맞는 적절한 품목의 선택은 매우 중요
  - 연관규칙분석은 각 품목들의 거래횟수가 비슷한 경우에 가장 효율적

- 연관규칙의 발견
  - 연관규칙을 찾을 때 연관규칙을 어떻게 표현하는 가라는 문제도 매우 중요
  - 분석결과 향상도가 1보다 작은 경우에는 결과를 역으로 나타내면 향상도가 1보다 커지는데 이러한 규칙이 의미가 있는지 고려
- 현실적인 문제의 해결
  - 품목 수가 증가하면 계산량은 기하급수적으로 증가하게 되며 연관규칙을 탐색하는데에 많은 시간이 소요
  - 따라서 최소지지도 가지치기를 적용
  - 품목수가 일정수를 넘는 규칙은 고려대상에서 제외

- 거래내역 개체를 생성(arules 패키지)

```
> read.transactions(file, sep="")
```

  - file : 파일 위치
  - sep : 구분자
- 연관과 거래내역출력(arules 패키지)

```
> inspect(x)
```

  - x : 연관규칙 또는 거래내역, 품목행렬 개체
- 품목 도수/지지도(arules 패키지)

```
> itemFrequency(x, type="relative")
```

  - x : 연관 또는 거래내역, 품목행렬 개체
  - type : 출력값
    - "relative"(지지도), "absolute"(도수)

▪ 도수/지지도 막대도표(arules 패키지)

```
> itemFrequencyPlot(x, type="relative",
+                   support=NULL, topN=NULL,
+                   lift=FALSE, horiz=FALSE)
```

- x : 연관 또는 거래내역, 품목행렬 개체
- type : 출력값
  - "relative"(지지도), "absolute"(도수)
- support : 최소지지도
- topN : 빈도 또는 향상도 상위 N개
- horiz : 수평 막대도표 출력(TRUE)

▪ 거래내역 시각화(arules 패키지)

```
> image(x, xlab="Items (Columns)",
+       ylab="Elements (Rows)")
```

- x : 연관 또는 거래내역, 품목행렬 개체
- xlab, ylab : 도표의 x-축, y-축 이름표

▪ 연관규칙모형(arules 패키지)

```
> apriori(data, parameter=NULL)
```

- data : 연관 또는 거래내역, 품목행렬 개체
- parameter : 연관규칙 생성을 위한 모수
  - 최소지지도(support=0.1), 최소신뢰도 (confidence=0.8), 최대 아이템 수 (maxlen=10), 최소 아이템 수(minlen=10)

예제 3.3

- **다음은 어떤 마트에서 고객들이 구매한 식료품 목록이다. 이를 이용하여 연관규칙 분석을 수행하시오.**

	A	B	C	D	E	F
1	citrus fruit	semi-finished bread	margarine	ready soups		
2	tropical fruit	yogurt	coffee			
3	whole milk					
4	pip fruit	yogurt	cream cheese	meat spreads		
5	other vegetables	whole milk	condensed milk	long life bakery product		
6	whole milk	butter	yogurt	rice	abrasive cleaner	
7	rolls/buns					
8	other vegetables	UHT-milk	rolls/buns	bottled beer	liquor (appetizer)	
9	potted plants					
10	whole milk	cereals				
11	tropical fruit	other vegetables	white bread	bottled water	chocolate	
12	citrus fruit	tropical fruit	whole milk	butter	curd	yogurt

```
> # 식료품 데이터를 희소 매트릭스로 로드
> library(arules)
> groceries<-read.transactions("groceries.csv",
+                                    sep = ",")
> summary(groceries)
transactions as itemMatrix in sparse format with
 9835 rows (elements/itemsets/transactions) and
 169 columns (items) and a density of 0.02609146

most frequent items:
whole milk other vegetables   rolls/buns  soda   yogurt
      2513             1903        1809      1715    1372
   (Other)
     34055
```

element (itemset/transaction) length distribution:
sizes

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
2159	1643	1299	1005	855	645	545	438	350	246	182	117	78	77	55	46	29

18	19	20	21	22	23	24	26	27	28	29	32
14	14	9	11	4	6	1	1	1	1	3	1

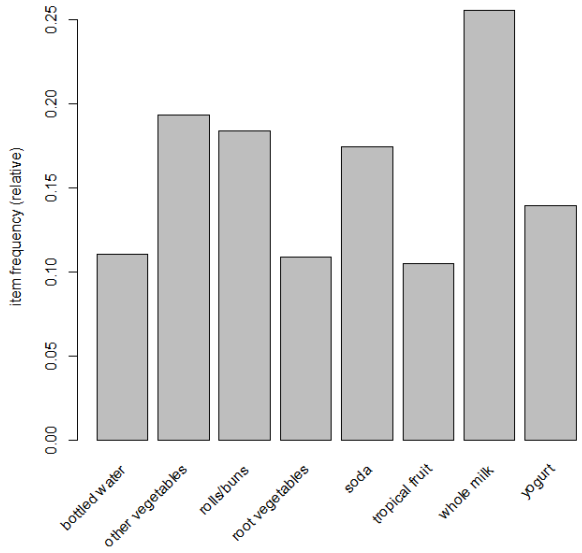
Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
1.000	2.000	3.000	4.409	6.000	32.000

includes extended item information - examples:

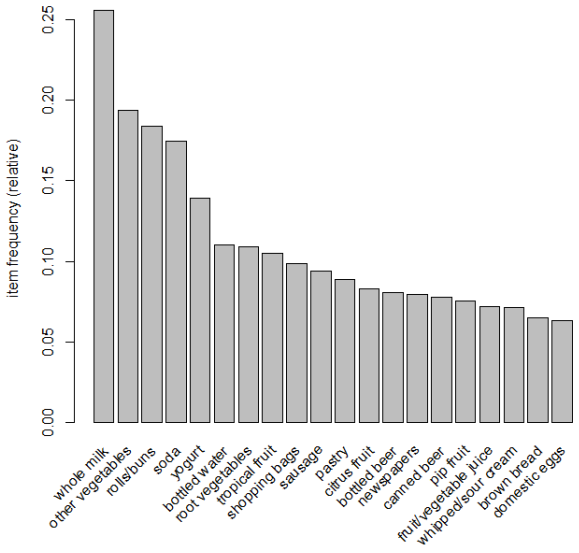
	labels
1	abrasive cleaner
2	artif. sweetener
3	baby cosmetics

```
> # 처음 5개 거래 확인
> inspect(groceries[1:5])
    items
[1] {citrus fruit,
     margarine,
     ready soups,
     semi-finished bread}
[2] {coffee,
     tropical fruit,
     yogurt}
[3] {whole milk}
[4] {cream cheese,
     meat spreads,
     pip fruit,
     yogurt}
[5] {condensed milk,
     long life bakery product,
     other vegetables,
     whole milk}
```

```
> # 3개 물품 빈도 확인
> itemFrequency(groceries[,1:3], type="relative")
abrasive cleaner artif. sweetener    baby cosmetics
  0.0035587189    0.0032536858      0.0006100661
> itemFrequency(groceries[,1:3], type="absolute")
abrasive cleaner artif. sweetener    baby cosmetics
            35                32                 6
>
> # 식료품의 빈도 시각화
> windows()
> itemFrequencyPlot(groceries, support=0.1)
> windows()
> itemFrequencyPlot(groceries, topN=20)
```

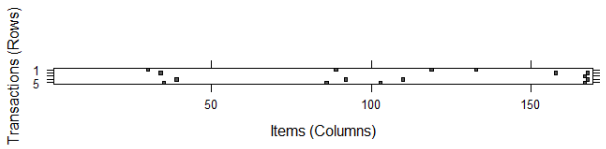


최소지지도 0.1 이상인 아이템

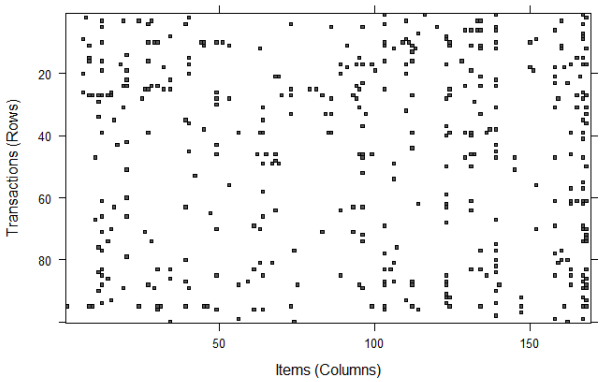


지지도 상위 20개 아이템

```
> # 처음 5개 거래에 대한 희소 매트릭스 시각화
> windows()
> image(groceries[1:5])
>
> # 100개 거래에 대한 무작위 샘플 시각화
> windows()
> image(sample(groceries, 10))
```



처음 5개 거래



임의로 선택된
100개 거래

```
> # 데이터에 대한 모델 훈련
> # 모델 훈련
> apriori(groceries)
Apriori

Parameter specification:
 confidence minval smax arem  aval originalSupport
        0.8    0.1    1 none FALSE            TRUE
maxtime support minlen maxlen
      5     0.1      1     10
 target   ext
  rules FALSE

Algorithmic control:
 filter tree heap memopt load sort verbose
    0.1 TRUE TRUE  FALSE TRUE    2    TRUE
```

```
Absolute minimum support count: 983

set item appearances ...[0 item(s)] done [0.00s].
set transactions ...[169 item(s), 9835 transaction(s)]
done [0.00s].
sorting and recoding items ... [8 item(s)] done
[0.00s].
creating transaction tree ... done [0.00s].
checking subsets of size 1 2 done [0.00s].
writing ... [0 rule(s)] done [0.00s].
creating S4 object  ... done [0.00s].
set of 0 rules
```

```
> # 규칙을 좀 더 학습히기 위해 지지도(support)와
> # 신뢰도(confidence) 설정 변경
> groceryrules<-apriori(groceries,
+      parameter=list(support=0.006,
+                              confidence=0.25, minlen = 2))
Apriori

Parameter specification:
confidence minval smax arem aval originalSupport
      0.25    0.1    1 none FALSE            TRUE
maxtime support minlen maxlen
      5   0.006       2      10
 target    ext
  rules FALSE
```

```
Algorithmic control:
 filter tree heap memopt load sort verbose
    0.1 TRUE TRUE   FALSE TRUE     2     TRUE

Absolute minimum support count: 59

set item appearances ...[0 item(s)] done [0.00s].
set transactions ...[169 item(s), 9835 transaction(s)]
done [0.00s].
sorting and recoding items ... [109 item(s)] done
[0.00s].
creating transaction tree ... done [0.00s].
checking subsets of size 1 2 3 4 done [0.00s].
writing ... [463 rule(s)] done [0.00s].
creating S4 object  ... done [0.00s].
```

```
> groceryrules
set of 463 rules
> # 모델 성능 평가
> # 식료품 연관 규칙의 요약
> summary(groceryrules)
set of 463 rules

rule length distribution (lhs + rhs):sizes
  2   3   4
150 297  16

   Min. 1st Qu.  Median   Mean 3rd Qu.    Max.
  2.000  2.000    3.000    2.711   3.000   4.000
```

```
summary of quality measures:
    support         confidence        lift            count
 Min.   :0.006101 Min.   :0.2500 Min.   :0.9932 Min.   : 60.0
 1st Qu.:0.007117 1st Qu.:0.2971 1st Qu.:1.6229 1st Qu.: 70.0
 Median :0.008744 Median :0.3554 Median :1.9332 Median : 86.0
 Mean   :0.011539 Mean   :0.3786 Mean   :2.0351 Mean   :113.5
 3rd Qu.:0.012303 3rd Qu.:0.4495 3rd Qu.:2.3565 3rd Qu.:121.0
 Max.   :0.074835 Max.   :0.6600 Max.   :3.9565 Max.   :736.0

mining info:
      data ntransactions support confidence
 groceries          9835   0.006       0.25
```

```
> # 처음 3개 규칙 확인
> inspect(groceryrules[1:3])
    lhs                    rhs
[1] {potted plants} => {whole milk}
[2] {pasta}         => {whole milk}
[3] {herbs}         => {root vegetables}
      support   confidence   lift  count
[1] 0.006914082 0.4000000  1.565460 68
[2] 0.006100661 0.4054054  1.586614 60
[3] 0.007015760 0.4312500  3.956477 69
```

- **whole milk 를 구매한 평균 소비자와 비교해 볼 때 potted plants 를 함께 구매한 소비자가 얼마나 더 있는지의 비율**

- **lift(X⇒Y) = confidence(X⇒Y)/support(Y) (X가 주어졌을 때 Y의 구매 확률)/(X가 주어지지 않았을 때의 Y의 구매 확률)**
- **lift값이 1 보다 크면 우연히 X,Y를 함께 구매했을 확률보다 크다는 의미**
- **lift 값이 크면 클수록 X,Y 의 구매 연관성이 높음**

```
> # lift로 규칙 정렬
> inspect(sort(groceryrules, by="lift")[1:5])
    lhs                          rhs
[1] {herbs}                   => {root vegetables}
[2] {berries}                 => {whipped/sour cream}
[3] {other vegetables,
     tropical fruit,
     whole milk}              => {root vegetables}
[4] {beef,
     other vegetables}        => {root vegetables}
[5] {other vegetables,
     tropical fruit}          => {pip fruit}
```

```
      support   confidence   lift    count
[1] 0.007015760  0.4312500 3.956477    69
[2] 0.009049314  0.2721713 3.796886    89
[3] 0.007015760  0.4107143 3.768074    69
[4] 0.007930859  0.4020619 3.688692    78
[5] 0.009456024  0.2634561 3.482649    93
```

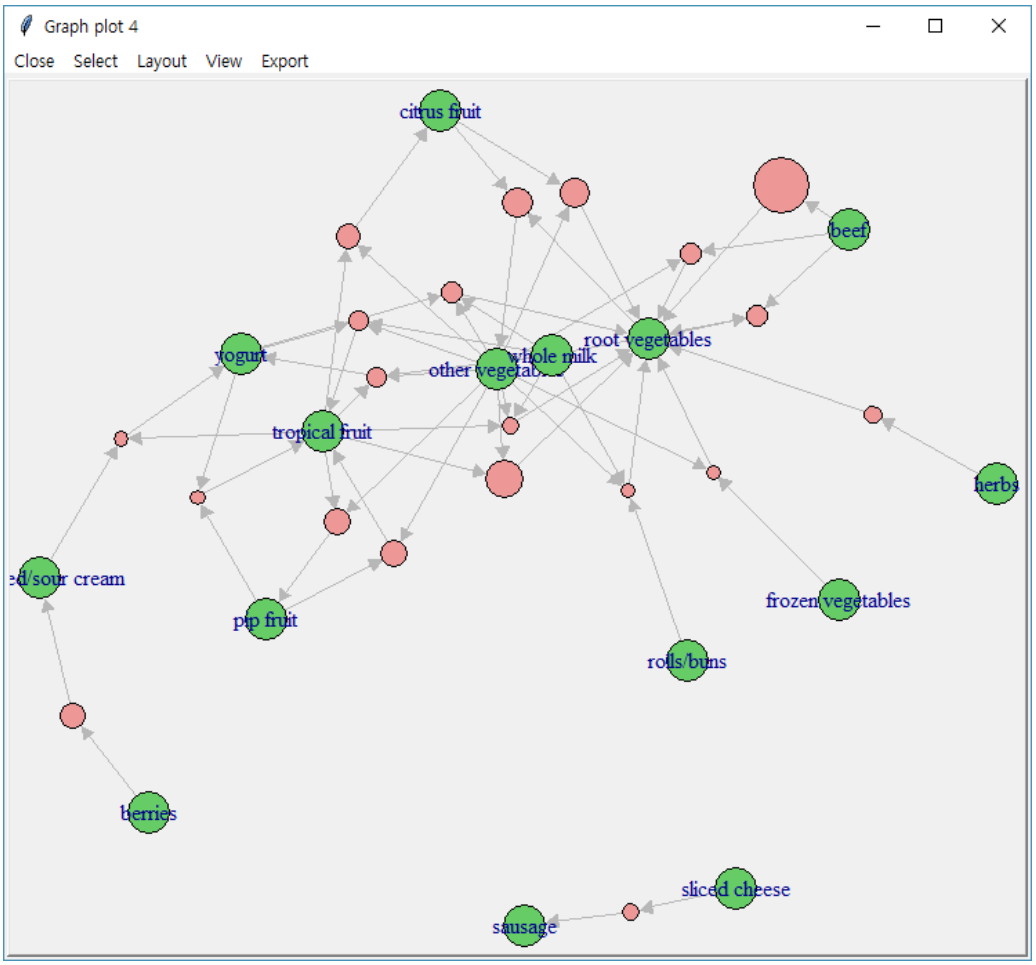
```
> # 딸기류 아이템을 포함하는 규칙의 부분 규칙
> berryrules<-subset(groceryrules,
+                    items %in% "berries")
> inspect(berryrules)
    lhs           rhs
[1] {berries} => {whipped/sour cream}
[2] {berries} => {yogurt}
[3] {berries} => {other vegetables}
[4] {berries} => {whole milk}
      support  confidence    lift  count
[1] 0.009049314 0.2721713  3.796886  89
[2] 0.010574479 0.3180428  2.279848 104
[3] 0.010269446 0.3088685  1.596280 101
[4] 0.011794611 0.3547401  1.388328 116
```

```
> # CSV 파일에 규칙 쓰기
> write(groceryrules,
+       file="groceryrules.csv", sep=",",
+       quote=TRUE, row.names=FALSE)
>
> # 규칙들을 데이터 프레임으로 변환
> groceryrules_df<-as(groceryrules, "data.frame")
> str(groceryrules_df)
'data.frame': 463 obs. of  5 variables:
 $ rules     : Factor w/ 463 levels "{baking powder} => {other
vegetables}",..: 340 302 207 206 208 341 402 21 139 140 ...
 $ support   : num  0.00691 0.0061 0.00702 0.00773
0.00773 ...
 $ confidence: num  0.4 0.405 0.431 0.475 0.475 ...
 $ lift      : num  1.57 1.59 3.96 2.45 1.86 ...
 $ count     : num  68 60 69 76 76 69 70 67 63 88 ...
```

```
> head(groceryrules_df)
                     rules
1    {potted plants} => {whole milk}
2              {pasta} => {whole milk}
3              {herbs} => {root vegetables}
4              {herbs} => {other vegetables}
5              {herbs} => {whole milk}
6 {processed cheese} => {whole milk}
     support   confidence    lift    count
1 0.006914082  0.4000000 1.565460    68
2 0.006100661  0.4054054 1.586614    60
3 0.007015760  0.4312500 3.956477    69
4 0.007727504  0.4750000 2.454874    76
5 0.007727504  0.4750000 1.858983    76
6 0.007015760  0.4233129 1.656698    69
```

	A	B	C	D	E
1	rules	support	confidence	lift	count
2	{potted plants} => {whole milk}	0.006914082	0.4	1.56545961	68
3	{pasta} => {whole milk}	0.006100661	0.405405405	1.58661447	60
4	{herbs} => {root vegetables}	0.00701576	0.43125	3.956477379	69
5	{herbs} => {other vegetables}	0.007727504	0.475	2.454873883	76
6	{herbs} => {whole milk}	0.007727504	0.475	1.858983287	76
7	{processed cheese} => {whole milk}	0.00701576	0.423312883	1.656698054	69
8	{semi-finished bread} => {whole milk}	0.007117438	0.402298851	1.574456504	70
9	{beverages} => {whole milk}	0.006812405	0.26171875	1.024275331	67
10	{detergent} => {other vegetables}	0.006405694	0.333333333	1.722718515	63
11	{detergent} => {whole milk}	0.008947636	0.465608466	1.822228117	88
12	{pickled vegetables} => {other vegetables}	0.006405694	0.357954545	1.849964769	63
13	{pickled vegetables} => {whole milk}	0.007117438	0.397727273	1.556564953	70
14	{baking powder} => {other vegetables}	0.007320793	0.413793103	2.138547122	72
15	{baking powder} => {whole milk}	0.009252669	0.522988506	2.046793456	91
16	{flour} => {other vegetables}	0.006304016	0.362573099	1.873834174	62
17	{flour} => {whole milk}	0.008439248	0.485380117	1.899607422	83
18	{soft cheese} => {other vegetables}	0.007117438	0.416666667	2.153398143	70
19	{soft cheese} => {whole milk}	0.007524148	0.44047619	1.723869213	74
20	{specialty bar} => {soda}	0.007219115	0.26394052	1.513618087	71
21	{misc. beverages} => {soda}	0.007320793	0.258064516	1.479921001	72
22	{grapes} => {tropical fruit}	0.006100661	0.272727273	2.59910148	60

```
> # extra visualization
> library(arulesViz)
> plot(sort(groceryrules, by="lift")[1:20],
method="graph", interactive=TRUE, shading=NA)
Warning message:
In plot.rules(sort(groceryrules, by = "lift")[1:20],
method = "graph",  :
  The parameter interactive is deprecated. Use
engine='interactive' instead.
```



```
> plot(sort(groceryrules, by="lift")[1:20],
+           method="graph", interactive=TRUE,
+           shading=NA)
> inspect(sort(groceryrules, by="lift")[1:5])
    lhs                         rhs
[1] {herbs}                  => {root vegetables}
[2] {berries}                => {whipped/sour cream}
[3] {other vegetables,
     tropical fruit,
     whole milk}             => {root vegetables}
[4] {beef,
     other vegetables}       => {root vegetables}
[5] {other vegetables,
     tropical fruit}         => {pip fruit}
```

```
      support     confidence     lift     count
[1] 0.007015760  0.4312500 3.956477    69
[2] 0.009049314  0.2721713 3.796886    89
[3] 0.007015760  0.4107143 3.768074    69
[4] 0.007930859  0.4020619 3.688692    78
[5] 0.009456024  0.2634561 3.482649    93
>
> detach("package:arulesViz", unload=TRUE)
> detach("package:arules", unload=TRUE)
```