

모두를 위한 R 데이터 분석 입문

2판



Chapter 10

워드클라우드와 구매 패턴 분석



목차

1. 워드클라우드 분석
2. 구매 패턴 분석
3. 인터넷 검색어 분석
4. 공공 빅데이터

Section 02

구매 패턴 분석

2. 구매 패턴 분석

- 상품의 유통, 판매 분야는 데이터 분석이 활발히 적용되는 분야중의 하나
- 계산대 부근에 껌이나 캔디류, 건전지 등이 진열되어 있는 것은 우연이 아니고 소비자의 구매 행태에 대한 철저한 분석의 결과
- 소비자의 구매 패턴(행태) 분석은 장바구니 분석(market basket analysis)으로도 알려져 있음



(이미지 출처: <https://pixabay.com/>)

2. 구매 패턴 분석

1. 연관 규칙

- **연관 규칙(association rule)** : 데이터 안에 포함된 일정한 패턴
- 구매 데이터에서 찾을 수 있는 연관 규칙의 예

“맥주를 사는 사람은 땅콩도 함께 구매한다”

“분유를 사는 사람은 기저귀도 함께 구매한다”

- 구매 패턴의 표현

{맥주} → {땅콩}

{분유} → {기저귀}

- 구매 패턴은 영수증을 분석하면 알 수 있다.

2. 구매 패턴 분석

2. 어프리오리 알고리즘

- **어프리오리(Apriori) 알고리즘:** 연관규칙 분석에 널리 이용되는 머신러닝 기법중의 하나로, 1994년 Agrawal 와 Srikant에 의해 제안됨.
- **구매 행렬:** 구매 내역에서 연관된 구매 상품을 찾는 가장 쉬운 방법

표 10-1 구매 내역 예제

거래 번호	구매 상품
1	맥주, 땅콩
2	맥주, 오징어
3	맥주, 라면, 땅콩
4	초콜릿, 껌
5	초콜릿, 생수, 껌

표 10-2 구매 행렬

상품	맥주	땅콩	오징어	라면	초콜릿	껌	생수
맥주	—	2	1	1	0	0	0
땅콩	2	—	1	0	0	0	0
오징어	1	1	—	0	0	0	0
라면	1	0	0	—	0	0	0
초콜릿	0	0	0	0	—	2	1
껌	0	0	0	0	2	—	1
생수	0	0	0	0	1	1	—

거래 단위: transaction
손님당 여러 물품 구매

2. 구매 패턴 분석

▪ 지지도(support):

상품 X,Y를 함께 구매한 비율이 전체 거래에서 차지하는 비율을 측정하는 척도

- $\text{support}(X \rightarrow Y)$, $\text{support}(Y \rightarrow X)$, $\text{support}(X, Y)$ 모두 같은 의미

$$\text{support}(\{X\} \rightarrow \{Y\}) = \frac{X, Y \text{를 함께 포함한 거래건수}}{\text{전체 거래건수}}$$

▪ {맥주}→{땅콩}의 지지도

$$\text{support}(\{\text{맥주}\} \rightarrow \{\text{땅콩}\}) = \frac{2}{5} = 0.4$$

Coverage: "조건(lhs)이 등장한 전체 비율"

맥주가 출현한 거래 비율 3/5

이 값이 클수록 조건(lhs)의 발생 빈도가 높다는 의미
즉, 자주 등장하는 조건일수록 높은 coverage 가짐

표 10-1 구매 내역 예제

거래 번호	구매 상품
1	맥주, 땅콩
2	맥주, 오징어
3	맥주, 라면, 땅콩
4	초콜릿, 껌
5	초콜릿, 생수, 껌

2. 구매 패턴 분석

▪ 신뢰도(confidence): 조건부 확률을 의미

상품 X를 구매했다는 전제하에 상품 X와 Y를 동시에 구매한 빈도수를 계산하는 척도

$$cconfidence(\{X\} \rightarrow \{Y\}) = \frac{X, Y \text{를 포함한 거래건수}}{X \text{를 포함한 거래건수}}$$

▪ {맥주} → {땅콩}의 신뢰도

맥주와 땅콩 거래 건수

분모의 비율을 coverage(조건 출현률, 조건 항목 등장 비율)라고도 부름

$$confidence(\{맥주\} \rightarrow \{땅콩\}) = \frac{2}{3} = 0.67$$

맥주 거래 건수

▪ {땅콩} → {맥주}의 신뢰도

$$confidence(\{땅콩\} \rightarrow \{맥주\}) = \frac{2}{2} = 1$$

땅콩 거래 건수

표 10-1 구매 내역 예제

거래 번호	구매 상품
1	맥주, 땅콩
2	맥주, 오징어
3	맥주, 라면, 땅콩
4	초콜릿, 껌
5	초콜릿, 생수, 껌

맥주를 산 경우에는 '많은 경우' 땅콩도 함께 사지만,
땅콩을 산 경우는 '반드시' 맥주를 함께 산다

2. 구매 패턴 분석

X를 구매한 사람이 Y를 구매할 확률과
X의 구매와 상관없이 Y를 구매할 확률의 비

$cconfidence(\{X\} \rightarrow \{Y\})$

- **향상도(lift):** 연관 규칙 $\{X\} \rightarrow \{Y\}$ 에서 X를 구매했을 때 Y를 구매한 비율이 그러한 조건이 없던 때(그냥 Y를 구매한 비율)에 비해 얼마나 증가하는가를 보여주는 척도
 - 값이 1보다 크면 X를 샀을 때 Y를 살 확률이 높은 것을 의미
 - 값이 1 미만이면 X를 샀을 때 Y를 사지않을 확률이 높은 것을 의미
 - 향상도가 1이면 X를 산 것과 Y를 산 것은 관계가 없다는 의미

$support(\{Y\})$

$$lift(\{X\} \rightarrow \{Y\}) = \frac{confidence(\{X\} \rightarrow \{Y\})}{support(\{Y\})}$$

X를 구매 했을 경우,
Y도 구매한 비율

Y를 구매한 비율

- {맥주}→{땅콩}의 향상도

$$lift(\{맥주\} \rightarrow \{땅콩\}) = \frac{2/3}{2/5} = 1.67$$

X(맥주)를 구매했을 때
Y(땅콩)를 구매한 비율

Y(땅콩)를 구매한 비율

맥주를 살 때 땅콩을 구매하는 빈도가
땅콩을 사는 것보다 1.67배 높다

연관분석 지표 계산 문제

아래의 문제를 통해 연관규칙 알고리즘을 이해해보자.



거래번호	거래 아이템
1	우유, 버터, 시리얼
2	우유, 시리얼
3	우유, 빵
4	버터, 맥주, 오징어

문제 1. 지지도(support)

$$s(\text{우유}, \text{시리얼}) = ?, s(\{\text{우유}\} \rightarrow \{\text{시리얼}\}) = ?, s(\{\text{시리얼}\} \rightarrow \{\text{우유}\}) = ?$$

문제 2. 신뢰도(confidence)

$$c(\{\text{우유}\} \rightarrow \{\text{시리얼}\}) = s(\text{우유}, \text{시리얼}) / s(\text{우유}) = ?$$

문제 3. 향상도

$$\text{lift}(\{\text{우유}\} \rightarrow \{\text{시리얼}\}) = c(\{\text{우유}\} \rightarrow \{\text{시리얼}\}) / s(\text{시리얼}) = ?$$

연관분석 지표 계산

아래의 문제를 통해 연관규칙 알고리즘을 이해해보자.



거래번호	거래 아이템
1	우유, 버터, 시리얼
2	우유, 시리얼
3	우유, 빵
4	버터, 맥주, 오징어

문제 1. 지지도

$$s(\text{우유}, \text{시리얼}) = n(X \cap Y) / N = 2/4 = 1/2$$

문제 2. 신뢰도

$$c(\text{우유} \rightarrow \text{시리얼}) = n(X \cap Y) / n(X) = n(\text{우유}, \text{시리얼}) / n(\text{우유}) = 1/2 / 3/4 = 2/3$$

문제 3. 향상도

$$\text{lift}(\text{우유} \rightarrow \text{시리얼}) = c(\text{우유} \rightarrow \text{시리얼}) / s(\text{시리얼}) = (2/3) / (2/4) = 1.333$$

우유를 살 때 시리얼을 구매하는 빈도가
시리얼을 사는 것보다 1.333배 높다

연관분석 지표 계산 (간단)

아래의 문제를 통해 연관규칙 알고리즘을 이해해보자.



거래번호	거래 아이템
1	우유, 버터, 시리얼
2	우유, 시리얼
3	우유, 빵
4	버터, 맥주, 오징어

문제 1. 지지도

$$s(\text{우유}, \text{시리얼}) = n(X \cap Y) / N = 2/4 = 1/2$$

문제 2. 신뢰도

$$c(\text{우유} \rightarrow \text{시리얼}) = n(X \cap Y) / n(X) = n(\text{우유}, \text{시리얼}) / n(\text{우유}) = 1/2 / 3/4 = 2/3$$

문제 3. 향상도(건수로 계산하는 방법)

$$\text{lift}(\text{우유} \rightarrow \text{시리얼}) =$$

$$\begin{aligned} & (\text{우유와 시리얼을 함께 산 거래건수}) * \text{총 건수} / (\text{우유 건수}) * (\text{시리얼 건수}) \\ & = (2 * 4) / (3 * 2) = 8 / 6 = 1.333 \end{aligned}$$

우유를 살 때 시리얼을 구매하는 빈도가
시리얼을 사는 것보다 1.333배 높다

2. 구매 패턴 분석

3. 구매 패턴의 분석 과정

- 아프리오리 알고리즘: "arules" 패키지 이용
- 실습 결과의 시각화: "arulesViz" 패키지 이용
- 실습용 데이터셋: Kaggle에서 제공하는 제과점 거래 데이터(BreadBasket_DMS.csv)
(<https://www.kaggle.com/datasets/sulmansarwar/transactions-from-a-bakery>)
- BreadBasket 데이터셋은 어떤 제과점의 1년간 거래(판매) 내역을 정리한 것
- Date, Time, Transaction, Item으로 구성: 중복도 있고, 상품에 NONE이 있음

Date	Time	Transaction	Item
2016-10-30	9:58:11	1	Bread
2016-10-30	10:05:34	2	Scandinavian
2016-10-30	10:05:34	2	Scandinavian
2016-10-30	10:07:57	3	Hot chocolate
2016-10-30	10:07:57	3	Jam
2016-10-30	10:07:57	3	Cookies
2016-10-30	10:08:41	4	Muffin
2016-10-30	10:13:03	5	Coffee
2016-10-30	10:13:03	5	Pastry
2016-10-30	10:13:03	5	Bread
2016-10-30	10:16:55	6	Medialuna
2016-10-30	10:16:55	6	Pastry
2016-10-30	10:16:55	6	Muffin
2016-10-30	10:19:12	7	Medialuna
2016-10-30	10:19:12	7	Pastry
2016-10-30	10:19:12	7	Coffee
2016-10-30	10:19:12	7	Tea
2016-10-30	10:20:51	8	Pastry
2016-10-30	10:20:51	8	Bread
2016-10-30	10:21:59	9	Bread
2016-10-30	10:21:59	9	Muffin
2016-10-30	10:25:58	10	Scandinavian
2016-10-30	10:25:58	10	Medialuna

2. 구매 패턴 분석

2.1 데이터 준비와 관찰하기

코드 10-5 (계속)

```
library(arules)          # 아프리오리 알고리즘
library(arulesViz)       # 연관규칙 시각화 도구

# 데이터 불러오기와 관찰
setwd("D:/source")
ds <- read.csv("BreadBasket_DMS.csv") # 거래 데이터 읽기
str(ds)
head(ds)
unique(ds$item)

# 'NONE' item 삭제
ds.new <- subset(ds, item != 'NONE')
write.csv(ds.new, "BreadBasket_DMS_upd.csv", row.names = F )
```

2. 구매 패턴 분석

2.1 데이터의 준비와 관찰

코드 10-5

```
# 트랜잭션 포맷으로 데이터 읽기
trans <- read.transactions("BreadBasket_DMS_upd.csv",
                           format="single", header=T,
                           cols=c(3,4), sep=";", rm.duplicates=T)

trans                                     # 트랜잭션 데이터 요약정보
dimnames(trans)[[2]]                   # 상품 목록 확인
toLongFormat(trans)                    # 거래별 상품 목록
inspect(head(trans, 10))                # 앞부분 10개 트랜잭션 출력
```

거래별 항목 하나가 있는 것을 거래별 여러 항목으로 정리한 결과

다시 바내로: 거래별 여러 항목으로 정리한 결과를 거래별 항목 하나가 있는 것으로

2. 구매 패턴 분석

인자 format="single"

구분	"single" 형식	"basket" 형식
구조	한 줄 = 하나의 아이템	한 줄 = 하나의 거래 전체
필요 컬럼	Transaction ID, Item	Item1, Item2, Item3...
예시 파일 내용	1,Bread 1,Coffee 2,Medialuna	Bread,Coffee Medialuna,Scandinavian
cols 인자	사용함 (cols = c(1,2))	보통 사용 안 함
주 용도	POS 데이터, DB 추출 결과	정제된 수기 데이터, 장바구니 예제

2. 구매 패턴 분석

```
> library(arules)                # 아프리오리 알고리즘  
> library(arulesViz)             # 연관규칙 시각화 도구
```

```
> setwd("D:/source")  
> ds <- read.csv("BreadBasket_DMS.csv")  # 거래 데이터 읽기
```

```
> str(ds)  
'data.frame':21293 obs. of  4 variables:  
 $ Date      : chr  "2016-10-30" "2016-10-30" "2016-10-30" "2016-10-30" ...  
 $ Time      : chr  "09:58:11" "10:05:34" "10:05:34" "10:07:57" ...  
 $ Transaction: int   1 2 2 3 3 3 4 5 5 5 ...  
 $ Item      : chr  "Bread" "Scandinavian" "Scandinavian" "Hot chocolate" ...
```

2. 구매 패턴 분석

```
> head(ds)
```

	Date	Time	Transaction	Item
1	2016-10-30	09:58:11	1	Bread
2	2016-10-30	10:05:34	2	Scandinavian
3	2016-10-30	10:05:34	2	Scandinavian
4	2016-10-30	10:07:57	3	Hot chocolate
5	2016-10-30	10:07:57	3	Jam
6	2016-10-30	10:07:57	3	Cookies

2. 구매 패턴 분석

```
> unique(ds$Item)
[1] "Bread"                "Scandinavian"
[3] "Hot chocolate"        "Jam"
[5] "Cookies"              "Muffin"
[7] "Coffee"                "Pastry"
[9] "Medialuna"             "Tea"
[11] "NONE"                  "Tartine"
...(중간 생략)
[89] "Argentina Night"      "Half slice Monster "
[91] "Gift voucher"          "Cherry me Dried fruit"
[93] "Mortimer"              "Raw bars"
[95] "Tacos/Fajita"
```

```
> # 'NONE' item 삭제
> ds.new <- subset(ds, Item != 'NONE')
> write.csv(ds.new, "BreadBasket_DMS_upd.csv", row.names =F )
```

```
> # 트랜잭션 포맷으로 데이터 읽기
> trans <- read.transactions("BreadBasket_DMS_upd.csv", format="single",
+                             header=T, cols=c(3,4), sep="," , rm.duplicates=T)
```

2. 구매 패턴 분석

- `"BreadBasket_DMS_upd.csv"`

읽어올 트랜잭션(거래) 데이터가 저장된 파일을 지정한다.

- `format="single"`

읽어올 파일의 포맷을 지정한다.

- `"single"`

예제 파일과 같이 한 줄에 하나의 상품만 저장된 경우(즉, 하나의 거래 데이터가 여러 줄에 걸쳐 저장)

- `header=T`

읽어올 파일의 첫째 줄이 열의 변수명인지를 지정한다.

- `cols=c(3,4)`

파일에서 읽어올 열을 지정한다(3번째(트랜잭션 ID)와 4번째(상품) 열만 읽음).

- `sep=","`

파일에서 열과 열의 구분자가 무엇인지 지정한다(예제 파일은 CSV 포맷이므로 구분자가 ","이다).

- `rm.duplicates=T`

동일 트랜잭션 안에 중복된 상품이 있는 경우 중복을 제거할 것인지 지정한다.

2. 구매 패턴 분석

```
> trans # 트랜잭션 데이터 요약 정보
transactions in sparse format with
9465 transactions (rows) and
94 items (columns)
```

```
> dimnames(trans)[[2]] # 상품 목록 확인
[1] "Adjustment" "Afternoon with the baker"
[3] "Alfajores" "Argentina Night"
[5] "Art Tray" "Bacon"
[7] "Baguette" "Bakewell"
...(중간 생략)
[87] "Tiffin" "Toast"
[89] "Truffles" "Tshirt"
[91] "Valentine's card" "Vegan Feast"
[93] "Vegan mincepie" "Victorian Sponge"
```

2. 구매 패턴 분석

프랜잭션 조회 함수

함수명	기능 설명
summary(trans)	전체 거래 수, 아이템 수, 밀도, 거래당 아이템 수 요약 등
length(trans)	거래 수 (rows)
size(trans)	각 거래마다 포함된 아이템 개수
itemLabels(trans)	전체 아이템 목록 반환
LIST(trans)	거래당 포함된 아이템을 리스트 형태로 반환
inspect(trans[1:5])	앞 5개 거래 상세 출력
image(trans)	희소행렬 형태 시각화 (많을 땐 그림)

2. 구매 패턴 분석

프랜잭션 요약

> summary(trans)

```
> summary(trans)
transactions as itemMatrix in sparse format with
 9465 rows (elements/itemsets/transactions) and
 94 columns (items) and a density of 0.02122827

most frequent items:
  Coffee   Bread     Tea    Cake  Pastry (Other)
    4528    3097    1350    983     815     8114

element (itemset/transaction) length distribution:
sizes
   1    2    3    4    5    6    7    8    9   10
3948 3059 1471  662  234   64   17    4    5    1

      Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
      1.000   1.000   2.000   1.995   3.000  10.000

includes extended item information - examples:
      labels
1           Adjustment
2 Afternoon with the baker
3           Alfajores

includes extended transaction information - examples:
transactionID
1             1
2             10
3            100
> |
```

항목(상품) 정보로
labels가 있음

거래 정보로
transactionID가 있음

2. 구매 패턴 분석

프랜잭션 요약

➤ summary(trans)

transactions as itemMatrix in sparse format with
9465 rows (elements/itemsets/transactions) and
94 columns (items) and a density of 0.02122827

거래 ID	bread	milk	beer	...
1	1	0	1	...
2	0	1	0	...
...				

- 이 객체는 transactions 클래스인데, 내부적으로 itemMatrix 클래스로 저장
 - 저장 형식은 sparse matrix (희소 행렬): 대부분이 0인 행렬이라 공간 절약을 위해 비어 있는 값은 저장하지 않음
- 각 거래(transaction)가 행, 각 아이템(item)이 열
 - 총 9465개의 거래(transaction)
- 고유 아이템 종류가 94개
 - 각 열은 하나의 상품을 의미하고, 각 거래는 그 중 일부를 1 또는 0으로 표현

2. 구매 패턴 분석

```
> toLongFormat(trans)
```

```
# 거래별 상품 목록
```

	TID	item
1	1	Bread
2	2	Medialuna
3	2	Scandinavian
4	3	Bread
5	4	Chimichurri Oil
6	4	Scandinavian
7	5	Bread
8	5	Truffles
9	6	Brownie
...(이하 생략)		

2. 구매 패턴 분석

```
> inspect(head(trans, 10))
```

```
# 앞부분 10개 트랜잭션 출력
```

	items	transactionID
[1]	{Bread}	1
[2]	{Medialuna, Scandinavian}	10
[3]	{Bread}	100
[4]	{Chimichurri Oil, Scandinavian}	1000
[5]	{Bread, Truffles}	1001
[6]	{Brownie, Focaccia}	1002
[7]	{Bread, Coffee}	1003
[8]	{Art Tray, Coffee, Cookies, Tea}	1004
[9]	{Coffee}	1005
[10]	{Bread}	1006

```
> toLongFormat(trans)
```

```
# 거래별 상품 목록
```

	TID	item
1	1	Bread
2	2	Medialuna
3	2	Scandinavian
4	3	Bread
5	4	Chimichurri Oil
6	4	Scandinavian
7	5	Bread
8	5	Truffles
9	6	Brownie
...(이하 생략)		

TID는 trans의 순번

2. 구매 패턴 분석

2.2 연관 규칙의 검색과 시각화

코드 10-6 (계속)

```
# 상품 판매 빈도
```

```
itemFrequencyPlot(trans, topN=10, type="absolute", xlab="상품명",  
  ylab="절대 판매빈도", main="판매량 많은 상품", col="green")
```

```
itemFrequencyPlot(trans, topN=10, type="relative", xlab="상품명",  
  ylab="상대 판매빈도", main="판매량 많은 상품", col="blue")
```

```
# 연관규칙 찾기
```

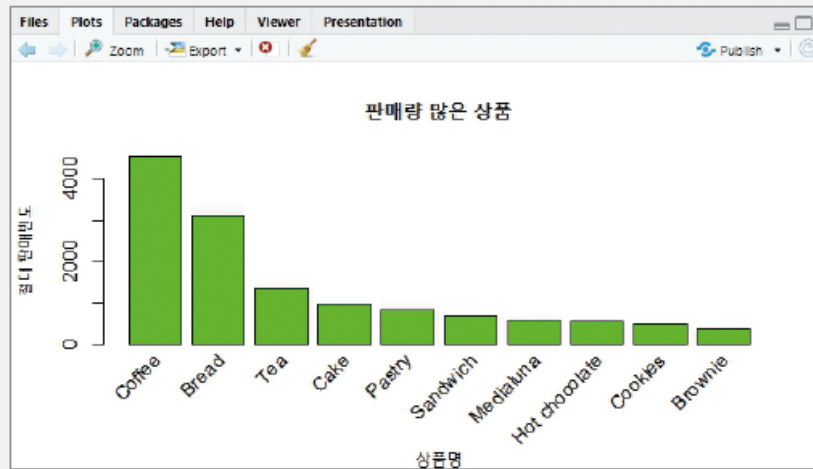
```
rules <- apriori(trans, parameter = list(supp = 0.001, conf = 0.7))  
rules
```

```
# 앞쪽 10개의 규칙 출력
```

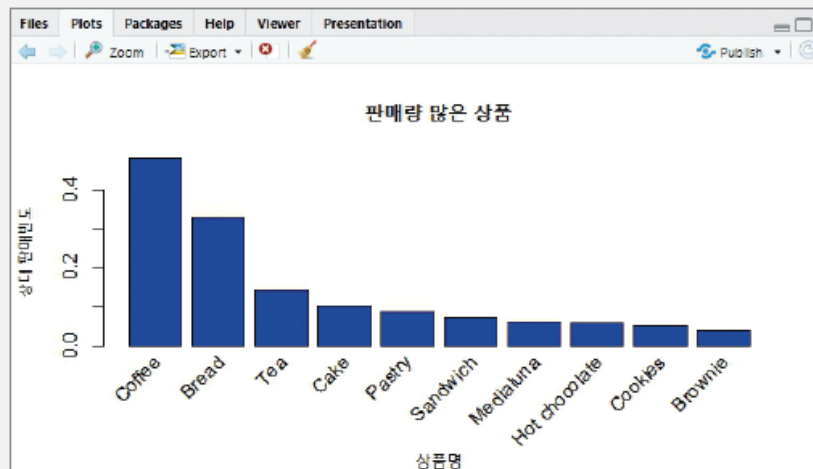
```
options(digits=2)          # 평가 척도 값의 자릿수 지정  
inspect(rules[1:10])
```

2. 구매 패턴 분석

```
> itemFrequencyPlot(trans, topN=10, type="absolute", xlab="상품명",  
+ ylab="절대 판매빈도", main="판매량 많은 상품", col="green")
```



```
> itemFrequencyPlot(trans, topN=10, type="relative", xlab="상품명",  
+ ylab="상대 판매빈도", main="판매량 많은 상품", col="blue")
```



2. 구매 패턴 분석

```
> # 연관규칙 찾기
```

```
> rules <- apriori(trans, parameter = list(supp = 0.001, conf = 0.7))
```

```
Apriori
```

Parameter specification:

confidence	minval	smax	arem	aval	originalSupport	maxtime	support	minlen
0.7	0.1	1	none	FALSE	TRUE	5	0.001	1
maxlen target		ext						
10 rules		TRUE						

Algorithmic control:

filter	tree	heap	memopt	load	sort	verbose
0.1	TRUE	TRUE	FALSE	TRUE	2	TRUE

Absolute minimum support count: 9

```
set item appearances ...[0 item(s)] done [0.00s].
```

```
set transactions ...[94 item(s), 9465 transaction(s)] done [0.00s].
```

```
sorting and recoding items ... [57 item(s)] done [0.00s].
```

```
creating transaction tree ... done [0.00s].
```

```
checking subsets of size 1 2 3 4 done [0.00s].
```

```
writing ... [14 rule(s)] done [0.00s].
```

```
creating S4 object ... done [0.00s].
```

```
> rules
```

```
set of 14 rules
```

- ``trans``: 입력 데이터셋 (transaction 데이터)
- ``parameter``: 연관 규칙을 생성할 때 사용할 파라미터 목록
 - ``supp = 0.001``: 최소 지지도 (support)를 0.1%로 설정
 - ``conf = 0.7``: 최소 신뢰도 (confidence)를 70%로 설정

- ``support``: 지지도 기준 (0.001)
- ``minlen``: 최소 항목 수 (1)
- ``maxlen``: 최대 항목 수 (10)
- ``target``: 생성할 대상 (rules, 기본값)

절대 최소 지지도는 9: 9465건의 트랜잭션 중 $0.001 * 9465 \approx 9$ 건 이상 등장해야 한다는 의미

57개의 상품으로 14개의 규칙이 생성

다음 R 코드 결과를 자세히 설명해 줘

```
> rules <- apriori(trans, parameter = list(supp = 0.001, conf = 0.7))
Apriori
```

Parameter specification:

```
confidence minval smax arem  aval originalSupport maxtime
      0.7    0.1    1 none FALSE      TRUE     5
support minlen maxlen target  ext
  0.001     1     10  rules TRUE
```

Algorithmic control:

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

Absolute minimum support count: 9

```
set item appearances ...[0 item(s)] done [0.00s].
set transactions ...[94 item(s), 9465 transaction(s)] done [0.00s].
sorting and recoding items ... [57 item(s)] done [0.00s].
creating transaction tree ... done [0.00s].
checking subsets of size 1 2 3 4 done [0.00s].
writing ... [14 rule(s)] done [0.00s].
creating S4 object ... done [0.00s].
```

```
> rules <- apriori(trans, parameter = list(supp = 0.001, conf = 0.7))
```

Apriori

Parameter specification:

confidence	minval	smax	arem	aval	originalSupport	maxtime	support	minlen	maxlen	target	ext
0.7	0.1	1	none	FALSE	TRUE	5	0.001	1	10	rules	TRUE

주요 파라미터:


- `confidence`: 최소 신뢰도 (Confidence) 설정.
 - 연관 규칙의 신뢰도가 0.7(70%) 이상이어야 규칙으로 포함.
- `support`: 최소 지지도 (Support) 설정.
 - 지지도 값이 0.001(0.1%) 이상인 규칙만 고려.
- `minlen`: 규칙의 최소 길이.
 - 최소 1개의 아이템을 포함한 규칙을 생성.
- `maxlen`: 규칙의 최대 길이.
 - 최대 10개의 아이템으로 이루어진 규칙을 생성.
- `target`: 생성할 대상.
 - `rules`: 연관 규칙을 생성.
- `maxtime`: 알고리즘 실행 시간 제한(초).
 - 5초로 설정(기본값).

기타:

- `originalSupport`: 트랜잭션의 원래 지지도를 기준으로 규칙을 생성할지 여부.
 - `TRUE`: 원래 지지도를 사용.
- `ext`: 확장된 규칙을 허용 여부.
 - `TRUE`: 확장된 규칙을 포함.

2. Algorithmic Control: 알고리즘 제어 설정

R

 코드 복사

Algorithmic control:

filter tree heap memopt load sort verbose

0.1 TRUE TRUE FALSE TRUE 2 TRUE

- `filter`: 항목 필터링 임계값(설정값: 0.1).
- `tree`: 트랜잭션 트리를 사용하여 효율적인 탐색 수행 여부(`TRUE`).
- `heap`: 힙 기반 메모리 최적화를 활성화 여부(`TRUE`).
- `memopt`: 메모리 최적화 옵션 (`FALSE` = 비활성화).
- `sort`: 아이템 정렬 수준 (설정값: 2).
- `verbose`: 실행 중 상세 정보를 출력 여부 (`TRUE`).

3. 절대 최소 지지도

R

 코드 복사

```
Absolute minimum support count: 9
```

- 의미:
 - 설정된 지지도(`supp = 0.001`)는 트랜잭션 개수(9465)의 0.1%에 해당.
 - $0.001 \times 9465 = 9$ 건 이상 나타난 규칙만 고려.

4. 실행 과정 설명

```
R

set item appearances ...[0 item(s)] done [0.00s].
set transactions ...[94 item(s), 9465 transaction(s)] done [0.00s].
```

코드 복사

- 데이터셋 준비:
 - 94개의 항목과 9465개의 트랜잭션을 처리.
 - `item appearances` 는 특정 항목에 대한 출현 조건이 없는 경우 0으로 표시.

```
R

sorting and recoding items ... [57 item(s)] done [0.00s].
```

코드 복사

- 항목 정렬 및 재코딩:
 - 빈도가 너무 낮은 항목을 제외하고, 57개의 유효 항목으로 축소.

```
R

creating transaction tree ... done [0.00s].
```

코드 복사

- 트랜잭션 트리 생성:
 - Apriori 알고리즘은 트랜잭션 트리를 생성하여 효율적으로 빈발 항목 집합을 탐색.

```
R

checking subsets of size 1 2 3 4 done [0.00s].
```

코드 복사

- 빈발 항목 집합 탐색:
 - 1개, 2개, 3개, 4개로 구성된 항목 집합의 빈발 항목 탐색 완료.

```
R

writing ... [14 rule(s)] done [0.00s].
```

코드 복사

- 연관 규칙 생성:
 - 설정한 조건(`supp = 0.001`, `conf = 0.7`)을 만족하는 14개의 규칙이 생성.

```
R

creating S4 object ... done [0.00s].
```

코드 복사

- S4 객체 생성:
 - 생성된 규칙을 S4 객체 형식으로 저장(`rules` 객체).

5. 주요 결과


데이터 요약

- 트랜잭션 수: 9465개.
- 항목 수: 94개 (유효 항목 57개).
- 최소 규칙 조건: 지지도 ≥ 0.001 (9건), 신뢰도 ≥ 0.7 .
- 생성된 규칙: 14개.

생성된 규칙 확인

규칙은 다음과 같이 확인할 수 있습니다:


R

 코드 복사

```
inspect(rules)
```

출력 예:

csharp


 코드 복사

	lhs	rhs	support	confidence	lift
[1]	{Bread}	=> {Coffee}	0.015	0.75	1.20
[2]	{Medialuna}	=> {Tea}	0.012	0.85	1.35
...					

규칙 필터링 및 정렬

- 예: `lift` 기준으로 상위 규칙 정렬.

R

 코드 복사

```
inspect(sort(rules, by = "lift")[1:5])
```

2. 구매 패턴 분석

- **trans**

읽어올 트랜잭션 데이터를 지정한다.

- **supp = 0.001**

지지도가 0.001 이상인 규칙만 검색한다.

- **conf = 0.7**

신뢰도가 0.7 이상인 규칙만 검색한다.

트랜잭션(거래) 수: 9465 건
 $9465 * 0.001 = 9.5$,
구매가 9건 이상 일어난 규칙만 검색

2. 구매 패턴 분석

summary(rules)

총 14개의 연관 규칙이 생성됨
예: {Bread} => {Coffee} 같은 규칙

```
> summary(rules)
set of 14 rules
```

```
rule length distribution (lhs + rhs):size
  2  3  4
  3 10  1
```

규칙 길이 분포 (lhs + rhs)

길이 2인 규칙: 3개 (ex. {A} => {B})

길이 3인 규칙: 10개 (ex. {A, B} => {C})

길이 4인 규칙: 1개 (ex. {A, B} => {C, D})

lhs (Left-Hand Side)와 rhs (Right-Hand Side) 아이템 수의 합 기준

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
2.000	3.000	3.000	2.857	3.000	4.000

Coverage: "조건(lhs)이 등장한 전체 비율"
이 값이 클수록 조건(lhs)의 발생 빈도가 높다는 의미
즉, 자주 등장하는 조건일수록 높은 coverage 가짐

summary of quality measures:

support		confidence	coverage	lift	count
Min.	:0.001057	Min. :0.7044	Min. :0.001268	Min. :1.472	Min. : 10.00
1st Qu.	:0.001294	1st Qu.:0.7404	1st Qu.:0.001506	1st Qu.:1.548	1st Qu.: 12.25
Median	:0.001585	Median :0.7995	Median :0.001955	Median :1.671	Median : 15.00
Mean	:0.003441	Mean :0.7919	Mean :0.004596	Mean :1.655	Mean : 32.57
3rd Qu.	:0.001955	3rd Qu.:0.8333	3rd Qu.:0.002641	3rd Qu.:1.742	3rd Qu.: 18.50
Max.	:0.023666	Max. :0.8750	Max. :0.033597	Max. :1.829	Max. :224.00

mining info:

data	ntransactions	support	confidence
trans	9465	0.001	0.7

call

```
apriori(data = trans, parameter = list(supp = 0.001, conf = 0.7))
```

```
> |
```

- Support (지지도): "전체 거래 중 해당 규칙이 등장한 비율"
- Confidence (신뢰도): "조건이 주어졌을 때 결론이 함께 나타날 확률"
- Lift (향상도): "우연히 발생할 확률에 비해 얼마나 유의미한 관계인지"
 - Lift > 1: 양의 상관관계 (자주 같이 산다!)
 - 평균 Lift: 1.655 → 전반적으로 유의미한 연관
- Count: 규칙에 해당하는 거래 수 (단위: 건수)

2. 구매 패턴 분석

코드 10-6

```
# 신뢰도 상위 10개 규칙 출력
rules.sort <- sort(rules, by='confidence', decreasing = T)
inspect(rules.sort[1:10])

# 산점도 (지지도-향상도)
plot(rules.sort, measure=c("support", "lift"), shading="confidence")

# Graph plot
plot(rules.sort, method="graph")

# Grouped Matrix Plot
plot(rules.sort, method="grouped")

## 연관 규칙의 저장
write(rules.sort, file="BreadBasket_rules.csv", sep=',', quote=T,
row.names=F)
```

옵션 digits=n

```
> options(digits=2)                                # 평가척도 값의 자리수 지정
> inspect(rules[1:5])
```

	lhs	rhs	support	confidence	coverage	lift	count
[1]	{Extra Salami or Feta}	=> {Coffee}	0.0033	0.82	0.0040	1.7	31
[2]	{Keeping It Local}	=> {Coffee}	0.0054	0.81	0.0067	1.7	51
[3]	{Toast}	=> {Coffee}	0.0237	0.70	0.0336	1.5	224
[4]	{Cake, Vegan mincepie}	=> {Coffee}	0.0011	0.83	0.0013	1.7	10
[5]	{Extra Salami or Feta, Salad}	=> {Coffee}	0.0015	0.88	0.0017	1.8	14

컬럼	의미	예시 (3번 규칙)
lhs	조건 항목 (If)	{Toast}
rhs	결과 항목 (Then)	{Coffee}
support	Toast와 Coffee가 동시에 등장한 비율	0.0237 → 전체 거래 중 2.37%
confidence	Toast가 있을 때 Coffee도 같이 나온 비율	0.70 → 70% 확률로 같이 나옴
coverage	Toast만 등장한 거래의 비율 (조건만)	0.0336
lift	Coffee가 우연히 등장하는 확률 대비 증가율	1.5 → 50% 더 자주 나옴
count	해당 규칙이 등장한 실제 건수	224건

옵션 digits=5

```
> options(digits=5)           # 평가척도 값의 자리수 지정
> inspect(rules[1:10])
```

	lhs	rhs	support	confidence	coverage	lift	count
[1]	{Extra Salami or Feta}	=> {Coffee}	0.0032752	0.81579	0.0040148	1.7053	31
[2]	{Keeping It Local}	=> {Coffee}	0.0053883	0.80952	0.0066561	1.6922	51
[3]	{Toast}	=> {Coffee}	0.0236661	0.70440	0.0335975	1.4724	224
[4]	{Cake, Vegan mincepie}	=> {Coffee}	0.0010565	0.83333	0.0012678	1.7419	10
[5]	{Extra Salami or Feta, Salad}	=> {Coffee}	0.0014791	0.87500	0.0016904	1.8290	14
[6]	{Hearty & Seasonal, Sandwich}	=> {Coffee}	0.0012678	0.85714	0.0014791	1.7917	12
[7]	{Salad, Sandwich}	=> {Coffee}	0.0015848	0.83333	0.0019017	1.7419	15
[8]	{Cake, Salad}	=> {Coffee}	0.0010565	0.76923	0.0013735	1.6079	10
[9]	{Juice, Spanish Brunch}	=> {Coffee}	0.0020074	0.73077	0.0027470	1.5275	19
[10]	{Pastry, Toast}	=> {Coffee}	0.0013735	0.86667	0.0015848	1.8116	13

기본적으로 **Apriori** 알고리즘이 발견한 순서대로 나열됨.

연관분석 결과 해석

```
> options(digits=2) # 평가척도 값의 자리수 지정
> inspect(rules[1:5])
```

	lhs	rhs	support	confidence	coverage	lift	count
[1]	{Extra Salami or Feta}	=> {Coffee}	0.0033	0.82	0.0040	1.7	31
[2]	{Keeping It Local}	=> {Coffee}	0.0054	0.81	0.0067	1.7	51
[3]	{Toast}	=> {Coffee}	0.0237	0.70	0.0336	1.5	224
[4]	{Cake, Vegan mincepie}	=> {Coffee}	0.0011	0.83	0.0013	1.7	10
[5]	{Extra Salami or Feta, Salad}	=> {Coffee}	0.0015	0.88	0.0017	1.8	14

[3] {Toast} => {Coffee}

해석: "Toast를 구매한 고객 중 70%는 Coffee도 함께 구매했다."

lift = 1.5: Coffee가 Toast 없이 등장할 확률보다 1.5배 더 많이 등장함

마케팅 인사이트:

→ 토스트 사는 고객에게 커피 프로모션 쿠폰 주면 먹인다!

[5] {Extra Salami or Feta, Salad} => {Coffee}

조건이 더 구체적: 이 조합으로 구매한 고객 88%가 커피도 샀어!

lift = 1.8 → 무려 80% 더 높은 커피 동반 구매율

정밀 타겟 마케팅 대상!

지표	설명	예시 ({Toast} => {Coffee})
support	조건과 결과가 동시에 등장한 비율	0.0237 (2.37%)
coverage	조건(lhs)만 등장한 비율	0.0336 (3.36%)
confidence	조건 등장 시 결과도 등장한 비율 = support / coverage	0.70 (2.37 / 3.36)

2. 구매 패턴 분석

```
> # 앞쪽 10개의 규칙 출력
```

```
> options(digits=2)
```

```
# 평가척도 값의 자리수 지정
```

```
> inspect(rules[1:10])
```

	lhs	rhs	support	confidence	coverage	lift	count
[1]	{Extra Salami or Feta}	⇒ {Coffee}	0.0033	0.82	0.0040	1.7	31
[2]	{Keeping It Local}	⇒ {Coffee}	0.0054	0.81	0.0067	1.7	51
[3]	{Toast}	⇒ {Coffee}	0.0237	0.70	0.0336	1.5	224
[4]	{Cake, Vegan mincepie}	⇒ {Coffee}	0.0011	0.83	0.0013	1.7	10
[5]	{Extra Salami or Feta, Salad}	⇒ {Coffee}	0.0015	0.88	0.0017	1.8	14
[6]	{Hearty & Seasonal, Sandwich}	⇒ {Coffee}	0.0013	0.86	0.0015	1.8	12
[7]	{Salad, Sandwich}	⇒ {Coffee}	0.0016	0.83	0.0019	1.7	15
[8]	{Cake, Salad}	⇒ {Coffee}	0.0011	0.77	0.0014	1.6	10
[9]	{Juice, Spanish Brunch}	⇒ {Coffee}	0.0020	0.73	0.0027	1.5	19
[10]	{Pastry, Toast}	⇒ {Coffee}	0.0014	0.87	0.0016	1.8	13

2. 구매 패턴 분석

```
> # 신뢰도 상위 10개 규칙 출력
```

```
> rules.sort <- sort(rules, by='confidence', decreasing = T)
```

```
> inspect(rules.sort[1:10])
```

	lhs	rhs	support	confidence	coverage	lift	count
[1]	{Extra Salami or Feta, Salad}	⇒ {Coffee}	0.0015	0.88	0.0017	1.8	14
[2]	{Pastry, Toast}	⇒ {Coffee}	0.0014	0.87	0.0016	1.8	13
[3]	{Hearty & Seasonal, Sandwich}	⇒ {Coffee}	0.0013	0.86	0.0015	1.8	12
[4]	{Cake, Vegan mincepie}	⇒ {Coffee}	0.0011	0.83	0.0013	1.7	10
[5]	{Salad, Sandwich}	⇒ {Coffee}	0.0016	0.83	0.0019	1.7	15
[6]	{Extra Salami or Feta}	⇒ {Coffee}	0.0033	0.82	0.0040	1.7	31
[7]	{Keeping It Local}	⇒ {Coffee}	0.0054	0.81	0.0067	1.7	51
[8]	{Cookies, Scone}	⇒ {Coffee}	0.0016	0.79	0.0020	1.7	15
[9]	{Juice, Pastry}	⇒ {Coffee}	0.0018	0.77	0.0023	1.6	17
[10]	{Cake, Salad}	⇒ {Coffee}	0.0011	0.77	0.0014	1.6	10

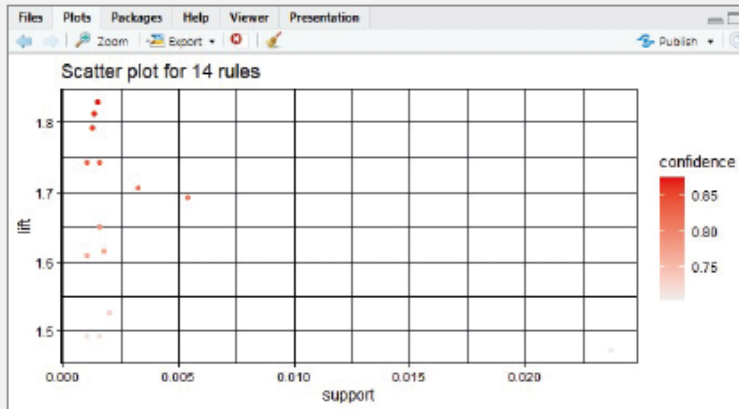
정렬

inspect(sort(rules, by = "support")[1:10])	# 지지도 순
inspect(sort(rules, by = "confidence")[1:10])	# 신뢰도 순
inspect(sort(rules, by = "lift")[1:10])	# 향상도 순
inspect(sort(rules, by = "coverage")[1:10])	# 조건출현률 순

2. 구매 패턴 분석

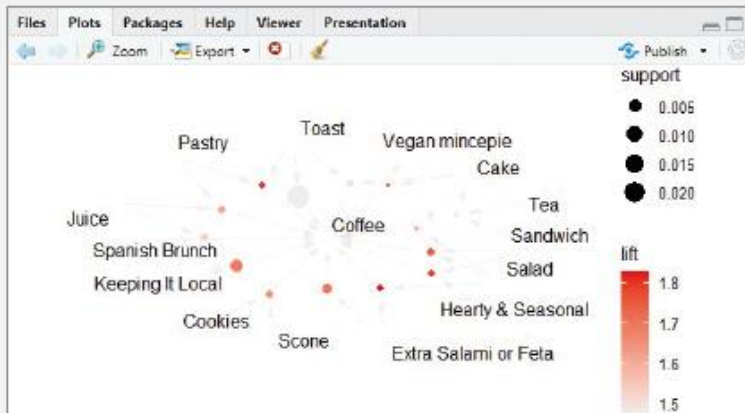
> # 산점도 (지지도-향상도)

> plot(rules.sort, measure=c("support", "lift"), shading="confidence")



> # Graph plot

> plot(rules.sort, method="graph")



(a) {Toast} → {Coffee}



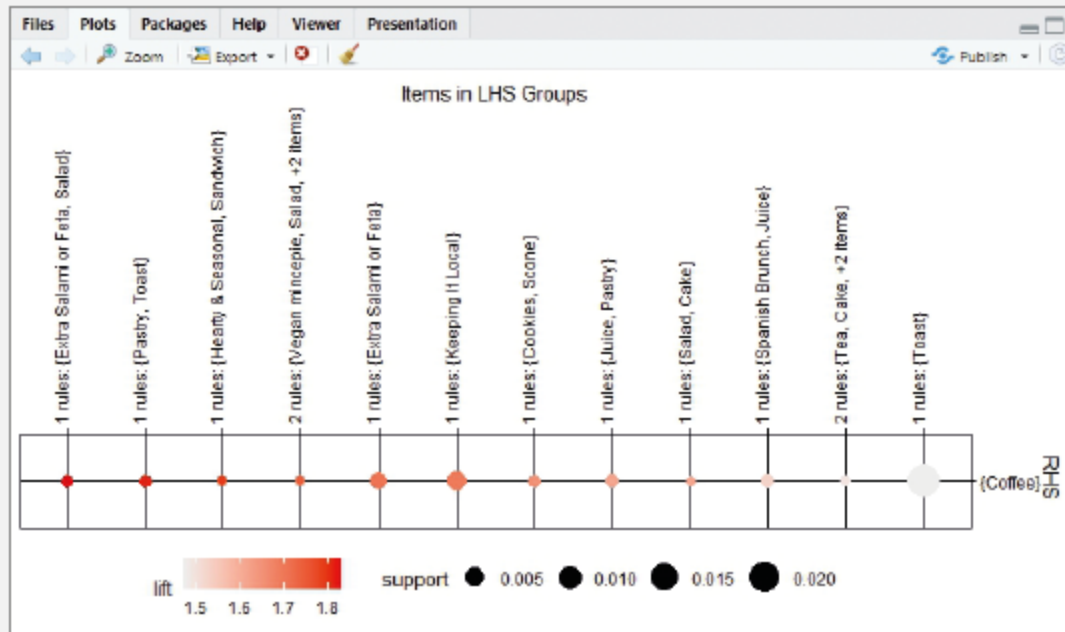
(b) {Pastry, Toast} → {Coffee}

그림 10-4 연관규칙의 그래프 표현

2. 구매 패턴 분석

```
> # Grouped Matrix Plot
```

```
> plot(rules.sort, method="grouped")
```



2. 구매 패턴 분석

> ## 연관규칙 저장

> write(rules.sort, file="BreadBasket_rules.csv", sep=',', quote=T, row.names=F)

	A	B	C	D	E	F
1	rules	support	confidence	coverage	lift	count
2	{Extra Salami or Feta,Salad} => {Coffee}	0.001479	0.875	0.00169	1.829036	14
3	{Pastry,Toast} => {Coffee}	0.001373	0.866667	0.001585	1.811617	13
4	{Hearty & Seasonal,Sandwich} => {Coffee}	0.001268	0.857143	0.001479	1.791709	12
5	{Cake,Vegan mincepie} => {Coffee}	0.001057	0.833333	0.001268	1.741939	10
6	{Salad,Sandwich} => {Coffee}	0.001585	0.833333	0.001902	1.741939	15
7	{Extra Salami or Feta} => {Coffee}	0.003275	0.815789	0.004015	1.705267	31
8	{Keeping It Local} => {Coffee}	0.005388	0.809524	0.006656	1.692169	51
9	{Cookies,Scone} => {Coffee}	0.001585	0.789474	0.002007	1.650258	15
10	{Juice,Pastry} => {Coffee}	0.001796	0.772727	0.002324	1.615253	17
11	{Cake,Salad} => {Coffee}	0.001057	0.769231	0.001373	1.607944	10
12	{Juice,Spanish Brunch} => {Coffee}	0.002007	0.730769	0.002747	1.527547	19
13	{Cake,Toast} => {Coffee}	0.001585	0.714286	0.002219	1.493091	15
14	{Cake,Sandwich,Tea} => {Coffee}	0.001057	0.714286	0.001479	1.493091	10
15	{Toast} => {Coffee}	0.023666	0.704403	0.033597	1.472431	224

그림 10-5 BreadBasket_rules.csv 파일

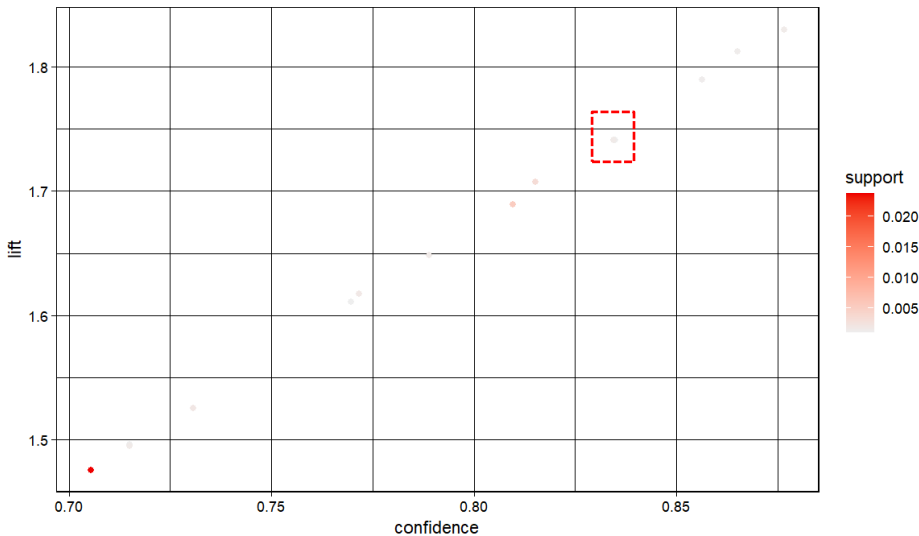
오픈 인자 jitter=T

```
plot(rules, measure=c("confidence", "lift"), jitter=T, shading="support")
```

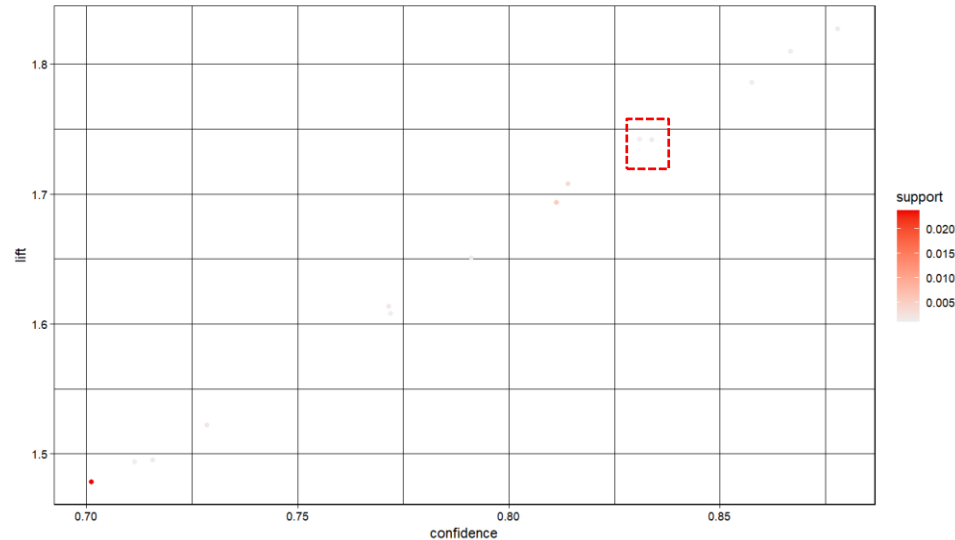
중복되는 점들을 약간 흩뿌려서 보기 쉽게 함

특히 동일한 confidence와 lift 값을 가지는 규칙이 많을 경우 유용

Scatter plot for 14 rules



Scatter plot for 14 rules



```
> inspect(rules.sort[1:10])
```

	lhs	rhs	support	confidence	coverage	lift	count
[1]	{Extra Salami or Feta, Salad}	=> {Coffee}	0.0014791	0.87500	0.0016904	1.8290	14
[2]	{Pastry, Toast}	=> {Coffee}	0.0013735	0.86667	0.0015848	1.8116	13
[3]	{Hearty & Seasonal, Sandwich}	=> {Coffee}	0.0012678	0.85714	0.0014791	1.7917	12
[4]	{Cake, Vegan mincepie}	=> {Coffee}	0.0010565	0.83333	0.0012678	1.7419	10
[5]	{Salad, Sandwich}	=> {Coffee}	0.0015848	0.83333	0.0019017	1.7419	15
[6]	{Extra Salami or Feta}	=> {Coffee}	0.0032752	0.81579	0.0040148	1.7053	31

Thank you!