International Journal of Business Management and Economic Review

Vol. 4, No. 02; 2021

ISSN: 2581-4664

USING APRIORI ALGORITHM TO SEARCH FOR ASSOCIATIVE RULES

Al-Mahmood Ahmed Wasfi Dhahir and Luayabdulwahidshihab

[1]Master student of program Applied Computer Data Analysis,Yanka Kupala State University of Grodno, Belarus, Grodno

[2]college of nursing –University of Basrah -Iraq

http://doi.org/10.35409/IJBMER.2021.3237

ABSTRACT

In this article is investigated Apriori algorithm and are showed some simple examples of it using on R-language to see how it's helpful.

Keyword: Apriori Algorithm , Search , Associative Rules.

1. INTRODUCTION

Is an algorithm for frequent item set mining and association rule learning over transactional databases?

It proceeds by identifying the frequent individual items in the database and extending them to larger and larger item sets as long as those item sets appear sufficiently often in the database.The frequent item sets determined by Apriori can be used to determine association rules which highlight general trends in the database: this has applications in
domains such as market basket analysis [1].

An algorithm that attempts to operate on database records, particularly transactional records, or records including certain numbers of fields or items. It is one of a number of algorithms using a "bottom-up approach" to incrementally contrast complex records, and it is useful in today's complex machine learning and artificial intelligence projects [3].

The Apriori algorithm was proposed by Agrawal and Srikant in 1994 [1] and It is very important for effective Market Basket Analysis and it helps the customers in purchasing their items with more ease which increases the sales of the markets, it has also been used in the field of healthcare for the detection of adverse drug reactions and it produces association rules that indicates what all combinations of medications and patient characteristics lead to ADRs [4].

Association rules:

Association rules are if-then statements that help to show the probability of relationships between data items within large data sets in various types of databases. Association rule mining has a number of applications and is widely used to help discover sales correlations in transactional data or in medical data sets [5].To contribute to increasing the reliability of the data and information generated in the Big Dataenvironments, this study proposes to carry out an analysis focusing on the characteristic veracity inBig Data, presenting the relations of this aspect

Vol. 4, No. 02; 2021
ISSN: 2581-4664

with the technological mechanisms of information [6].
security..

What this algorithm designed for?

Apriori is designed to operate on databases containing transactions (for example, collections of items bought by customers, or details of a website frequentation or IP addresses) [1].

Apriori Property:

All nonempty subset of frequent item set must be frequent the key concept of Apriori algorithm is its anti-monotonicity of support measure Apriori assumes that [2].

Example 1: [2]

TID	Items
T1	I1,I2,I5
T2	I2,I4
T3	I2,I3
T4	I1,I2,I4
T5	I2,I3
T6	I2,I3
T7	I1,I3
T8	I1,I2,I3,I5
T9	I1,I2,I3

Minimum support count is 2, Minimum confidence is 60%

Step1: K=1

Create a table containing support count of each item present in dataset – Called **C1 (candidate set)**

Item-Set	Sup-Count
I1	6
I2	7
I3	6
I4	2
I5	2

Compare candidate set item's support count with minimum support count(here min_support=2 if support count of candidate set items is less than min_support then remove those items) this gives us itemset L1.

Vol. 4, No. 02; 2021
ISSN: 2581-4664

Itemset	Sup-Count
I1	6
I2	7
I3	6
I4	2
I5	2

Step-2: K=2
Generate candidate set C2 using L1 (this is called join step). Condition of joining is L_{k-1} and L_{k-1} is that it should have (K-2) elements in common.
Check all subsets of a itemset are frequent or not and if not frequent remove that itemset.(Example subset of{I1, I2} are {I1}, {I2} they are frequent. Check for each item set).
Now find support count of these itemsets by searching in dataset.

Item-Set	Sup-Count
I1,I2	4
I1,I3	4
I1,I4	1
I1,I5	2
I2,I3	4
I2,I4	2
I2,I5	2
I3,I4	0
I3,I5	1
I4,I5	0

Compare candidate (C2) support count with minimum support count(here min_support=2 if support_count of candidate set item is less than min_support then remove those items) this gives us item set L2.

Item-Set	Sup-Count
I1,I2	4
I1,I3	4
I1,I5	2
I2,I3	4
I2,I4	2
I2,I5	2
I2,I5	2

Vol. 4, No. 02; 2021
ISSN: 2581-4664

Step-3:
Generate candidate set C3 using L2 (join step). Condition of joining L_{k-1} and L_{k-1} is it should have (K-2) elements in common. So here for L2 first element should match. So itemset generated by joining L2 is {I1, I2, I3}{I1, I2, I5}{I1, I3, i5}{I2, I3, I4}{I2, I4, I5}{I2, I3, I5}
Check all subsets of these itemsets are frequent or not and if not remove that item set.(Here subset of {I1, I2, I3} are {I1, I2}{I2, I3}{I1, I3} which are frequent. For {I2, I3, I4} subset {I3, I4} is not frequent so remove this. Similarly check for every item setfind support count of these remaining itemset by searching in dataset.

Item-Set	**Sup-Count**
I1,I2,I3	2
I1,I2,I5	2

Compare candidate (C3) support count with minimum support count(here min_support=2 if support_count of candidate set item is less than min_support then remove those items) this gives us item set L3.

Item-Set	**Sup-Count**
I1,I2,I3	2
I1,I2,I5	2

Step-4:
Generate candidate set C4 using L3 (join step). Condition of joining L_{k-1} and L_{k-1}(K=4)is these should have (K-2) elements in common. So here for L3 first 2 element(items) should match.
Check all subsets of these itemsets are frequent or not (Here itemset formed by joining L3 is {I1, I2, I3, I5} so its subset contain {I1, I3, I5} which is not frequent). so no itemset in C4
We stop here because no frequent itemset are found frequent further

Example 2 about Apriori using R language:
So I used Groceries database as an example because it has represents a set of transactions.
By using the commend:

library (arules)
data (Groceries)

And then I used the command to get the result:

rules <- apriori(Groceries, parameter = list(supp = 0.0015, conf = 0.8))

Vol. 4, No. 02; 2021
ISSN: 2581-4664

```
rules<-sort(rules, decreasing=TRUE,by="confidence")
inspect(rules)
```

And then i used the command to see information about yogurt and compare it to the other products in the basket it's like comparison:-

```
rules<-apriori(data=Groceries, parameter=list(supp=0.015,conf =  0.2,minlen=2),
appearance = list(default="rhs",lhs="yogurt"),
control = list(verbose=F))
rules<-sort(rules, decreasing=TRUE,by="confidence")
inspect(rules)
```

Then I used the command to see that the milk is close to the yogurt and the brown bread is the far one:-

```
rules<-apriori(data=Groceries, parameter=list(supp=0.001,conf = 0.5,minlen=2),
appearance = list(default="rhs",lhs="butter"),
control = list(verbose=F))
rules<-sort(rules, decreasing=TRUE,by="confidence")
inspect(rules)
```

And then I used the Plot command to get the visual result:

```
data(Groceries)
rules <- apriori(Groceries, parameter=list(support=0.001, confidence=0.8))
rules
```

*Scatterplot

```
plot(rules)
```

*Scatterplot with custom colors

```
library(colorspace) # for sequential_hcl
plot(rules, control = list(col=sequential_hcl(100)))
plot(rules, col=sequential_hcl(100))
plot(rules, col=grey.colors(50, alpha =.8))
```

*See all control options using verbose

```
plot(rules, verbose = TRUE)
```

*Interactive plot (selected rules are returned)

```
sel<- plot(rules, engine = "interactive")
```

*Create a html widget for interactive visualization

```
plot(rules, engine = "htmlwidget")
```

*Two-key plot (is a scatterplot with shading = "order")

```
plot(rules, method = "two-key plot")
```

Vol. 4, No. 02; 2021
ISSN: 2581-4664

*Matrix shading
*The following techniques work better with fewer rules
subrules<- subset(rules, lift>5)
subrules

*2D matrix with shading
plot(subrules, method="matrix")

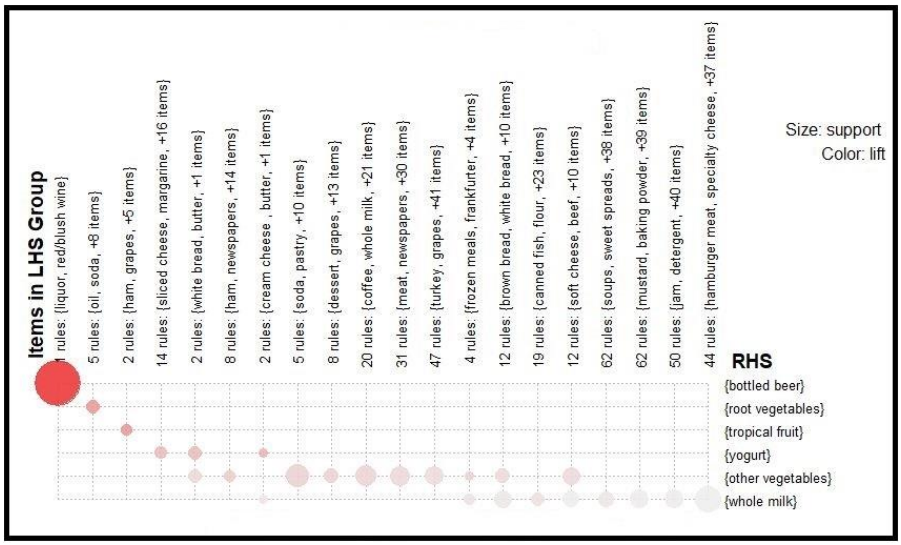
*3D matrix
plot(subrules, method="matrix", engine = "3d")

*Matrix with two measures
plot(subrules, method="matrix", shading=c("lift", "confidence"))

*Interactive matrix plot (default interactive and as a html widget)
plot(subrules, method="matrix", engine="interactive")
plot(subrules, method="matrix", engine="htmlwidget")

*Grouped matrix plot
plot(rules, method="grouped matrix")
plot(rules, method="grouped matrix",
col = grey.colors(10),
gp_labels = gpar(col = "blue", cex=1, fontface="italic"))

Groped Matrix for 410 Rules



Vol. 4, No. 02; 2021
ISSN: 2581-4664

And as we see the big red circle represents the few items that have less connection with the other products

*Interactive grouped matrix plot
sel<- plot(rules, method="grouped", engine = "interactive")

*Graphs
*Graphs only work well with very few rules
subrules2 <- sample(subrules, 25)
plot(subrules2, method="graph")

*Custom colors
plot(subrules2, method="graph",
nodeCol = grey.colors(10), edgeCol = grey(.7), alpha = 1)

*Igraph layout generators can be used (see ? igraph::layout_)
plot(subrules2, method="graph", layout=igraph::in_circle())
plot(subrules2, method="graph",
layout=igraph::with_graphopt(spring.const=5, mass=50))

*Graph rendering using Graphviz
plot(subrules2, method="graph", engine="graphviz")

*Default interactive plot (using igraph'stkplot)
plot(subrules2, method="graph", engine = "interactive")

*Interactive graph as a html widget (using igraph layout)
plot(subrules2, method="graph", engine="htmlwidget")
plot(subrules2, method="graph", engine="htmlwidget",
igraphLayout = "layout_in_circle")

*Parallel coordinates plot
plot(subrules2, method="paracoord")
plot(subrules2, method="paracoord", reorder=TRUE)

*Doubledecker plot
*Note: only works for a single rule
oneRule<- sample(rules, 1)
inspect(oneRule)
plot(oneRule, method="doubledecker", data = Groceries)

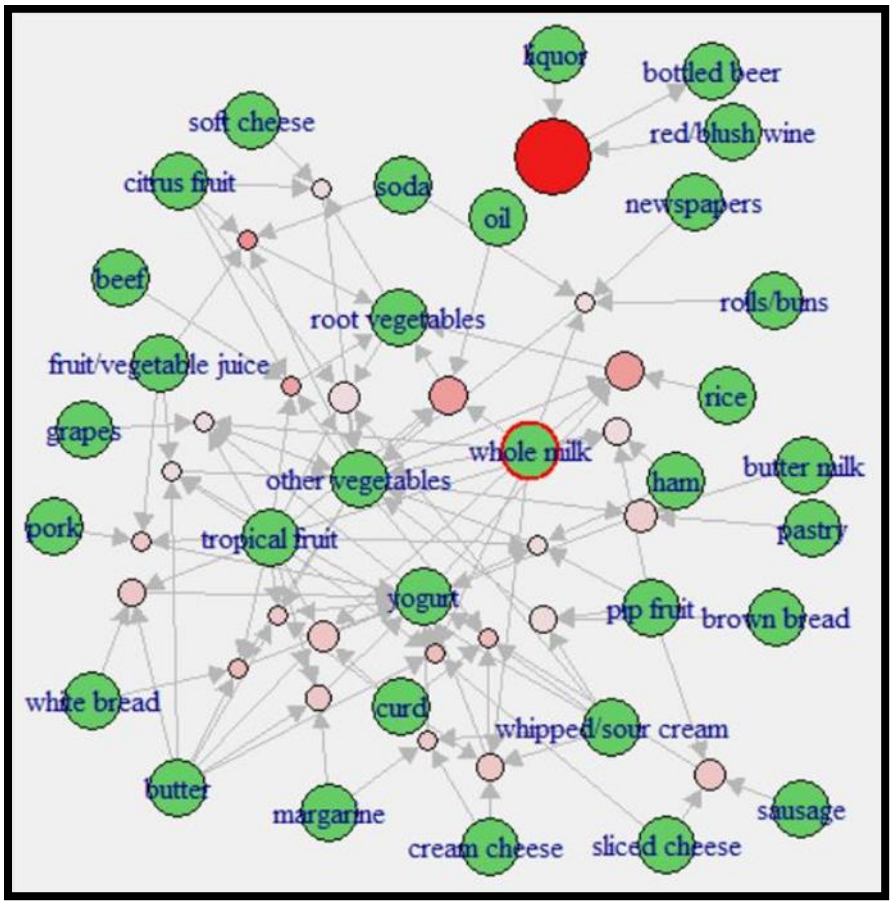
*Itemsets
itemsets<- eclat(Groceries, parameter = list(support = 0.02, minlen=2))
plot(itemsets)

Vol. 4, No. 02; 2021
ISSN: 2581-4664

plot(itemsets, method="graph")
plot(itemsets, method="paracoord", alpha=.5, reorder=TRUE)

*Add more quality measures to use for the scatterplot
quality(itemsets) <- interestMeasure(itemsets, trans=Groceries)
head(quality(itemsets))
plot(itemsets, measure=c("support", "allConfidence"), shading="lift")

*Save HTML widget as web page
p <- plot(rules, engine = "html")
htmlwidgets::saveWidget(p, "arules.html", selfcontained = FALSE)
browseURL("arules.html")



As can be seen from the figure, the most likely pair for yogurt is milk and the most unlikely is black bread.

2. CONCLUSIONS

1st - that the milk is the most requested product in the market.

Vol. 4, No. 02; 2021
ISSN: 2581-4664

2nd- that the domestic egg and the brown breed are not requested that much in the market.
3rd- We see if we put the yogurt closer to the milk that's mean most of the buyers how bought the milk bought the yogurt with it.
4th - And every buyer who bought the brown bread or the domestic egg didn't bought the yogurt.

From the four information above, we see that more products close to milk have a greater chance to be bought with milk and the far products from the milk have less chance in the market.

REFERENCES

https://en.wikipedia.org/wiki/Apriori_algorithm
https://www.geeksforgeeks.org/apriori-algorithm/
https://www.techopedia.com/definition/33156/apriori-algorithm
https://www.hackerearth.com/blog/machine-learning/beginners-tutorial-apriori-algorithm-data-mining-r-implementation/
https://searchbusinessanalytics.techtarget.com/definition/association-rules-in-data-mining.
LuayAbdulwahidShihab, TECHNOLOGICAL TOOLS FOR DATA SECURITY IN THE TREATMENT OF DATARELIABILITY IN BIG DATA ENVIRONMENTS International Transaction Journal of Engineering Management, & Applied Sciences & Technologies ,28 March 2020.