Data Mining Term Project: Market Basket Analysis with Apriori Algorithm

Kerim Safa Embel

Ezgi Berfin Şahin

***Abstract*—**In this document, the definition of the Apriori Algorithm, application areas, and how to apply this algorithm on market basket analysis are explained.

Keywords—Data Mining, Basket Data Analysis, Apriori Algorithm, Association Rule Mining

I. INTRODUCTION

This report is based on the studies and works of the Apriori algorithm for market basket analysis. Market basket analysis or affinity analysis, is a data analysis and data mining technique that discovers co-occurrence relationships among activities performed by specific individuals or groups.

A. Apriori Algorithm

Apriori is an algorithm for frequent itemset mining and association rule learning over relational databases. It is given by R. Agrawal and R. Srikant in 1994. The name of the algorithm is Apriori because it uses prior knowledge of frequent itemset properties.

Apriori algorithm uses frequent item-sets to generating boolean association rules. It is based on the concept that a subset of a frequent itemset must also be a frequent itemset. The frequent itemset is an itemset whose support value is greater than a threshold value(support).

B. Association Rule Mining

Association Rule Mining is used when you want to find an association between different objects in a set, find frequent patterns in a transaction database, relational databases, or any other information repository. The applications of Association Rule Mining are found in Marketing, Market Basket Analysis in retailing, clustering, and classification.

The most common approach to find these patterns is Market Basket Analysis to analyze customer buying habits by finding associations between the different items that customers place in their shopping baskets.

$$\text{Support (A)} = \frac{\text{Number of transaction in which A appears}}{\text{Total number of transactions}}$$

$$\text{Confidence (A}\longrightarrow\text{B)} = \frac{\text{Support(AUB)}}{\text{Support(A)}}$$

Bread=> butter [support=2%, confidence-60%]

The above statement is an example of an association rule. This means that there is a 2% transaction that bought bread and butter together and there are 60% of customers who bought bread as well as butter.

Association rule mining consists of 2 steps:

1. Find all frequent item-sets.
2. Generate association rules from the frequent item-sets.

II. RELATED WORKS

Market basket analysis or affinity analysis examines the products customers tend to buy together and uses the information to decide which products should be cross-sold or promoted together.

Market Basket Analysis takes data at the transaction level, which lists all items bought by a

customer in a single purchase. The technique determines relationships of what products were purchased with which other product(s). These relationships are then used to build profiles containing If-Then rules of the items purchased.

A. *Market Basket Analysis In a Multiple Store Environment*

In this article, market basket analysis was performed for more than one market environment. The purpose of doing this is to reveal products that customers prefer to buy together, regardless of stores. In this way, they discover the purchasing model of the customers and make this model usable as a marketing, sales, service, and operation strategy.

B. *Apriori Algorithm Within Different Work Fields*

In this article, the Apriori Algorithm is adapted to the medical data which has abundant scope for improvement of the quality of service in the healthcare industry. Researchers work with electronic health records and other historical medical data available in textual and graphical formats. Determining the frequently occurring diseases in the local databases by data mining techniques is possible. Apriori algorithm is one key solution to discovering locally frequent diseases both efficiently and coherently.

C. *Market Basket Analysis Using Apriori and FP Growth Algorithm*

Apriori and FP Growth are the most common algorithms for mining frequent item-sets. For both algorithms predefined minimum support is needed to satisfy for identifying the frequent item-sets. But when the minimum support is low, a huge number of candidate sets will be generated which requires large computation.

In this paper, an approach has been proposed to avoid this large computation by reducing the items of the dataset with top-selling products. The results show that if top-selling items are used, it is possible to get almost the same frequent item-sets and association rules within a short time comparing with that outputs which are derived by computing all the items. From time comparison it is also found that the FP Growth algorithm takes a smaller time than the Apriori algorithm.

III. Steps of Apriori Algorithm

Apriori algorithm is a sequence of steps to be followed to find the most frequent itemset in the given database. This data mining technique follows the join and the prune steps iteratively until the most frequent itemset is achieved. A minimum support threshold is given in the problem or it is assumed by the user.

Consider the following dataset and we will find frequent item-sets and generate association rules for them.

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

minimum support count is 2

minimum confidence is 50%

Step-1: K=1

(I) Create a table containing the support count of each item present in the dataset, called the candidate set(C1).

(II) Compare candidate set item's support count with minimum support count. If candidate set items are less than minimum support, then remove those items. This gives us itemset L1.

Itemset	sup_count
I1	6
I2	7
I3	6
I4	2
I5	2

Step-2: K=2

(I) Generate candidate set C2 using L1. The condition of joining Lk-1 and Lk-1 is that it should have (K-2) elements in common.

(II) Check all subsets of an itemset are frequent or not and if not frequent remove that itemset.

Itemset	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

(III) Now find support count of these item-sets by searching in the dataset.

(IV) Compare candidate set C2 item's support count with minimum support count. If candidate set items are less than minimum support, then remove those items. This gives us itemset L2.

Itemset	sup_count
I1,I2	4
I1,I3	4
I1,I5	2
I2,I3	4
I2,I4	2
I2,I5	2
I2,I5	2

Step-3:

(I) Generate candidate set C3 using L2. 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}

(II) Check if all subsets of these item-sets are frequent or not and if not, then remove that itemset. 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 it.

(III) Find support count of these remaining itemset by searching in the dataset.

Itemset	sup_count
I1,I2,I3	2
I1,I2,I5	2

Step-4:

(I) Generate candidate set C4 using L3. So here, for L3, the first 2 elements should match.

(II) Check all subsets of these item-sets are frequent or not. Here itemset formed by joining L3 is {I1, I2, I3, I5} so its subset contains {I1, I3, I5}, which is not frequent. So, no itemset in C4.

(III) Stop here because no frequent item-sets are found further.

Thus, all the frequent item-sets were discovered. Now the generation of strong association rules is needed. For that the confidence of each rule must be calculated.

Itemset {I1, I2, I3} from L3
[I1^I2] =>[I3] sup(I1^I2^I3)/sup(I1^I2) = 2/4*100=50%
[I1^I3] =>[I2] sup(I1^I2^I3)/sup(I1^I3) = 2/4*100=50%
[I2^I3] =>[I1] sup(I1^I2^I3)/sup(I2^I3) = 2/4*100=50%
[I1] =>[I2^I3] sup(I1^I2^I3)/sup(I1) = 2/6*100=33%
[I2] =>[I1^I3] sup(I1^I2^I3)/sup(I2) = 2/7*100=28%
[I3] =>[I1^I2] sup(I1^I2^I3)/sup(I3) = 2/6*100=33%

So, if minimum confidence is 50%, then the first 3 rules can be considered as strong association rules.

IV. MATERIALS AND METHODS

A. *Datasets*

The proper dataset used in this project is labeled as "groceries.csv". The dataset contains 9835 transactions from the previous recording of the shopping. 169 unique items are listed as available products on the market for customers to shop from.

B. *Method*

Each transaction of the CSV file is stored as a list. A mapping is created between the items in the dataset to unique integers. A reverse mapping was created from the integers to the item in the output file.

C. *Library*

The "apriori" and "association_rules" from the Mlxtend library will be used for the Apriori algorithm in Python.

Packages "pandas" and "numpy" will be imported for reading the data and numerical features.

IV. ENVIRONMENTS

A. *Executive Summary*

Dataset tested with 3 different machine learning platforms that are Azure ML Studio, R Studio, and Phyton to show that some platforms give different results for the same support and confidence values.

One of the aims of this study is the determination of the best tool to use market basket analysis.
This study showed which support and confidence values gave better results for the dataset used.

B. Data preparation

Most data mining platforms have different requirements. To try the dataset on different platforms, it had to be arranged so that it could work on the platforms.

The algorithm first started to experiment in the Python environment.

In order to run the Apriori algorithm with the "mlxtend" library in the Python environment, the dataset had to be converted to an array of transactions. To do this, first, the column holding the row numbers had to be deleted and then the row holding the column names had to be cleared.

After these operations were done, commas that showed that the values in the lines did not exist also had to be removed. After these processes were completed, the data set was converted to a transaction array with the Transaction Encoder method in the "mlxtend" library.

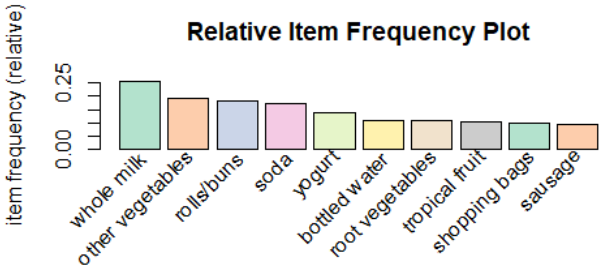
In order to work with the Apriori algorithm in Azure ML Studio, comma values are added to the column places that have no value in the data set, meaning there are no more products. In this platform, header values are left in the dataset because it is necessary data for it to work.

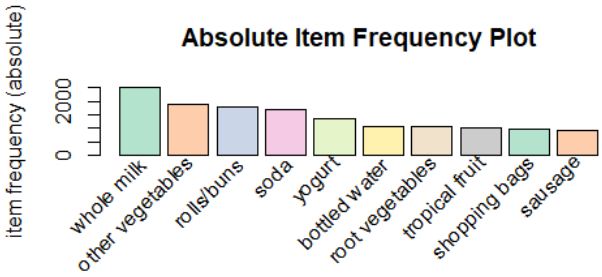
To run the algorithm on the R Studio platform, the dataset had to be as it was run in the Python environment. In other words, the first column holding the row numbers and the row holding the column names, i.e. the header part, has been removed.

After these procedures commas that showed that the values in the lines did not exist also had to be removed.

C. R Script

R Studio is the first choice of machine learning tool that is applied to the project for analyzing market basket purchases. In the script written in the R studio 8 libraries and packages are used in order to perform individual actions upon the dataset. Arules package which contains the read transaction function is used for reading the transaction and CSV files from the directory. Both absolute and relative top 10 most frequently bought items are displayed visually by using the plotting functions.





Apriori rules are generated from the dataset and these rules are saved as text or CSV files in order to save the output data. In addition to those rules support, confidence, lift, and count variables are also displayed in the files. The three attributes support, confidence and lift are crucial in order to generate appropriate rules. Support is the fraction of which our item set occurs in our dataset while confidence is the probability that a rule is correct for a new transaction with items on the left. Lift is the ratio by which the confidence of a rule exceeds the expected confidence. The list of rules would also change accordingly to change of the values of these 3 attributes.

The two different outcomes are generated from the support 0.01 & confidence 0.6 and support 0.001 & confidence 0.8.

```
> association.rules <- apriori(tr, parameter = list(supp=0.01, conf=0.6))
Apriori

Parameter specification:
 confidence minval smax arem  aval originalSupport
        0.6    0.1    1 none FALSE            TRUE
 maxtime support minlen maxlen target   ext
       5    0.01      1     10  rules FALSE

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

Absolute minimum support count: 98
```

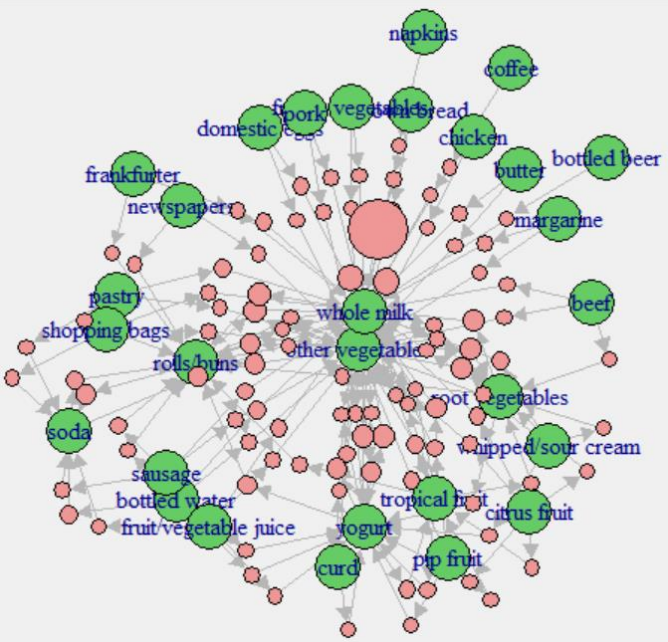
```
> rules <- apriori(tr, parameter = list(supp = 0.001, conf = 0.8))
Apriori

Parameter specification:
 confidence minval smax arem  aval originalSupport
        0.8    0.1    1 none FALSE            TRUE
 maxtime support minlen maxlen target   ext
       5   0.001      1     10  rules FALSE

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

Absolute minimum support count: 9
```

The mapping of the rules is also generated. This map shows that the whole milk and other vegetables are the most common 2 center values. The common rules can be seen by following the links that are coming from one value to another.



D. Python Language

After installing the libraries required to test in the Python environment, the cleaned dataset was read and each line was added to the array as a list. Then, using this array, the data set was converted to a transaction array to be suitable for the mlextend library. Using this array, the Apriori algorithm was run to find frequent itemsets and the 'length' column, which indicates the number of items in this output, was added. Then this itemset array is sorted to decrease according to the support value. This result is printed in a text document to make it easier to read.

After these operations, an algorithm was run to extract the association rules from the final version of the frequent itemset array. Similar procedures are repeated for these association rules and are printed in the file as a text document.

	support	itemsets	length
0	0.074835	(whole milk, other vegetables)	2
1	0.056634	(whole milk, rolls/buns)	2
2	0.056024	(whole milk, yogurt)	2
3	0.048907	(root vegetables, whole milk)	2
4	0.047382	(root vegetables, other vegetables)	2
5	0.043416	(yogurt, other vegetables)	2
6	0.042603	(rolls/buns, other vegetables)	2
7	0.042298	(whole milk, tropical fruit)	2
8	0.040061	(whole milk, soda)	2
9	0.038332	(soda, rolls/buns)	2

In the screenshot above, some of the frequent itemsets can be seen. This image shows the products most commonly taken together. The itemsets column shows the products bought together, and the length column shows the number of products in these item sets. In addition, the support column shows how often these products were purchased in the entire data set.

Considering the row with the highest support value, it can be concluded that in 0.074 of the whole data set, whole milk and other vegetable products are taken together. Similar comments can be made for other product groups in this itemset column.

	antecedents	consequents	antecedent support	consequent support
0	(root vegetables, other vegetables)	(whole milk)	0.047382	0.255516
1	(yogurt, other vegetables)	(whole milk)	0.043416	0.255516
2	(rolls/buns, other vegetables)	(whole milk)	0.042603	0.255516
3	(other vegetables, tropical fruit)	(whole milk)	0.035892	0.255516
4	(yogurt, rolls/buns)	(whole milk)	0.034367	0.255516
5	(soda, other vegetables)	(whole milk)	0.032740	0.255516
6	(yogurt, tropical fruit)	(whole milk)	0.029283	0.255516
7	(other vegetables, citrus fruit)	(whole milk)	0.028876	0.255516
8	(other vegetables, whipped/sour cream)	(whole milk)	0.028876	0.255516
9	(yogurt, soda)	(whole milk)	0.027351	0.255516

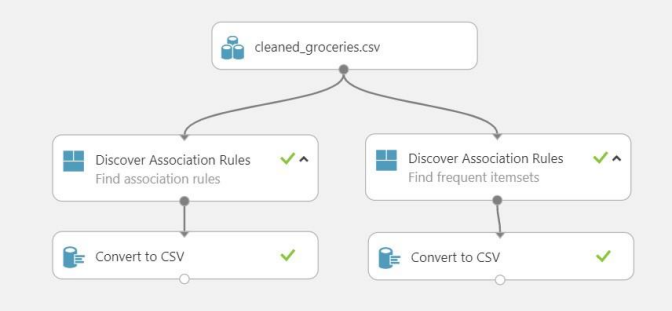
The screenshot above is an image taken from the association rules text. Here, the value of antecedents shows the products taken together, and the value of antecedent support shows the support value of these products taken together. Consequent value, on the other hand, indicates which product the antecedents value products are prone to buy, and the consequent support value indicates their tendency to purchase this product.

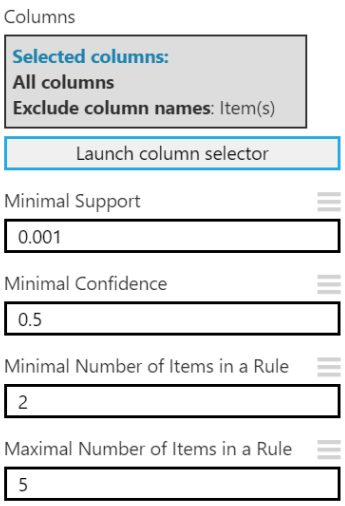
Considering the row with the highest consequent support value, it can be interpreted that 25% of customers who buy root vegetable and other vegetable products also buy the whole milk product.

Similar comments can be made for other rows by looking at the columns with antecedents, consequents and their support values.

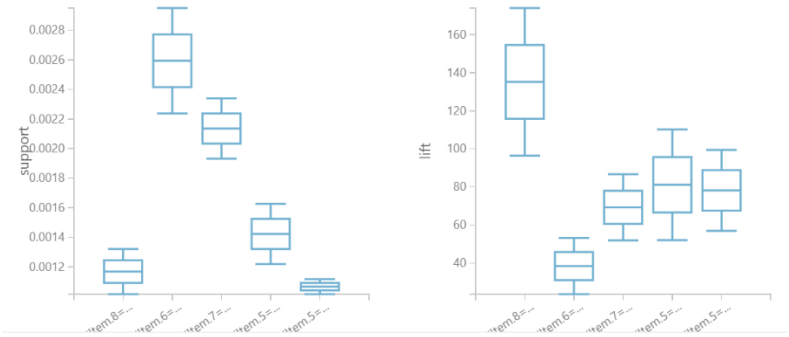
E. Azure Machine Learning Studio

After creating a new experience in the Azure Machine Learning Studio, the dataset which is used for the other machine learning tools also added to the saved datasets section of the studio. The model below is created for getting the expected association rules and the frequent itemsets of the dataset. To get results the cleaned dataset is linked to the discover association rules blocks. Doing that the outputs are generated according to the minimum support, minimum confidence, and several inputs that are changeable. After that the output results are written into CSV files.





The images displaying the frequent itemset rules and the support-lift diagram are based on the values taken as an input that is displayed in the image above. For the datasets' each discovered rule set after specifying the minimal support value to 0.001, minimal confidence value 0.5, max number of rule items to five and the max number of rule items to two, the lift values are in the range of exactly like the graphic included in the below image.



The image below shows the list of rules items that are frequently bought together. The support values are not in order but confidence values are in descending order.

By looking at the confidences of the items that are bought together it is clear to see that each and every time someone bought a sausage, they also bought frankfurter since the first row items have a confidence value of 1.

	id	lhs	rhs	support	confidence	lift
0	1	{Item.2=sausage}	{Item.1=frankfurter}	0.010066	1.000000	16.956897
1	2	{Item.8=butter}	{Item.7=whole milk}	0.001322	0.866667	173.952381
2	3	{Item.4=whole milk,Item.7=whipped/sour cream}	{Item.6=yogurt}	0.001118	0.846154	89.483044
3	4	{Item.2=other vegetables,Item.5=yogurt}	{Item.3=whole milk}	0.001118	0.846154	16.446488
4	5	{Item.7=whole milk}	{Item.6=other vegetables}	0.004067	0.816327	118.067227
5	6	{Item.9=whipped/sour cream}	{Item.8=yogurt}	0.001729	0.809524	194.186992
6	7	{Item.6=butter}	{Item.5=whole milk}	0.002949	0.805556	53.172073
7	8	{Item.6=whole milk}	{Item.5=other vegetables}	0.007016	0.793103	65.001437
8	9	{Item.1=root vegetables,Item.3=whole milk}	{Item.2=other vegetables}	0.003254	0.780488	13.956541
9	10	{Item.2=pip fruit,Item.5=whole milk}	{Item.3=root vegetables}	0.001423	0.777778	36.776175
10	11	{Item.7=butter}	{Item.6=whole milk}	0.002339	0.766667	86.668582
11	12	{Item.5=onions}	{Item.6=other vegetables}	0.001627	0.761905	110.196078
12	13	{Item.1=other vegetables,Item.3=yogurt}	{Item.2=whole milk}	0.002440	0.727273	10.936892
13	14	{Item.2=liquor}	{Item.1=bottled beer}	0.002237	0.709677	39.211671
14	15	{Item.5=whole milk}	{Item.4=other vegetables}	0.010574	0.697987	27.026370
15	16	{Item.2=pip fruit,Item.4=other vegetables}	{Item.3=root vegetables}	0.001830	0.692308	32.734837

The image below shows the top 7 most commonly bought pairs from the dataset. The other vegetables and whole milk happen to bought with a ratio of 0.018302 from the whole dataset while citrus fruit and tropical fruit only form the 0.011591 ratios of the dataset.

	id	items	support
0	1	{Item.2=other vegetables,Item.3=whole milk}	0.018302
1	2	{Item.3=other vegetables,Item.4=whole milk}	0.017285
2	3	{Item.1=other vegetables,Item.2=whole milk}	0.014032
3	4	{Item.2=root vegetables,Item.3=other vegetables}	0.012506
4	5	{Item.1=citrus fruit,Item.2=tropical fruit}	0.011591
5	6	{Item.4=other vegetables,Item.5=whole milk}	0.010574
6	7	{Item.1=frankfurter,Item.2=sausage}	0.010066

IV. Conclusion

Comparing the results obtained by testing the same dataset in 3 different platforms, similar results were obtained that can be said to be the same.

If a cross-platform comparison is made, the R script runs faster and may be the preferred platform as it can be said to be more successful in visualizing the output data.

Looking at the Python language, it can run a little slower than the R script, but when looking at the output data, it can be said that it outputs more meaningful and readable than others.

As for Azure ML Studio, it can be preferred for someone who likes to control from the user interface while working because it works with the drag and drop method. Apart from this feature, its results are considered the same as other platforms, but its performance was not seen as effective as other platforms.

References

[1] Market basket analysis in a multiple store environment, Yen-Liang Chen, Kwei Tang, Ren-Jie Shen, Ya-Han Hu, 2005

[2] Market Basket Analysis, Nagesh Singh Chauhan, 2019

[3] A Gentle Introduction on Market Basket, Susan Li,2017

[4] Introduction to Market Basket Analysis in Python, Chris Moffitt,2017

[5] Finding Locally Frequent Diseases Using Modified Apriori Algorithm, Mohammed Abdul Khaleel, Sateesh Kumar Pradhan, 2013

[6] Market Basket Analysis Using Apriori and FP Growth Algorithm, Maliha Hossain, A H M Sarowar Sattar, Mahit Kumar Paul, 2019