Apriori Alogorithm Analysis Using Grocery Dataset

MGMT 635 SPRING 2017

08 APR 2017

Group Project 2

Team Alpha

Matthew Eustice

Mujahidul Islam

Sharlene Mak

Gabrielle Mack

Abstract

The Apriori algorithm is one of the most prevalent methods of pattern recognition in machine learning. It allows for the generation of association rules from large datasets, providing a means for which stakeholders can analyze valuable criterion in user interaction. To illustrate the usefulness of the algorithm, a large grocery dataset was acquired for analysis. Our team then utilized the Orange Package provided by Python to analyze the data and generate strong and interesting association rules.

Overview

The objective of this project is to utilize a specific algorithm that has proven to be successful in data mining and machine learning in order to draw conclusions from a specific dataset. We have selected a grocery dataset that is commonly utilized in performing market basket analysis. We then applied the algorithm to the dataset to demonstrate its efficiency in pattern recognition. Team Alpha's goals are to utilize the Apriori algorithm to identify support, confidence, and interesting rules from the grocery dataset and apply these useful conclusions to drive real life business decisions.

Other methodologies and algorithms that are used in pattern recognition include neural networks, decisions trees, clustering, and sequential mining patterns. Our team members have had positive experiences using the Apriori algorithm in other projects. It was recommended by those group members to use this algorithm for this data mining project. In addition, there are many advantages in using the Apriori algorithm to discover frequent item sets. The algorithm has a simple rule that dictates if one item in a dataset is frequent, then its remaining subsets must also be frequent. Overall, this is one of the best and most efficient algorithms to identify frequent item sets and assists in drawing useful business conclusions.

Association Rules Overview

Association rule analysis and generation provides a means for which patterns of behavior can be recognized within a dataset. If a dataset comprises of X amount of items, how are these items related? What patterns can we distinguish from these items? Is there any type of correlation between one item and another?

Item sets are comprised of multiple elements. For example, a customer can shop at a store that sells thousands of distinct elements, but the subset of elements they choose to buy are contained in one item set. Patterns of distinction can be discovered within these item sets and analyzed.

These patterns are known as association rules and can dictate with a certain level of support and confidence, how likely the rule is to have an outcome.

Association rules are generally presented in the following format:

Data Mining Concepts -> Tylenol [4%, 75%]

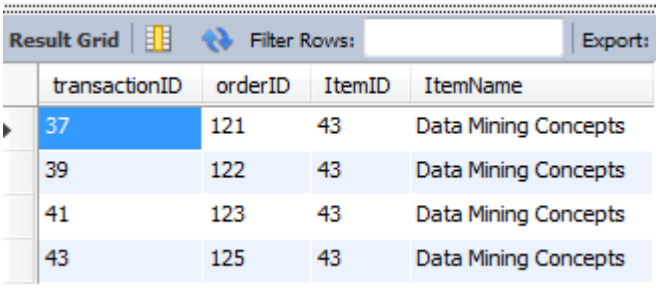
The above rule is a relationship between the book "Data Mining Concepts" and the medication "Tylenol", followed by a 4% support and 75% confidence. Association rules are read as simple if-then statements where the first item (if) is known as the antecedent and the second item (then) is known as a consequent.

Antecendent -> Consequent [Support, Confidence]

Support for the antecedent is calculated by the frequency in which the items appear in the record set.

If we had a database for an online store that sold books, medications, and mobile applications, it would contain orders with unique OrderID's. Each orderID would be considered a 'basket' and its individual items would be referred to as the 'item set'.

For example, in this sample database, 4 distinct orders contain the item 'Data Mining Concepts'.

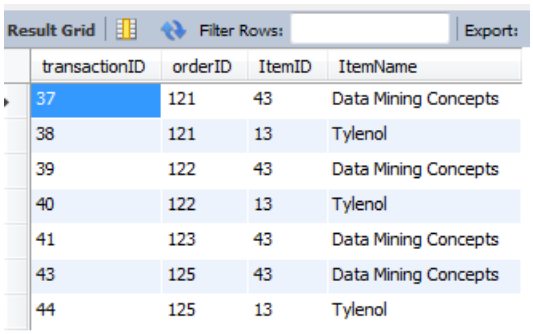

Result Grid | Filter Rows: | Export:

transactionID	orderID	ItemID	ItemName
37	121	43	Data Mining Concepts
39	122	43	Data Mining Concepts
41	123	43	Data Mining Concepts
43	125	43	Data Mining Concepts

Confidence is calculated by multiplying the support of both the antecedent and consequent and dividing it by the support for the antecedent:

$$\boldsymbol{Confidence}\,\{\boldsymbol{X},\boldsymbol{Y}\} = \frac{\textbf{Support}\,\{\mathbf{X},\mathbf{Y}\}}{\textbf{Support}\,\mathbf{X}}$$

Using the above example of the orders with the antecedent "Data Mining Concepts", it is observed below that although there are 4 orders with that item, only 3 have Tylenol associated with them. This would suggest that although this book is not purchased frequently, it is almost always purchased with a bottle of Tylenol. One can draw conclusions from this observation. Perhaps suggesting that this book gives its readers headaches?

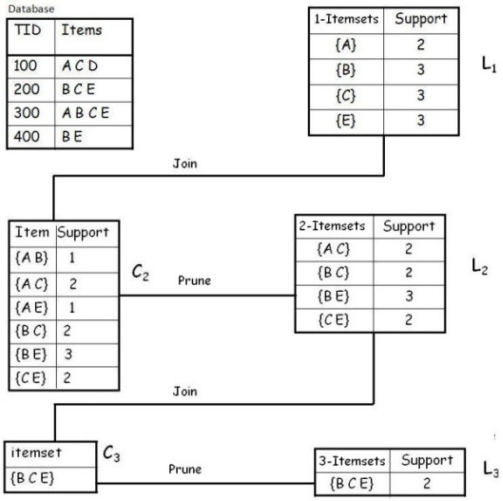

Result Grid | Filter Rows: | Export:

transactionID	orderID	ItemID	ItemName
37	121	43	Data Mining Concepts
38	121	13	Tylenol
39	122	43	Data Mining Concepts
40	122	13	Tylenol
41	123	43	Data Mining Concepts
43	125	43	Data Mining Concepts
44	125	13	Tylenol

Association rules like these can be used at great length to analyze customer purchase patterns. Businesses can use the information gained from these rules to modify sales techniques, product placement, and overall, increase profits.

The Apriori Algorithm

The Apriori algorithm generates association rules provided a support and confidence through a series of Cartesian product calculations. Given a minimum support and confidence, the algorithm prunes the dataset and combines all possible combinations of the item sets thus generating strong association rules.



Rules	Support (X Y)	Support(X)	Confidence
{A} -> {C}	2	2	100
{B} -> {C}	2	3	66.66666667
{B} -> {E}	3	3	100
{C} -> {E}	2	3	66.66666667
{B} -> {C E}	2	3	66.66666667
{C} -> {B E}	2	3	66.66666667
{E} -> {B C}	2	3	66.66666667
{C} -> {A}	2	3	66.66666667
{C} -> {B}	2	3	66.66666667
{E} -> {B}	3	3	100
{E} -> {C}	2	3	66.66666667
{C E} -> {B}	2	2	100
{B E} -> {C}	2	3	66.66666667
{B C} -> {E}	2	2	100

(Reference: codeproject.com)

Using the above example containing 4 transactions (baskets), the algorithm first calculates the support of each of the items within the recordset. It then uses the frequency (support) of each item and compares it against the support provided by the user. The algorithm then proceeds to prune the items that do not satisfy the support threshold. Those items left which satisfy the support threshold are used to create candidate lists through cross joins. At each iteration of the cross join, the same calculation and pruning technique is used eventually yielding association rules which satisfy the given support and confidence thresholds.

Market Basket Dataset

Based on the association rules previously described, we have identified that the grocery data set is perfect for demonstrating the market basket analysis methodology. The grocery data set is comprised of 2,000 customer transactions. Each transaction represents a receipt of the purchased items from the client's trip to the grocery store. You can think of the receipt as a representation of the items the customer has placed into their basket, hence how the term "market basket analysis" is coined. The grocery data set contains transactions ranging from one item to twenty-three items. By applying market basket analysis utilizing Apriori, association rules are generated by the algorithm to identify the patterns of certain items being purchased together. The abundance of data available in the grocery data set allows for our team to test many variations by setting specific support and confidence levels to yield the top rules.

The source our team utilized to download the original grocery database was retrieved from http://www.salemmarafi.com/code/market-basket-analysis-with-r/. Upon deciding to use the Orange Data Mining Package for Python, we took two approaches to utilize the dataset. Our very first step was to create a test dataset and get approval from Professor Yu to see if our approach aligned with the directions

for Project 2. In order to create a smaller sample of transactions to test the functionality of the code, we created a sample of 30 translation examples. We manually generated data by selecting 20 items at random from the original list of items in the grocery data set. We then combined the 20 items into various transactions by using a regular expression to parse the dataset and strip the commas. We utilized the test data set and successfully ran the Jupyter notebook python code to run the association function in Orange. We also had the ability to modify the confidence and support values in the python code to adjust the results. After obtaining feedback from Professor Yu, we were advised to expand the script to accommodate reading data from a file rather than importing a manually created dataset. After attempts from our group members, we discovered the version of Orange that we had installed could not read the Panda dataframe and repeatedly gave our team members' errors while attempting to run the code. Eventually the team was able to correctly format the data into a .tab file which was parsed properly through the Orange function.

Figure 5.1 Example of the manually generated grocery dataset

1	citrus_fruit, 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, bakery_product
6	whole_milk, butter, yogurt, rice, abrasive_cleaner
7	rolls, buns
8	other_vegetables, UHT_milk, rolls, buns, bottled_beer, liquor
9	pot_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, flour, bottled_water, dishes
13	beef
14	frankfurter, rolls, buns, soda
15	chicken, tropical_fruit

Tools

For our Python framework, we used Anaconda available from https://www.continuum.io/downloads. This package includes both Python and Jupyter notebook. It enabled us to directly make use of the Orange package which includes an Apriori function and gives an easy way to implement association rules based on a dataset. The Orange package can be easily configured and used for Python code while using Anaconda Navigator and Jupyter notebook.

We had to face some challenges installing Orange. Initially, after installing Anaconda, the Orange installation would update qtpy to the latest version. This corrupted the Anaconda Navigator installations and prevented the module from launching. Here is the blog related to the issue: https://github.com/ContinuumIO/anaconda-issues/issues/1095

We were able to get Anaconda Navigator working by downgrading qtpy. We installed Anaconda for Python 2.7 version and Orange 2.7.8 and then ran command "`conda install qtpy=1.0`" to downgrade this package.

We can easily invoke an Orange function to retrieve the rules by defining support and confidence for the dataset.

```
rules = Orange.associate.AssociationRulesSparseInducer(data,
support=0.01, confidence=0.2)
```

We could also achieve the association rules implementation by using Matlab. The article below shows how association rules functions and sample code that uses the Apriori algorithm in Matlab.

http://www.mathworks.com/matlabcentral/fileexchange/42541-association-rules

We did not choose Matlab for implementing association rules because our prior work on Neural Networks utilized Matlab. We instead opted to learn something new for Project 2, and decided upon Python. We wanted to understand how this popular approach worked and become familiar with Python and its various Data Mining and Machine Learning packages. We also attempted to use an Excel file with Panda and load it into a numpy array to work with grocery dataset. This was unsuccessful, and instead, we opted for using a .tab file.

Regarding association rules and data mining, there are several tools available commercially as well as free versions. IBM SPSS Modeler is one of the popular commercial tools performs market basket analysis. Azmy SuperQuery includes association rule finder as well. For relational database, we can use LPA Data Mining Toolkit that supports the discovery of association rules. For a flexible option in finding associations in data, MAGNUM Opus includes statistical support for avoiding spurious discoveries.

As far as free tools go, to work with association rules, we can manually program a version of Apriori to find association rules with the algorithm. ARtool is a collection of algorithms and tools for the mining of association rules in binary databases. ARules is also a free R extension package which provides the infrastructure for representing, manipulating and analyzing transaction data association rules. [http://www.kdnuggets.com]

Data Preprocessing

The raw data included in the downloaded dataset was not directly usable as we initially used a regular expression and it expects strings of products. In addition to data type being incorrect, the original data formatted transactions contained items that consisted of multiple words, white space, and special characters. We had to apply general excel formulas and utilize a find and replace operation to strip these characters for use in the Jupyter notebook. Finally, we created a tabbed file as input to the Orange table and used that for applying association rules on it. For tabbed file generation we had to format data into tab delimited products in the fixed file which is used and loaded directly into Orange table.

We also tried to maximize the number of transactions to find interesting rules based on support and confidence defined. Modifying support and confidence resulted on different dataset output.

We tried using panda to import data from excel file but were unsuccessful. Alternatively, we used a regular expression to load the manually inputted dataset into the orange domain. This would loop through the strings and find associations rules. In the program, the regular expression loaded the string of texts separated by commas and parsed the data into the Orange table as output. Eventually, we were able to parse the data to the Orange function using a .tab file.

Apriori Algorithm Using Orange

For this project, Team Alpha used Python and Orange for generating association rules and displayed the output using a Jupyter notebook. The dataset went through an initial pre-processing step, in which the team had to remove any characters or spaces that could potentially break the entry into the algorithm. Further, the dataset was loaded into a .tab file extension for allowing easy passage to the orange algorithm.

In order to create association rules, the transaction database list includes several individual transactions with a common set of items included within each transaction. For Group Project 2, Team Alpha generated a grocery transaction dataset from an existing dataset but altered the formatting to remove unnecessary characters, such as '-' or '/'. The items within the transactions were common items found at retail stores or grocery stores.

Creating a .tab file

With the data pre-processed, as part of the Orange library, the next step of the data preparation process is to upload the data as a.csv file into Orange and save as a .tab file. To do this, Orange provides a desktop client for uploading and saving the file to be the compatible format.

Using Orange

```
import Orange
#To install Orange, conda install -c anaconda orange=2.7.8
data = Orange.data.Table("grocery_list.tab")

rules =
Orange.associate.AssociationRulesSparseInducer(data,
support=0.005, confidence=0.8)
print "%4s %4s  %s" % ("Supp", "Conf", "Rule")
for r in rules[:50000]:
    print "%8.3f %8.3f  %s" % (r.support, r.confidence, r)
```

First, we need to import the Orange algorithm library into our Python code. Doing so will allow us to use the association functions that Orange provides. Simply call the import function and pass "Orange" into the function to tell python that you want to use the Orange library. Next, our .tab file is loaded into an Orange table for passing the data into the function.

Python makes calling the Orange algorithm itself look simple, with just a single line of code, but it's actually comprised of several steps to achieve the end output of a transactional items association rules. The two main components of the algorithm are: identifying frequent item sets and determining the confidence of those frequent items. Frequent items are simply those that appear at a minimum number of occurrences within a transactional database. The minimum threshold to define a frequent item is passed into the algorithm as the support value variable. These frequent items are than used to determine if two items have any confidence correlation. Again, the confidence value to validate is passed to the algorithm via a confidence value variable.

All frequent item-sets are to be identified by scanning each transaction, looking for duplicate items across all transactions. For each duplicate occurrence of an individual item, a counter will be incremented to be

used to identify the number of occurrences of each individual item. This total count of individual occurrences is next compared against the minimum support value. If an item does not have enough occurrences in order meet the minimum support value, the infrequent item will be pruned from the array list of possible frequent items. The remaining items that have meet the minimum support threshold are considered "frequent" and will be used in the next step of the algorithm.

The algorithm will next take the individual frequent items and concatenate two unique items to form a new item set that contains two items. The new two item sets are then compared against the original transactions array to count the number of occurrences the two item-set items appear within each individual transaction. If both items appear within a single transaction, a counter is incremented to be used for calculating if the new item set will meet the minimum support requirement. With the count of occurrences two item sets, the same frequent calculation is used to determine if the two item sets have met the support threshold.

The algorithm will once again concatenate the unique individual items within each of these frequent two item-sets to form a new three item item-set. The new three item-set is once again compared against the original transaction array to count the number of occurrences the three item-set appears within each individual transaction. If all three of the items appear within an individual transaction, a counter is incremented to be used for calculating the minimum support requirement.

Finally, if there's only a single frequent item-set remaining, the algorithm will stop creating new combinations of item-sets. The algorithm will next move onto next major component: confidence calculation.

To calculate confidence calculation the algorithm will use the frequent item sets that have at least 2 items included. Association rules can't be created with single frequent items so we exclude those single items from the association rule calculation. The algorithm will step through each of the frequent item sets containing at least two unique items and create different unique combinations. Each of the combinations that exist will have a "left" side and a "right" side. The confidence calculation will use both the left and right side items frequency value as the numerator and the left side item's frequency value as the denominator. If the division of the numerator divided by the denominator is greater than or equal to the confidence value, an association rule is generated.

Saving Data as a .tab file

```
data.save("new_data.tab")
f = open('new_data.tab', 'r')
print f.read()
f.close()
```

We save our dataset as a new .tab file so as to ensure we're keeping the original data backed up for reference or further investigation, if necessary. This is a best practices step for ensuring your data doesn't get lost and a backup is always present.

Printing the count of instances of each itemset:

```
for inst in data:
    print inst
```

The last block of code will print out the individual items in each transaction along with a count of the instances of each item.

Results and Observations

Algorithm Results: Support=0.005, Confidence=0.8

Supp	Conf	Rule
0.008	0.889	frozen -> fish
0.009	0.947	bags -> cling_film
0.009	1.000	cling_film -> bags
0.009	1.000	photo -> film
0.009	1.000	film -> photo
0.022	1.000	red -> blush_wine
0.022	1.000	blush_wine -> red
0.009	1.000	seasonal_ -> products
0.022	0.977	vegetables -> packaged_fruit
0.022	1.000	packaged_fruit -> vegetables
0.062	1.000	vegetable_juice -> fruit
0.062	1.000	fruit -> vegetable_juice
0.005	0.909	cereals -> whole_milk
0.203	1.000	rolls -> buns
0.203	1.000	buns -> rolls

For an association rule to be considered 'interesting,' it needs to have a positive or negative interest value of above 0.5. In the association rules listed from the algorithm, we will list the most interesting rules.

The 'interest' formula used for determining the most interesting association rules is as follows:

Confidence value minus fraction of baskets that item on the right of the formula is listed.

Interesting association rules

0.872 – frozen -> fish

0.937 – bags -> cling_film

0.9015 – cling_film -> bags

0.982 - photo -> film

0.991 – film -> photo

0.9785 – red -> blush_wine

0.9785 – blush_wine -> red

0.9535 - seasonal_ -> products

0.9785 - vegetables -> packaged_fruit

0.6025 - packaged_fruit -> vegetables

0.647 - vegetable_juice -> fruit

0.9375 - fruit -> vegetable_juice

0.7415 - cereals -> whole_milk

0.797 - rolls -> buns

0.797 - buns -> rolls

```
Algorithm Results: Support=0.05, Confidence=0.8
   Supp     Conf   Rule
   0.062    1.000  vegetable_juice -> fruit
   0.062    1.000  fruit -> vegetable_juice
   0.203    1.000  rolls -> buns
   0.203    1.000  buns -> rolls
```

Interesting association rules

0.647 - vegetable_juice -> fruit

0.9375 - fruit -> vegetable_juice

0.797 - rolls -> buns

0.797 - buns -> rolls

```
Algorithm Results: Support=0.02, Confidence=0.75
   Supp     Conf   Rule
   0.022    1.000  red -> blush_wine
   0.022    1.000  blush_wine -> red
   0.022    0.977  vegetables -> packaged_fruit
   0.022    1.000  packaged_fruit -> vegetables
   0.062    1.000  vegetable_juice -> fruit
   0.062    1.000  fruit -> vegetable_juice
   0.203    1.000  rolls -> buns
   0.203    1.000  buns -> rolls
```

Interesting association rules

0.9785 – red -> blush_wine

0.9785 – blush_wine -> red

0.9785 - vegetables -> packaged_fruit

0.6025 - packaged_fruit -> vegetables

0.647 - vegetable_juice -> fruit

0.9375 - fruit -> vegetable_juice

0.797 - rolls -> buns

0.797 - buns -> rolls

Conclusion

Our team found that utilizing this method of association rule mining was incredibly effective. Python as a language contains multiple packages for data mining and machine learning. Its popularity also provides an extensive user base for feedback and support. Python is also an open source tool that does not require the use of expensive licensing requirements as is the case with MATLab and SPSS.

Association rule mining using the Apriori algorithm can be applied to a number of different transactional datasets. As our project progressed, we found that we were not only more cognizant of the algorithms use in storefronts like Amazon, Target, but also services like Netflix and Hulu. Each industry can use this algorithm for various techniques to modify the consumer/user experience; ultimately yielding a more customized experience for the user, but a more profitable experience for the stakeholder.