Tampere University

Hussain Syed

COMBINING ASSOCIATION RULE MINING AND CLUSTERING

Faculty of Information Technology and Communication Sciences
MDP in Computer Science. Thesis
October 2019

Abstract

Clustering and association analysis are known data mining techniques used for manipulation of data in processes such as machine learning, correlations, finding frequent itemset and hidden patterns in datasets. This work presents a practical implementation of both methods combined on different data sets to get association rules and clustering them based on the importance. Both techniques work around the idea of similarities and correlations between the elements within data. The work done discusses ideas and benefits of using similar approach in other fields and giving impact on decision makers.

A number of useful methods and concepts namely Apriori algorithm, K-means, data context, lattice window previously known to researchers and analysts are essential part of this implementation as they smooth the process in accordance to the data samples used in the study. These are also important to limitations and interpretation of the work done. At the same time, it is essential to know the right order while combining both clustering and association analysis which varies depending on the data and the mining problem. As main objective is to traverse through less data and focus on more interesting and correlated data, implementation is adjusted to use strengths of association analysis first as it suits the data set at hand.

Consequently, the implementation provides information and technique relevant to use this combination on other data sets intending to find similar results. This study is a useful tool to explore different combinations provided by these two data mining techniques and It also opens the door for more related research and application areas.

Key words: Data mining, association analysis, clustering, data context, Apriori algorithm.

Table of Contents

Table of Tables

Table of Figures

1. Introduction

Cluster analysis and association rule mining are helpful in solving data mining problems. Some of the information analysis and mining applications such as content mining, web mining, and data classification utilize both of these methods.

Clustering is a machine learning technique that involves grouping of data. In data science it can be used to gain valuable insight from data by seeing what groups a certain type of data falls into after applying a clustering algorithm. Cluster analysis is likely to come as a pivotal transitional stage for related data mining methods. For example, in creating a summarized report with all the necessary features and figures neatly fitted in one view of classified data objects, discovery of patterns, generation of hypothesis and testing, etc. It can likewise be utilized for anomaly discovery (anomalies that are not near any cluster), summarizing the data, compressing and reducing data, image processing, vector quantization, sequence analysis, human genetic clustering, market segmentation (which is very important to divide a broader consumer based), product positioning in order to enhance the sales, new product development and selecting test markets to make sales better in future are the key roles of clustering.

Clustering can be useful in retail business to group items according to their requirement and placement. Real activities in retail are the measure of stock you ought to have as per the necessities and the overall revenue it will convey regarding the things available. Both of these processes depend on grouping and categorizing items based on their movement in the later stages. Clustering strategies, for example, K-means have been utilized to construct models that can gather things into classes such as quick and moderate moving relying upon their necessity in the market and on the centroids of the clusters to be made. (Kusrini Kusrini, 2015). Apart from that clustering results can be used for itself as a stand-alone process when goal is to find grouping of data points with similar characteristics. Cluster analysis itself is not one specific algorithm but general task to be solved. It can be achieved by various algorithms even though their notion of what constitutes a cluster is significantly different and how to efficiently find them.

Association analysis is the task of uncovering relationships among data. Rules made as a result of analysis are a model that identifies how the data items are associated with each other. There are two key issues that needs to be addressed while applying association analysis. First step is finding the pattern, but it has to be kept in mind that some of the patterns are maybe happening only by chance. That is why after finding rules their strength is tested using certain measures. Market basket analysis describes the whole

process and can be seen as a good example to learn about the whole process. (Akash Rajak et al, 2008).

Market Basket Analysis is a generally known example of association rule mining that has helped marketer a great deal amid recent decades to support their business and enhanced racking of the popular things. Proteins are important ingredients of cellular mechanism inside the body of an organism. Genetic molecules formed by recombination advancements have created ways for the quick goals of DNA designs and, by deduction, the amino acid sequence of proteins. Clustering can help facilitate patterns of various proteins based on genetic or molecular groupings. Customer relationship management (CRM) is a model, with the assistance of which, banks try to flag the preferring different customer groups dependent on their requirements, items and administrations explicitly intended to their liking at the chance to support up the conservative connection among credit card holders and the bank, which has transformed in the form of a subject known as extraordinary significance. (Akash Rajak et al, 2008).

Due to large amount of data today and its ability to grow rapidly overnight, both cluster analysis and association rule mining face challenges such as time, complications in data, high dimensional data etc. There are many used clustering techniques that do not necessarily perform well in data mining scenarios. Some of it is due to data distribution issues and some are due to application constraints. At times the data to be processed is very large that most clustering algorithms become either too slow or they end up giving varying clusters where intra-cluster distances are not up to the desired minimum standards. Also, the clustering algorithms should be flexible for the scaling issues to be faced in later stages. At times quadric and cubic scaling may also be looked at but a more desirable outcomes are linear behavior of data.

Another important factor to consider here is of high dimensionality. The number of features in data to be processed can be high and can easily exceed the samples used. So, there is higher chance of facing curse of the dimensionality. Also, number of times, skewness of data is too much that it is difficult to mold it by normal distributions. Outliers in the data can have significance of their own and it is not advisable to ignore all of them. Locating these is usually important and purging them completely is not necessarily required.

Large systems have heterogeneous sources of data that are distributed evenly. Clustering results from local systems have to be transformed into global models to be available for other stakeholders. (Alexander, 2002).

Frequent itemset mining is already a difficult problem that comes along with many challenges. Due to combinational explosions it is possible that number of frequent itemset is large and we face the challenge how to enumerate, explore and save them.

Accessing data from a single source is an important factor while managing high volumes of data. In most of the data streams that usually needs to be handled in inventory business and other related data sets are always flowing at a very high speed and in large size. As a result, it is not possible to store all of that on solid state drives and getting access to that data multiple times can also turn out to be expensive. So, the main focus is to find frequent itemset where data usually needs to be accessed only once.

Getting timely access to data is highly desirable and most of the time response needs to be real time. As most these data applications on these streams are time critical, there are some standards that should be fulfilled in order to facilitate timely access to data. In some situations, algorithms that are slower than the rate at which that data is coming turned out to be useless, so there is a need to mine frequent itemsets in real time (Mahesh Kr Singh et el, 2012).

Most important of all these challenges is time in case of very large data sets. Cases where either of these analyses cannot be ignored, the combination of both association analysis and clustering would help better understand the data in a shorter amount of time. This is something that has been done previously to some extent. Here, goal is to rigorously apply these techniques on different data sets and compare the results. Similarities between clustering and association analysis has motivated researchers to use hybrid of both methodologies on challenging datasets.

Both these mining methods have basic principle of analyzing relationship between elements of data. In the following chapters idea of joining cluster analysis with association analysis centers on value-based databases. As association analysis typically involves transactional databases where it is easier to generate rules while checking the presence of items in order to get specific values for measures (support, confidence etc). In case of non-transactional databases some sort of intermediary method is used to represent the data with a binary coding (0,1 or Yes, No) and then rules are created.

The aim of this study is to discover advantage of using both clustering and association analysis on a given data set and how it can be beneficial in terms of making decisions by extracting useful statistical information. It is also an attempt to know what are the bottlenecks that keeps us from combining the two methods frequently or using either of them. The primary goal is to get familiarized with the combination of association rule mining and clustering as data analysis techniques and also build on to the learning curve that comes with using both on interesting data sets. When used separately both association analysis and clustering have very high time cost to deal with large sets of

data. That is why combination of both is more feasible according to task at hand. One way is to apply association analysis on the given data and then make clusters for more interesting rules. An algorithm has been devised for that task which also considers converting data into more relatable objects (data context, concept lattice, formal concept), before applying association rules (Fu, 2008). Another approach is to cluster variables to build homogeneous group of attributes and then mine association rules inside of each group (Plasse, Niang, et al, 2007). In the analysis of this study the first approach has been considered as it is more suitable for the sample data in question. Also, it is easier to understand and implement. After finding the frequent items in the data set association rules have been generated and then K-means clustering technique is used to group the rules that can be more useful.

In chapter 2, clustering analysis is explained, and types of clustering are discussed. Simple diagrams and equations are used to explain the processes graphically. In chapter 3, details of association analysis are defined and explained how it works. Apriori algorithm and how it is used to generate frequent itemsets is demonstrated on a simple example. Chapter 4 explains in detail the research work done in combining both of these methods. Techniques used after getting insight from the research previously done, flow of the work and generated results are given in chapter 5. Chapter 6 gives conclusion to the work done and room of improvement.

2. Clustering

2.1 Introduction

Unsupervised machine learning problem in the domain of AI infers patterns from a dataset without reference to known, or labelled, outcomes. In this process, an effort is made to model the latent structured information present within the data. Clustering can be well-thought-out as the most pressing and critical unsupervised learning solution to date, so, it usually works out by finding a certain structure in a group of unlabelled data. This is mainly because when there is a problem when the goal is to determine the intrinsic grouping in a set of unlabelled data, clustering seems to be the suitable solution. Also, in certain situations clustering is used as a preparation of data for other AI techniques. An unformal definition for the process of clustering can be described as the procedure of shaping items into certain groups in which members are alike in some means. Cluster analysis is the process of where we try to group similar data based on their underlying pattern into clusters. An important issue in clustering is how to define the resemblance and similarity of two objects inside the cluster, so eventually clusters can come into existence from objects which have high similarity. Mostly while finding similarities between the objects we consider, distance functions, such as greatest distance, Mahalanobis distance, Manhattan and Euclidian distance etc.

Euclidean distance is D_1 (A, B) is $\sqrt{\sum_{k=1}^{n}(a_i - b_i)^2}$,

Squared Euclidean distance is D_2 (A, B) = (D_1 (A, B))^2

Manhattan distance: D_3 (A, B) is $\sum_{k=1}^{n}|(a_k - b_k)|$,

Greatest distance: D_4 (A, B) is $max_k = \{|a_k - b_k|\}$ and

Mahalanobis distance D_5 (A, B) is $\sqrt{((\mathrm{A}-\mathrm{B})^T\, S^{-1}(\mathrm{A}-\mathrm{B}))}$ S is a covariance matrix
The elements of the formulas calculating the distance D, can be described as follows:
For example, in Manhattan distance between two vectors, A, B, in an n-dimensional real vector space, with fixed coordinates, is the sum of length of projections between the points where (A,B) are formed with vectors A $(a_1, a_2, a_3, \dots, a_n)$ and B $(b_1, b_2, b_3, \dots, b_n)$ of dimension n. The same applies for the rest of the equations. (Mishra, 2017).

Cluster can be defined as a group of items that are much alike each other and are unrelated to the items which belong to the other clusters. A simple ideal sample of clustering data points is given in figure 1.

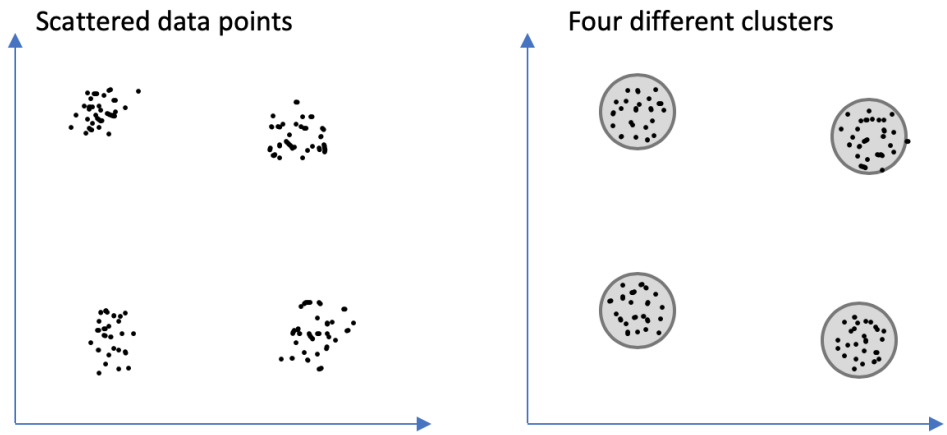

Figure 1 : Clustering data points

Four clusters are identified in the above example within which one can divide the data. Distance is used here as the similarity measure. If the measure (in this case distance) used between the data points is less they belong to the same cluster. It is called a geometric distance. It is known as distance-based clustering. (Mishra, 2017).

Use cases of clustering:

Clustering is widely used today for problem solving. Some of the core use case are as follows:

- Segmentation and exploratory data analysis are widely used in telecom industry for more revenue.
- Data scientists extensively use clustering for filling missing values in the domain of social network analysis, as like-minded people have like-minded attributes.
- Pattern recognition in number of different ways.

Types of Clustering

There are different kinds of clustering approaches, which include:

- Soft Clustering
- Hard Clustering

In terms of hard clustering, every member object either belongs to the cluster completely or it does not belong to the cluster. It is based on classical set theory that an element belongs to one set or other but not both. Whereas in soft clustering aka fuzzy clustering, each data point has probability in each of the like cluster. It allows the object to belong to several clusters simultaneously. In many situations, soft clustering is more natural than hard clustering.

2.2 Partitioning Clustering Model

In a set of n data points, the partitioning method splits k partitions of data, where every single partition represents a particular cluster and $k \leq n$. An objective function is used to make clusters. Clustering must have two characteristics: (1) each data point must exist in one cluster and (2) each cluster must have at least one data point. Most commonly used partitioning clustering models is k-means.

K-means method is a commonly used clustering algorithm where each data point will belong to any of K clusters. K-means algorithm's output is a set of cluster labels which assigns each data point to one of the k groups. In k-means clustering, there will always be a centroid for each group. The centroids are like the heart of the cluster, which consist of the closest points. A larger number of clusters have less data points with more granularity, a low number of clusters means more data points to one group and less granularity. Overall, a large number of clusters probably decreases the error but can lead to overfitting. The goal is to have less of total intra-cluster variance and squared error function. (Mishra, 2017). Squared error function is

$$J = \sum_{j=1}^{k} \sum_{i=1}^{n} ||x_i^{(j)} - c_j||^2$$

where

$$||x_i^{(j)} - c_j||^2$$

is the distance measure in this case between data point x_i and cluster centroid c_j, (an indicator of the distance of n data points from respective cluster centroid).
In case of intra-cluster variance, the same distance measure J will be calculated for different clusters hence if one of the measures has been calculated accurately the other can be calculated on others expense.

K-means Algorithm:

1. Select K random clusters.
2. Initialize centroid for each cluster

3. Loop

    a) Similarity measure (distance etc) for each data point and centroid is calculated using necessary metrics. All data points are associated with their nearest cluster.
    b) New centroid is computed by taking the mean of the attribute in each cluster.
    Until centroid does not change.

Some of the real use cases of K-means clustering is segmentation and classifying handwritten digits. (John A Hartigan et al 1979)

2.3 Connectivity/ Hierarchal Clustering models

The main idea behind this type of clustering is that data points are more similar to nearby objects as compared to far away objects in data space/plane. It groups the set of nearby data points in such a way that clusters are similar internally. Furthermore, these similar data points produce the hierarchal series, then these series are joined to produce a hierarchy of clusters so as the name suggests, hierarchy of clusters is built in hierarchal clustering. Hierarchy is represented by inverted tree known as dendrogram in figure 2.

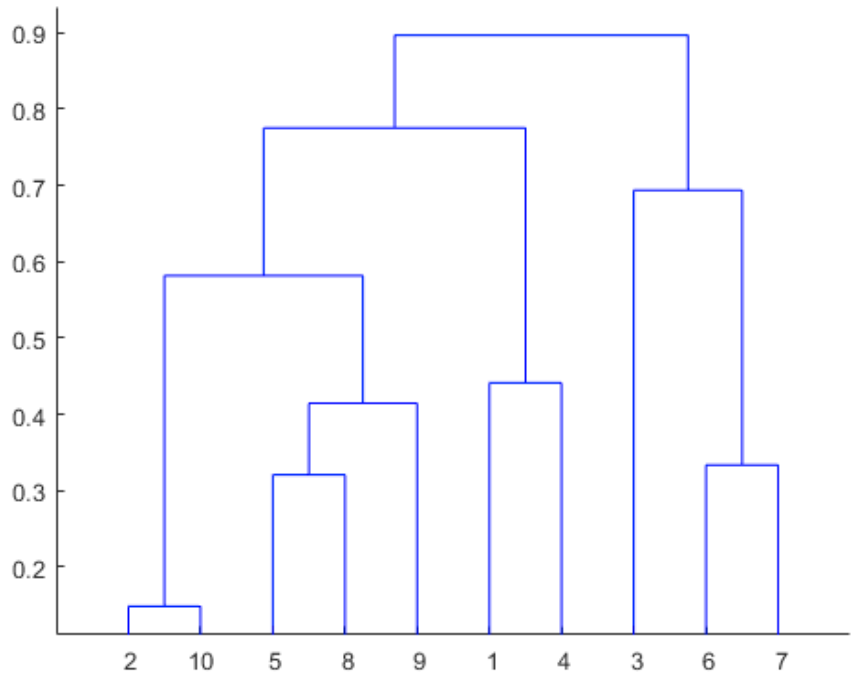

Figure 2 Dendrogram made in Matlab

Each leave of the dendrogram shows one case of data. First, similar leaves of the tree joined to make different clusters, then these lower level clusters join with the next

level cluster depending on how much similar they are. All these nodes then end up into full tree. (Berkhin, 2006)

Two types of hierarchal clustering are there as explained briefly below.

(i) Agglomerative clustering (bottom-up)

In agglomerative clustering the main idea behind the algorithm is to combine smaller clusters of slightly similar characteristics into larger clusters until one big cluster is formed. It starts with one data point as one cluster. Then it starts combining similar points and ends up into tree. There are many ways to do these by using common methods defining the pairwise distances between clusters such as complete linkage, single linkage, group averages and centroid similarities. (Berkhin, 2006)

(ii) Divisive clustering (top-down)

In divisive clustering, one big large cluster is divided into smaller clusters until each cluster has only one data point that's why it is called top down approach. It is less used because of its complexity as compared to the agglomerative clustering. (Berkhin, 2006)

Advantages of hierarchical clustering includes:

- It can perform in any level of granularity
- Any similarity matric can be used.
- It's applicability to any attribute types

Some of the real use cases of hierarchal clustering are DNA sequencing and computer virus tracking (Liu, Libin et al, 2006).

2.4 Density-Based Clustering models

Density based calculation keeps on developing the given group as long as the thickness (areas where there are higher number of items than the remainder of the dataset) in the area surpasses certain limit. This calculation is reasonable for dealing with noise in the dataset. The accompanying focuses are identified as the highlights of this calculation.

1. Handles groups of arbitrary shape (thick areas of items in the dataspace that are isolated by the locales of low thickness).

2. Handle noise (handling of erroneous data, data items which are not important to the clustering tasks and can degrade the performance of the clustering algorithm).

3. Needs just a single sweep of the dataset.
4. Needs thickness parameters to be introduced.

DBSCAN, DENCLUE and OPTICS are precedents for this calculation.

2.4.1 Model-Based Clustering models.

Partitioning and connectivity clustering algorithms are exploratory, so different runs of algorithms will often yield different results. So, one can say that they are not based on any formal formula or model. To deal with these types of challenges we have model-based clustering that attempts to optimize the fit between data and model. A technique of model-based clustering is:

- Expectation Maximization

  Parametric probability distribution is used to define clusters. This algorithm finds the parameters that characterize each group. In simple words, given some data, the algorithm computes the value of a parameter that best explains the data. Each component (i.e. cluster) k is modelled by the normal or Gaussian distribution which is characterized by the parameters. (Berkhin, 2006)

- Conceptual Clustering

  These algorithms produce a classification schema for the set of unlabeled data points by finding the characteristics description for each class. E.g. COWEB clustering method. (Berkhin, 2006)

2.4.2 Graph Based Clustering

Graphs structures have set of vertices known as nodes and a set of edges that are connecting the pairs of vertices. With the improvement in clustering algorithms, recent advancement has occurred in graph clustering. In graph clustering, algorithms cluster nodes in such a way that inter cluster edges are maximum whereas intra cluster edges are minimum. More formal definition “Partition a graph into natural groups so that the nodes

in the same cluster are closer to each other than those in other cluster". This type of clustering falls in the category of within graph clustering which is widely used in social network analysis. Clustering different graphs is known as between graph clustering, for example, clustering similar structures of chemical compounds. Procedure of graph clustering is given in figure 3. (Schaeffer, 2007)

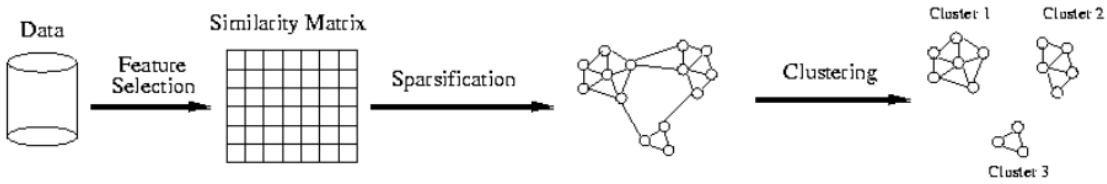

Figure 3 Procedure of graph clustering (Berkhin, Pavel,2006)

Graph clustering uses different principles in different algorithms, some of the principles are described as follow;

Nearest Neighbour Searching: In this the data point (i.e. node in case of graph) have the same characteristics as that of the nearest point.

High Dimensional Space: It is the extension of regular graph, but it does not have any limiting conditions that edge should be joining two vertices only, it can also join more than two.

Sparsify the input matrix: Sparsifying the similarity matrix before beginning the actual clustering process will reduce complexity and perform better. (Berkhin, 2006)

Popular algorithms within graph cluster algorithms are as follows

- K-Spanning Tree,
- Shared Nearest Neighbor,
- Betweenness Centrality Based,
- Highly Connected Components and
- Kernel K-means

3 Association Analysis

Association rules are widely used in machine learning to identify the association between items, thus patterns between items are identified. It uses the concept of co-occurrences not causality, whereas co-occurrence defines as the more items occur together. It analyses transaction data and is intended to identify strong rules based on different measures like support, confidence and lift. It is a rule-based model, or we can say it is if and then statement that describes an interesting relation between data and finds frequent patterns between data.

Today association rules are commonly used in different domains. Market basket investigation can be utilized to strategically pitch items. Amazon broadly utilizes an algorithm to recommend things that you may be keen on, in view of your purchasing history or what other individuals have acquired. An example is that a supermarket, after executing marked basket analysis, found that men were probably going to purchase beer and diapers together. Sales expanded by putting brew beside the diapers. (Mittal, Pareek Agarwal, 2015)

3.1 Association rules mining and interestingness measures

To get a better understanding of the tasks ahead related to mining association rules, there are the interestingness measures that are usually kept into consideration.

Support

Frequency of occurrence for itemset is called support. We have to determine some threshold and drop all the items that have less support than the threshold. Support is given by:

$$Support\ (A->B) = \frac{Freqeuncy\ (A\ and\ B)}{N}$$

For example, if we only want to examine item sets that occur in 2 out of 5 transactions, then support should be set as 0.4, or 40%.

Confidence

Confidence is interpreted as the conditional probability of {d} given {c}, where {c} is the antecedent and {d} is the consequent of the rule. In other words, we can say that if item A is chosen how likely item B is also chosen. The maximum value of confidence is 1.

$$Confidence\ (A->B) = P\ (B \mid A)$$

For example, if we have found a rule of {c} in association to {d}, we would want to know how often this rule is true in the entire dataset.

Transactions	Books
T-001	{Scala codebook, learning scala }
T-002	{ Scala codebook, Doing Data science, Learning Spark, Hadoop}
T-003	{learning scala, Doing Data science, Learning Spark, Advance analytics with spark }
T-004	{Scala codebook, learning scala, Doing Data science, Learning Spark }
T-005	{ Scala codebook, learning scala, Doing Data science, Advance analytics with spark }

Table 1: Transactional Data

In the table 1 above, there are 5 orders in total and {Scala codebook, learning scala} occurs in 3 of them, so:

support{Scala codebook, learning scala} = 3/5 or 60%

"Given two items, A and B, confidence measures the percentage of times that item B is purchased, given that item A was purchased. It can be stated as:

confidence{A→B} = support{A,B} / support{A}

Confidence values range from 0 to 1, where 0 indicates that B is never purchased when A is purchased, and 1 indicates that B is always purchased whenever A is purchased. Note that the confidence measure is directional. This means that we can also compute the percentage of times that item A is purchased, given that item B was purchased:

confidence{B→A} = support{A,B} / support{B}"
(Vijay Raghani, 2019)

confidence{Scala codebook, learning scala} = support{Scala codebook, learning scala} / support{Scala codebook}

= (3/4)

= .75 or 75%

A confidence value of 1 implies that out of all orders that contain Scala codebook, 75% of them also contain learning scala. Now, we look at the confidence measure in the opposite direction (i.e. learning scala → Scala codebook):

confidence{learning scala → Scala codebook } = support{ Scala codebook, learning scala} / support{ learning scala}

= (3/4)

= .75 or 75%

Lift

Lift is a measure that assesses both the confidence probability of consequent in the whole dataset. It shows the independency or dependency between {c} and {d}. When value of lift is 1, that implies {c} and {d} are independent to one another.
When value of lift is over 1, it implies that {c} and {d} are positively correlated (negatively correlated if lift is under 1) and could be valuable in future predictions.

"Given items, A and B, lift shows whether there is a relationship between A and B, or whether the items A and B are occurring together in the same orders simply by chance (ie: at random). Unlike the confidence metric whose value may vary depending on direction (eg: confidence{A->B} may be different from confidence{B->A}), lift has no direction. This means that the lift{A,B} is always equal to the lift{B,A}:

lift{A,B} = lift{B,A} = support{A,B} / (support{A} * support{B})

Through our example, lift can be computed as follows:

lift {Scala codebook→ Learning scala} = support {Scala codebook, Learning scala} / (support {Scala codebook} * support {Learning scala})

= (3/5) / (4/5 * 4/5)

= 0.937 "

(Vijay Raghani, 2019)

One way to understand lift is to think of the denominator as the likelihood that A and B will appear in the same transaction if there was no relationship between them. In the example above, if Scala codebook occurred in 80% of the transactions and learning scala occurred in 80% of the transactions, then if there was no relationship between them, we would expect both of them to show up together in the same order 64% of the time (ie: 80% * 80%). The numerator represents how often Scala codebook and learning scala actually appear together in the same transaction. In this example, that is 60% of the time. Taking the numerator and dividing it by the denominator, we get to how many more times Scala codebook and learning scala actually appear in the same transaction, compared to

if there was no relationship between them (ie: that they are occurring together simply at random).

"In summary, lift can take on the following values:
* lift = 1 → no relationship between A and B.
(ie: A and B occur together only by chance)

* lift > 1 → that there is a positive relationship between A and B.
(ie: A and B occur together more often than random)

* lift < 1 → that there is a negative relationship between A and B.
(ie: A and B occur together less often than random)"
(Jain, 2018)

3.2 Frequent Item Set Generation:

A lattice is an easy way to go through the list of possible items available in the data. Figure 5 below shows a lattice containing the following elements in set I = (A, B, C, D, E). An itemset with k items will eventually have up to $2^n - 1$ frequent item sets, excluding the empty set. In most of the practical applications that are used in data mining the number k is usually huge, hence the search space of item sets that are required to be observed, will grow exponentially. (Tan, Steinbach, Karpatne, Kumar, 2015)

There is a brute force technique to find frequent item sets in order to calculate support count of candidate itemset. To do this each candidate itemset will be compared with every transaction in the database as shown in figure 4. If the candidate exists in the given transaction, variable carrying the support count will be increased by one. As according to the figure 4, the support for item set {Scala Codebook, learning scala} is 3 as the item set is present in transactions 1, 4, and 5. This technique is costly in terms of computation time required as it takes O(NMw) traversing, Variables in the parenthesis can be broken down as under:

N = transactions, M = $2^n - 1$ (candidate itemsets), w = max transaction width.

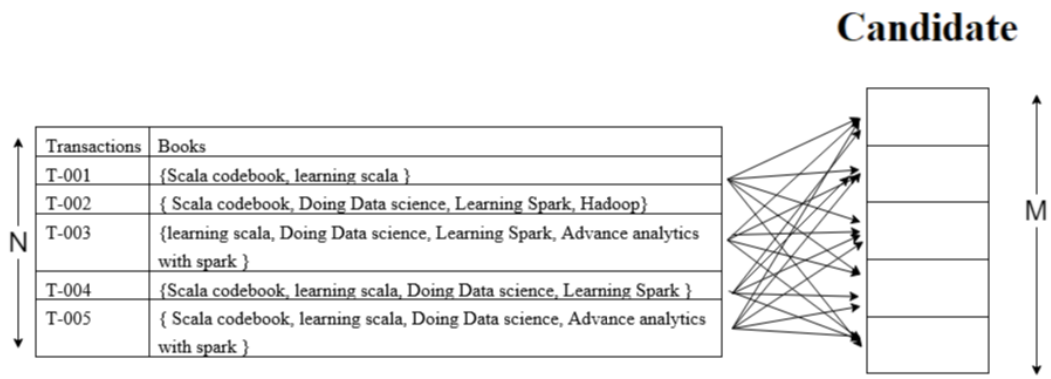

Figure 4 : Support Count of the candidate sets

Computational time to calculate frequent itemsets can be reduced in a number of ways.

1. One simple way is decreasing candidate itemsets (M). Apriori algorithm, mentioned ahead in detail, comes up with a very effective technique to get rid of the candidate itemsets that are not required by taking into account their support.

2. The number of times candidate itemsets traverse against the transactions to make the comparisons can be decreased as well. In this way there is no computations of comparisons of candidate itemset to every transaction. More advanced data structures are used in order to avoid further comparisons and those structures can either store candidate itemsets or compress data. The idea of monotonicity and use of frequent itemset is used by Apriori to generate frequent itemsets. The algorithm is most common when it comes to association analysis as it is easy to implement and can be parallelized according to the needs. Although in some cases it might require more data scans than initially anticipated. (Tan, Steinbach, Karpatne, Kumar, 2015)

The next section walks through the working of Apriori with the help of a simple example.

3.3 The Apriori Algorithm

This section describes how support count can help reducing the number of candidates generated during the process above.

Item set is frequent then its subsets are similarly frequent.

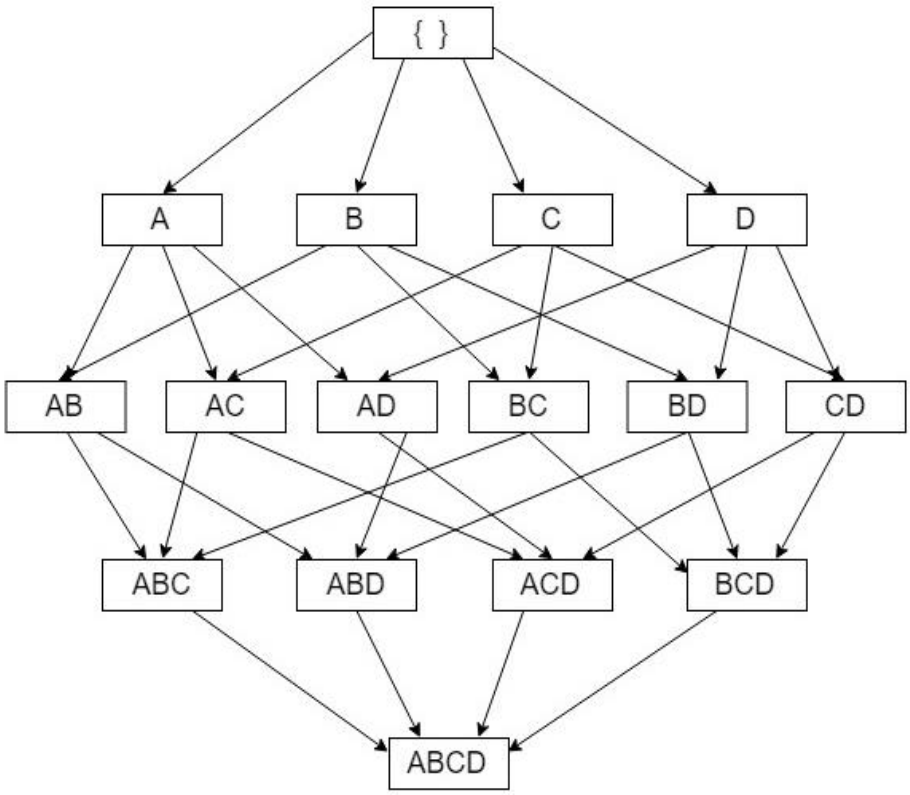

Figure 5 : Frequent Item Set graph

In order to demonstrate the key characteristic for Apriori algorithm, itemset lattice in figure 5 is considered. Assuming {B, C, D} as a frequent item set. This means that, a transaction that has the set { B, C, D } which can be any, should definitely have the sets, {B, C}, {C, D}, {D, B}, {B, D}, {B}, {C}, {D}. So, we can conclude that, whenever {B, C, D} is frequent, it is assured that subdivisions are definitely frequent which consists of upper right area in figure. (Tan, Steinbach, Karpatne, Kumar, 2015)

On the other hand, when we examine a specific case, let's assume that item set {A, B} is not frequent, all its supersets are obliged to be non-ferquent. The lattice clearly demonstrates the subgraph that has supersets {A,B}. Such technique which is used to find the frequent and non-frequent itemsets based on their support count is called support-based pruning. This is based on the property that support count of a superset never surpasses the support count of its subset. In theory such attribute is called the anti-

monotone of support count. Shaded area in figure 6 below show the non-frequent itemset. (Tan, Steinbach, Karpatne, Kumar, 2015)

Let I = set of items K = 2^I. Measure f is monotone if:

$\forall$ x, y $\in$ K: (x $\subseteq$ y) $\rightarrow$ f (x) $\leq$ f (y),

When x is subset of y, f(x) must never surpass f(y). Contrary side, f is anti-monotone when

$\forall$ x, y $\in$ K: (x $\subseteq$ y) $\rightarrow$ f (y) $\leq$ f (x),

when x is subset of y, f(y) should never surpass f(x).

Any measure having the anti-monotone property can be included in a frequent itemset mining algorithm to effectively trim the search space of candidates.

Figure 6:

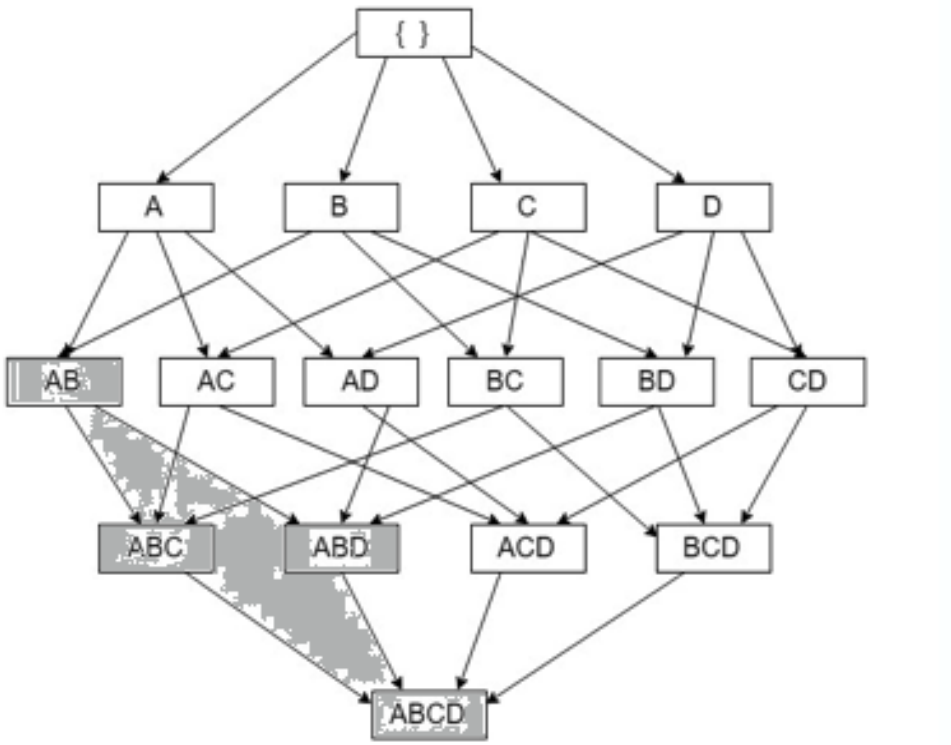

Figure 6: Frequent Item Set graph explanation.

Creation of frequent item set using the Apriori algorithm

The first algorithm along with other algorithms to use the support count trimming technique is Apriori. It can easily control the exponential growth of candidates in a systematic and understandable manner which is handy to most researchers. The example follows (Tan, Steinbach, Karpatne, Kumar, 2015) gives us a top-level demonstration of

a frequent item set creation as part belonging to Apriori algorithm for the transactions in table 1.

Minimum Support Count = 3

Item	Count
Scala Codebook	4
Learning Scala	4
Learning Spark	3
Advance Analytics with spark	2
Hadoop	1
Doing Data Science	4

Table 2: Single Itemset

Item	Count
{Learning Spark, Scala Codebook}	2
{Learning Spark,.Doing Data Science}	3
{Learning Spark, Learning Scala}	2
{Scala Codebook, Doing data science}	3
{Scala Codebook, Learning Scala}	3
{Doing data science, Learning Scala}	3

Table 3 : Calculated Pairwise Itemset

Item	Count
Scala Codebook, Doing Data Science, Learning Scala	2

Table 4 : Calculated Thrice Itemset

In the beginning every object is candidate 1-item set. After the first repetition and counting support count of all objects available, when the candidate item sets {Advance analytics with spark} and {Hadoop}appear to be in less than 3 transactions (minimum support = 3), they are eliminated. During the next step, candidate 2-item sets are created using frequent 1-item sets as the Apriori rule confirms that superset of non-frequent 1-

item sets should also be non-frequent. As, there were four 1-item sets previously that satisfies minimum support count, the total of candidate 2-item sets returned by running the algorithm is C (4,2) combinations = 6.
Two candidates out of these six, {Learning Spark, Scala Codebook} and {Learning Spark, Learning Scala}, also originate to be non-frequent after examining their support values. The other four candidates are frequent, they are further used to create candidate 3-item sets. when support-based pruning is not used, there will be C (6,3) combinations = 20 candidate 3-item sets that can be created using the six objects shown in Table 1. Using apriori principle, the only requirement is to keep candidate 3-item sets which has frequent subsets. The candidate to have that attribute is {Scala Codebook, Doing Data Science, Learning Scala}.

The usefulness of the Apriori trimming approach can be revealed by counting the candidate item sets created. Using brute-force approach of numbering all of the item sets (up to size 3) as candidates will yield

C(6,1) + C(6,2) + C(6,3) = 41 candidates. Using Apriori principle with support-based pruning, 6 + 6 + 1 = 13 candidates are considered.

These candidates signify a 68% decrease in the number of candidate item sets even in the simple example like above.

The pseudocode for the frequent item set creation section of the Apriori algorithm is explained below in accordance with the steps followed in the above example. Assume Ck represents a set of candidate k-item sets and Fk represent the set of frequent k-item sets:

• The algorithm at first, makes a solo pass over the data set to count the support of each object. When this step is achieved, the set of all frequent 1-item sets, F1, will be identified. (Tan, Steinbach, Karpatne, Kumar, 2015)

• In the next step, the algorithm will iteratively create new candidate k-item sets using the frequent (k − 1)-item sets returned in the preceding repetition.

Algorithm Frequent item set creation of the Apriori algorithm.

k = 1.

Fk = { i | i ∈ I ∧ σ({i}) ≥ N × minimum support}. {Find all frequent 1-item sets}
repeat

k = k + 1.

Ck = apriori-gen(Fk−1). {create candidate item sets}

for each transaction t ∈ T do

Ct = subset (Ck, t). {Recognize all candidates that belong to t}

for each candidate itemset c ∈ Ct do

σ(c) = σ(c) + 1. {Increment support count}

end for
Fk = { c | c ∈ Ck ∧ σ(c) ≥ N × minimum support}.
until Fk = ∅
{Extract the frequent k-item sets}
Result = Fk.
(Tan, Steinbach, Karpatne, Kumar, 2015)

Closed Frequent item set:
It is a frequent item set which is essentially both closed and its support is greater than or equal to minimum support.
An item set is closed in a data set if there exists no superset that has the same support count as this original item set.
First, find all the frequent item sets. Then from this group find the ones that are closed by inspecting to see if there happens to be a superset that has the same support as the frequent item set, if there happens to be one, that item set is disqualified, but if none of the item set is found, the item set is closed. (Han, Kamber 2012)
Another method is that, we can first recognize the closed item sets and then utilize the minimum support to govern which ones are frequent.
In conclusion, it is a central goal to figure out the connection among frequent item sets, closed frequent item sets and maximal frequent item sets. As mentioned earlier closed and maximal frequent item sets are subsets of frequent item sets but the maximal frequent item sets are a denser depiction because it is a subset of closed frequent item sets. Closed frequent item sets are more broadly used than maximal frequent item set because when effectiveness is more important than space, they provide us with the support of the subsets, so no extra pass is required to find this information. (Tan, Steinbach, Karpatne Kumar, 2015)

3.4 Generating association rules from frequent itemsets.

The process of generating strong association rules based on frequent itemsets is straightforward. All that need is for the ones which have minimum support and confidence, keeping in mind the definitions for confidence and support explained earlier.

→ for each frequent itemset, generate all non-empty subsets.
→ for every non-empty subset output the rule, if the ratio
support_count(fis)/supportcount(ss)>= minimum confidence.
//Where as fis is (frequent itemset) and ss is (non-empty subset)

As the rules are created from frequent itemsets each one automatically satisfies the minimum support.
Trying the simple example from the transactional data given in the table one here are some of the rules that can be generated. The data contained the frequent itemset say X= {Scala Codebook, Doing Data Science, Learning Scala}. The non-empty subsets of X are {Scala Codebook, Doing Data Science}, {Doing Data Science, Learning Scala}, {Scala Codebook, Learning Scala}, {Scala Codebook}, {Doing Data Science} and {Learning Scala}.

Few of the rules that can be generated are:
{Scala Codebook, Doing Data Science}→{Learning Scala}, confidence =2/3=66%
{Doing Data Science, Learning Scala}→{Scala Codebook}, confidence=2/3=66%
{Doing Data Science}→{Scala Codebook, Learning Scala}, confidence=1/4=25%

If the minimum confidence threshold is set to 60% then first two rules will be considered as strong. Association rules can have more than one item on the right side of the rule as seen in the third rule generated above. (Han, Kamber 2012)

4 Related research work

There are a number of studies done for using association rule mining on top of a clustering algorithm and vice versa. Each study, although slightly different in actual flow work has produced very productive work that paved the way for studies and research work that followed. Here are some mentions in previous years.

4.1 Association Rules and Clustering:

Clustered association rules consider clustering rules that have value ranges using inequalities. This approach is helpful in reducing the number of rules that are typically generated by existing algorithms, therefore making the rendered rules easier to understand. An algorithm bitOp is embedded with association rule mining and clustering system which also introduces concept of range association rules such as (age=>50) ^ (salary = 40,000) => (owns_property=yes). (Brian Lent, et al, 1997)

Phases of the whole procedure are as follows:

- Heuristic Optimizer →Record Data→Binner
- Association Rule Engine→Segmentation Criteria within the Engine
- Clustering→Clustered AA →test Data → Verifier.

Before digging further into the bitOp algorithm and rest of the process that leads to clustered association rules, it is of high importance to comprehend two-dimensional association rules. It is the basis of how rules will be made in this study. The basic example is for association rules for two numeric attributes and one Boolean attribute. For example, if we take an example of the bank data base where age and balance are considered as numerical attributes or non-categorical if we consider the terminologies used earlier. One of the Boolean attributes in the same database could be card loan. Now considering the point (age, balance) in two-dimensional space, an association rule can be of the form:
(age, balance) $\in$ P → card loan = Yes. To generate rules (two-dimensional association rules) that are dependent on more than one attribute so that the probability that they will satisfy a certain condition, the following example shows how two-dimensional associations can work for any row R.
For row R, assume R[a], R[b] are two number attributes values;
R[a] = age while R[b] = Balance. Then R is mapped as (R[a]; R[b]) in Euclidean plane E for region K in E, if row R meets the state (Age; Balance) belongs to P, if row R is mapped to the point P.

The requirement is to recognize rules of the form (a,b ∈ P) → C:
To interpret it in simpler words the rule can be written as: if (balance, age) in P → C(loan approved) = yes.
C is the set of loan approved.
In reality there are a huge number of rows in the database framework subsequently and there is a need to oversee a great many indexes which can consume more room in the principle memory. To manage this issue, estimations of numeric characteristics are circulated into N buckets. The Euclidean plane gets divided into N x N pixel squares and row R has to be mapped to a single pixel that has point (R[a], R[b]). The area of a two-dimensional association rule is a combination of pixels. The probability that the area fulfilled the condition C is known as the confidence of region. It is critical to discover an area whose certainty is over a specific limit required by the stakeholders. The shape of the region is essential for getting great association rule. For instance Union of specific pixels has confidence over a specific edge, and combination of these pixels is characterized as region P, then P is confident with high support. (Takeshi Fukuda, et al, 1996).
The most important phase of the process is Clustering of the rules. The main BitOp algorithm will come under this section. At first, it will enumerate all the clusters of the grid. Then it will choose the biggest cluster from the list, iteratively applying calculation until the point when no further cluster are left in the list. This greedy methodology will deliver ideal clusters in O(Sum|S|) time, where S is last arrangement of cluster. This approach provided the same results as non-greedy approach but in better time frame. At any step during the clustering process, location of a cluster can be determined using the bitmap by ANDing of the bits. Another important step in clustering is pruning which helps better understand the process. Clusters found by BitOp that don't meet the required criteria are progressively pruned from the last set. Generally the number of clusters that are not valuable is not high. The process of pruning smaller clusters helps reducing outliers and noise. Although if all clusters are sufficiently large, pruning is not required. The idea of pruning comes from AI and decision trees. (Brian Lent, et al, 1997)

It is of high value that rules extracted from the data for a particular purpose are easy to understand and use for the stakeholders. Clustering association rules as seen in the example above are instrumental in decreasing the rule count that are extracted from used methods.

Results have shown the productiveness of the combination of these methods and demonstrate how this new system of producing association rules can be scaled more easily than the linear with respect to data to be handled. Because clustering in this case is

a conjunction of adjacent dimensional values or baskets of dimensional values, it always makes more sense to make these range-oriented rules in case of very large databases. It takes expression of the form $X_c \rightarrow Y_c$. X_c and Y_c are items of the form (att = val) OR (bin <att <bin i+1) where bin on either side defines upper and lower bound of the rule.

The problem can be considered of the procedure A & B → C where LHS attributes (left hand side) is quantitative while RHS attributes (right hand side) categorical. Once association rules are in place for a given support and confidence, table of those rules that give useful information on RHS is formed. BitOP algorithm is applied to this table to do the clustering of adjacent rules. (Brian Lent, et al, 1997)

Accuracy of the ARCS was tried on various databases and was contrasted with past techniques and calculations that are known for building profoundly precise decision trees so as to arrange data. From these trees a method called c4.5 rules generate the final set of rules that were similar in form compared to clustered association rules. In case of c4.5 rules, when it comes to larger databases there was a depletion of virtual memory which resulted into its inability to get desired results. Obviously c4.5 isn't suited for substantial data. Additionally, it includes some major disadvantages of delivering more rules that are less imperative when contrasted with ARCS which prunes out rules. Remembering that guidelines are handled by front end user keeping number of rules little is very crucial.

Similarly, as it comes to scalability, ARCS has been tested on different databases of various sizes. Since it maintains a BinArray and a bitmap grid, this algorithm only requires constant RAM irrespective to the extent of data, which makes this algorithm unique and more useful. (Brian Lent, et al, 1997)

4.2 EOQ approximation for flawed quality objects using association rules mining with clustering.

Inventory systems are usually used to see the flow of products as it involves monitoring the units in the organization and trying to prevent the quantity of a particular product from getting too high or low. Effective management is to keep the right quantity in right place for a product on a given time. In some situations, there are defective items that need to be considered. There are usually result of man-handling or technical faults. To properly ascertain the role of defective items in more realistic situations researchers have been developing EOQ models. The studies show that combined use of association rules and clustering methods is more applicable on EOQ estimation. Previously researchers have not explored the joint effect of cross selling effect, association rule mining and clustering. Keeping in mind this motivation, this paper brings focus on EOQ estimation using apriori algorithm with clustering and without clustering the transactions.

This technique gives important increase in number of rules which brings higher expected profit on the retailer's side. (Mittal, Pareek, Agarwal, 2015)

This paper presents strategy for reorganizing the ordering policy by introducing cross selling effect. Cross selling is a method of suggested related services or products to customers based on their buying behavior. The idea is that profit that particular product generates usually does not only come from itself but also with the help of other products that play a role it its sale. A comparison for ordered quantity for imperfect quality items in frequent items keeping in mind the cross-selling effect is made with and without clustering. Cross-selling is a way of telling customers related products or services. Idea is to get profit from a particular product by influence of other products. It can be done using association rules. It will help find interesting associations in data. Apriori calculation is utilized to create rules which has support and certainty higher than the ones given by user. Frequent itemsets are found based on minimum support and association rules are then generated keeping in mind the minimum confidence.

Clustering here means dividing transactions in clusters of various sizes to such a way that similar transactions have a place in the same cluster and dissimilar transactions are assembled in different clusters. Apriori is applied on each cluster to calculate frequent itemsets. Clustering is done to obtain homogeneous clusters. (Mittal, Pareek, Agarwal, 2015)

4.3 Cluster Analysis and Association Analysis for the same data.

This study had methods that are easier to follow and understand. In some application both cluster analysis and association rule mining are used in order to do well in terms of time and results. In this paper idea of unifying both analyses focus on the database of transactions. At first a data context matrix is generated depending on transactions and items. Closed sets and lattice of closed sets is designed based on the given transactions in data context. In each closed set adding additional information such as support information, contributes to the design of knowledge lattice. The base of mining is called Data Context undertaking here. (Fu, 2008)

Unifying both cluster analysis and association analysis is done based on the following similarities:

Both analysis techniques take into consideration the relationships and links between items or elements in data sets, similarity being the mandatory and most important. Only the description and limits of the link between data sets is different. Frequent patterns indicate one kind of closeness between items of data. Clustering may demonstrate the

relationships in data that will prompt mining the models or rules. Mining closed sets is an important step for both techniques in transactional database. Existing studies have already proven that we can get clusters and frequent patterns from closed sets. (Carpineto, Romano, 1993). It has been noticed that interpretation of clustering and frequent patterns is difficult. Closed sets can help pattern interpretation because closed sets are derived from formal concept analysis. In human understandings the items and objects are grouped by concepts and attributes. Henceforth, concept-based strategies are useful for interpreting clusters and frequent patterns.

In this paper the center is around transactional databases, creating the data context with names of items or transactions. After that comes the progression to mine closed itemsets from transactional databases with formal concept analysis. (Fu, 2008)

Definitions, terminologies and notations that have been used keeping in mind the previous work are as follows:

Data Context is a triple (O, A, R), where O and A are two sets with R being the relation between them.

Set of objects or transactions is represented as O and A is a set of attributes or items. (Fu, 2008). It is typically presented as binary data and values of non-binary attributes can be easily transformed into binary data context by concept scaling.

Formal Concept of (O; A; R) is a pair
(O_1, A_1) and $O_1 \subseteq O$, $A_1 \subseteq A$, $O_1 = A_1'$, $A_1 = O_1'$
We call O_1 extent, and A_1 intent. (Fu, 2008)

Ordered Data Context:

We call data context as an ordered data context in an event that we arrange the things of data context by number of objects of each item from the smallest to the greatest one, and items with same objects are merged as one item. We note ordered data context $(O, A^{\triangleright}, R)$ of the data context (O, A, R). The small arrow on top of A indicates A is in order. (Fu, 2008)

Concept Lattice: All formal concepts with hierarchal order of concepts creates a whole lattice which is called concept lattice.

Closed item set lattice: When C1 and C2 are closed itemsets and C1 is subset of C2, then there is a hierarchal order between C1 and C2. All closed sets with hierarchal order of closed itemsets form a complete lattice called closed itemset lattice.

Knowledge Lattice: An extended version of formal concepts (including e-g, the support of concept) makes a graph of (O, A, R) which provides a clearer picture of associations between the elements. (Fu, 2008).

These are the main terms and notations on the association rules mining part which were considered while generating rules for the given data sets.

From the transactions of database, we can form a data context that contains transactions and items. When generating formal concepts, some extended information such as intent, extent, support and similarity description should be extracted. Closed frequent patterns and clusters can be generated using the same extended concepts. The flow for unifying the cluster analysis and association analysis is following:

Database → data context → formal concepts → ordered data context →knowledge lattice →closed frequent pattern and clusters.

Data Context is the base for mining descriptions used for items in data context should be understandable and meaningful. Sometimes it is necessary to order the data context.

For example, the below figure 7 represents a data context (O, A, R). Let O = {1,2,3,4,5,6,7,8} be the set of objects and A= {a1, a2, a3, a4, a5, a6, a7, a8} be the set of items. The ticks in the table describe the relation R of O and A. (Fu, 2008).

	a1	a2	a3	a4	a5	a6	a7	a8
1	✓	✓					✓	
2	✓	✓					✓	✓
3	✓	✓	✓				✓	✓
4	✓		✓				✓	✓
5	✓	✓		✓		✓		
6	✓	✓	✓	✓		✓		
7	✓		✓	✓	✓			
8	✓		✓	✓		✓		

Figure 7: Example data context (Fu, 2008).

4.4 Theory of Concept Lattices:

A simple illustration of data context is given in Table 5.

TID	*A*	*B*	*C*	*D*	*E*
1	x		x	x	x
2		x			
3	x	x	x		x
4		x			
5	x	x	x		x

Table 5 : Data Context Transactional Data

The respective lattice for table above can be found below in Figure 7.

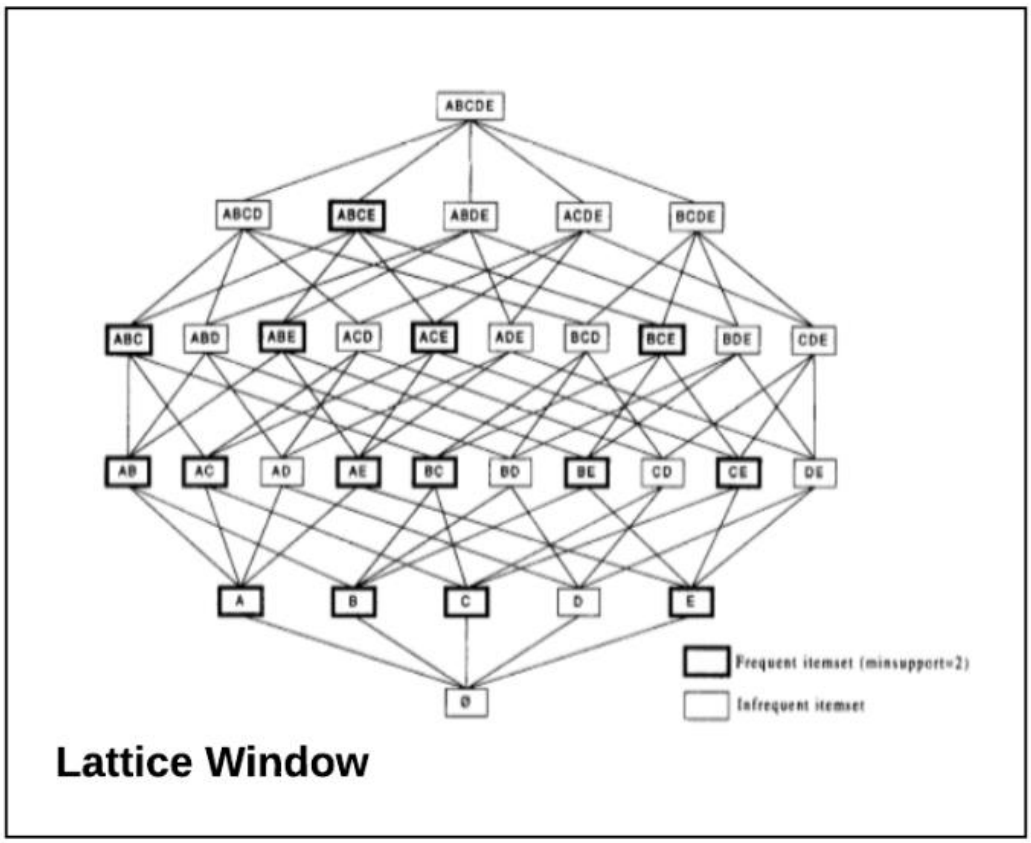

Figure 7: lattice for all objects of Table 5.

Each node is a pair containing subset of items and subset of transactions. More precisely, understanding of concept lattices begins by taking into account binary relationship between a set of objects (O) and set of descriptions (D) which is called a context. Context can be written as (O, D, I). A set of descriptors connected with an object can be well-thought-out of a bit vector. An individual bit can link to a single descriptor and it can be 0 or 1 depending on whether the object has the descriptor. This is the simplest way of scaling attributes and items in a transactional database into to binary data to form easy to store data contexts. So, idea of a context (O,D,I) can be demarcated in terms of pair (X, Y) where X is a set of every object holding every descriptor in Y. Y is a set of all descriptors mutual to every object in X. X and Y can be defined as extent and intent of concept. (Carpineto, Romano, 1993)

4.5 Pre-processing:

There are few tweaks to be done with data in order to get it ready for analysis. Redundancy is one of the major issues in most data sets which can be a hindrance in producing association rules. There are a number of ways in which data can be prepared for association rules. The aim of preprocessing is to reduce a number of objects to be processed by the Apriori procedure or any other association rules algorithm in order to consider only those items that consist the real knowledge of the database. In theory most of the time the preprocessing tasks usually use clustering to get hold of the local knowledge, putting together similar information. Usually this technique leads to items with a low support count to be considered by the association rules algorithm and dominating items will end up splitting into subsets and will become less dominant.

"In (Aggarwal, et al, 2002) a clustering algorithm is presented, known as CLASD (Clustering for Association Discovery), in order to cluster the transactional data available in databases. A similarity measure was proposed to compare the items. The measure is shown in Equation 1, CountT() tells the number of transactions that have the items inside the parenthesis, Tx ∩ Ty will extract the common items between both transactions while Tx ∪ Ty will give the items that at least one transaction has. The algorithm is considered to be hierarchical as it uses bottom up strategy, meaning every transaction starts with an n*n grouping and merges until a parameter k, that represents the number of desired groups, is reached. The merge between the groups is done using the complete linkage clustering method. This approach was capable of finding more frequent itemsets according to results shown, compared to the random partitioning and original data, generating those new rules that cannot be extracted using other methods". (Padua, Renan de et al, 2016).

$$Sim(Tx, Ty) = CountT(Tx \cap Ty)/CountT(Tx \cup Ty)$$

Equation (1)

“In other studies, such as (Plasse, Niang, et al, 2007) a different idea has been presented. It will still cluster the data, but, rather than clustering the transactions, this method will cluster the items in the database. This approach will determine the similarity among items based on the transactions they support, as shown in Equation 2, where Trans(Ix) gives all the transactions that have the item Ix and ∩ and ∪ have the same meaning of Equation 1. Apart from this measure, three other measures to calculate the similarity of the items have been used and some clustering algorithms are applied over the database. The results demonstrated that this approach performs well on sparse data, when general support is extremely low, extracting rules that would be generated if the association rule algorithm was applied over a non-clustered data set”. (Padua, Renan de et. al, 2016).

$$Sim(Ix, Iy) = CountT(T\ rans(Ix) \cap T\ rans(Iy))/CountT(T\ rans(Ix) \cup T\ rans(Iy))$$

Equation (2)

“Both methods had one common purpose that is to find rules that could not be extracted in the original database. However, the analysis that was performed by the authors is merely on the number of new rules (or new itemsets) that were mined. It is important to know that analysis is not made for the results measuring the ratio of newly generated rules and the number of maintained rules, among other facets, which can verify how qualitative preprocessing algorithms are used”. (Padua, Renan de et. al, 2016).

“In (Videla-Cavieres, I. F et al, 2014), a structure called Product Network is used: each vertex of the network represents an item and the edge (link) between them represents the number of transactions they occur together. First of all, the authors created the Product Network and applied a filter, reducing the number of connections among different products. Then, a community detection algorithm was applied to group the products. The authors obtained a great number of communities, around 28+ on each data set they used, each one containing a mean of 7 items per group. The entire discussion is made on the amount of knowledge that will be explored in each group, making the exploration and understanding of each group easier for the user”. (Padua, Renan de et. al, 2016).

5 Proposed Work

To give the simplest idea of how the flow cycle of the analysis will go about, here is a representation in Figure 8

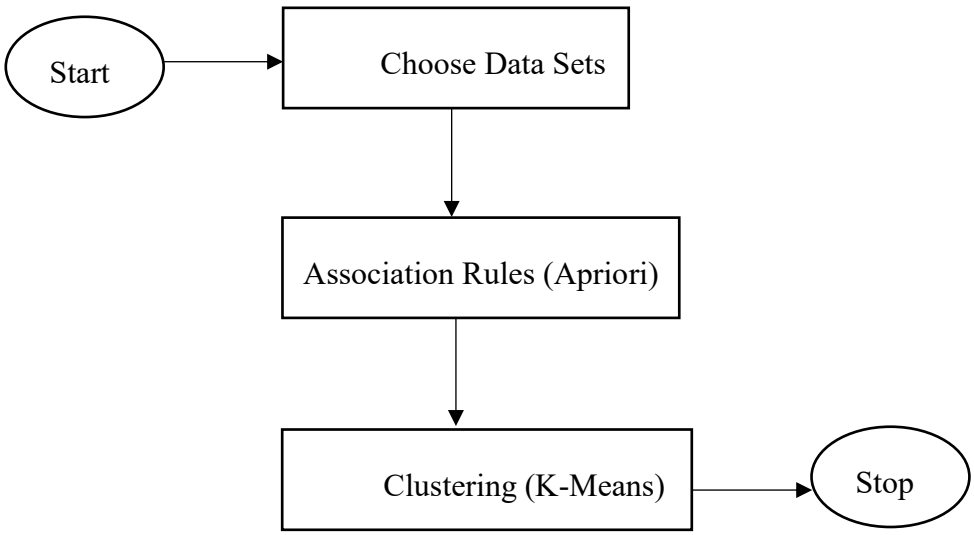

Figure 8 : Basic flow of work

5.1 Finding data set for Association rules

Finding the data set that suits well for association rules mining analysis and clustering is important. As discussed, transactional datasets that can easily be scaled into binary data to make the association rules easily are best suited for the task in hand. As it is always easy to make the data context out of the data set that has been scaled to an equivalent binary version or has flagged yes/no form. Two data sets have been explored up to this point. One is downloaded from Kaggle (site for datasets designed for association analysis) and other from UCI library. Groceries dataset is borrowed from github.com. It comprises of one column and 700 observations. Data is in csv format and easy to manipulate with R. In case of first data set there was not much need to prepare the data as there is only one column and based on that a customer's buying habits were identified. Online retail data set is from UCI library and copy of that dataset is available also on Kaggle. As for results being published in this document, only results for groceries data set are used as results from online retail had similar findings.

5.2 Visualization

These rules are to be plotted to give the graphic look to the analysis, thus making it easy to understand.

In order to count the support of the candidates, an additional pass over the item set is needed by algorithm to execute. The subdivision function is required to govern all the candidate item sets in Ck that are confined in individual transaction t. Function implementation is explained in Section 3.3.

• After counting their support, the algorithm removes all candidate item sets whose support counts are less than minsup.

• The algorithm dismisses when there are no new frequent item sets created, i.e., Fk = ∅. (Tan, Steinbach, Karpatne, Kumar, 2015)

The frequent data set creation part of the Apriori algorithm has two important features. Initial, it is a level-wise algorithm; i.e., it navigates the data set lattice one level at a time, from frequent 1-itemsets to the maximum size of frequent data sets. Second, it services a creates-and-test policy for finding frequent data sets. At each repetition, new candidate data sets are created from the frequent data sets found in the preceding repetition. The support for each candidate is then counted and verified against the minsup threshold. The total number of repetitions needed by the algorithm is kmax + 1, where kmax is the maximum size of the frequent data sets.

	Items
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 b...
6	whole milk,butter,yogurt,rice,abrasive cleaner
7	rolls/buns
8	other vegetables,UHT-milk,rolls/buns,bottled beer,liquo...
9	pot plants
10	whole milk,cereals
11	tropical fruit,other vegetables,white bread,bottled water...

Figure 9 : Grocery Dataset

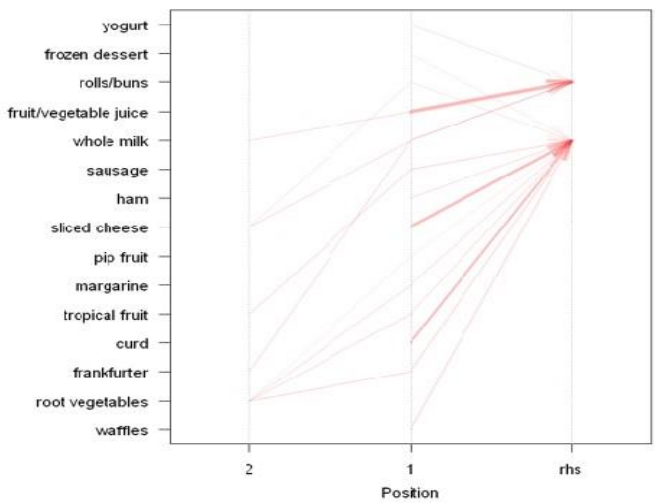

Figure 10: Parallel Coordinate plot for 17 rules

This graph in figure 10 represents association between items bought. From the above visualization it can be deduced that whole milk is bought frequently with other items.

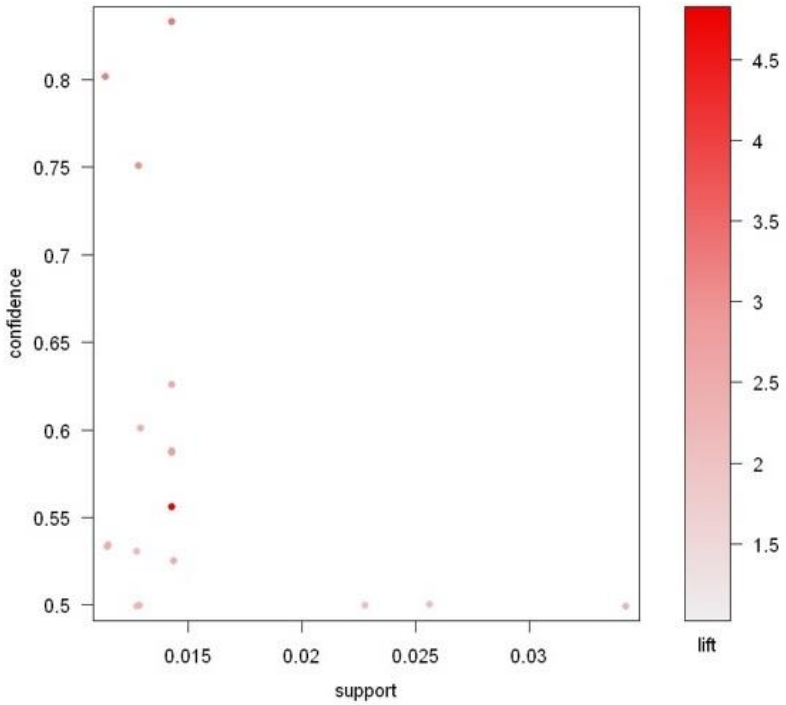

Figure 11 : Scatter Plot of Generated Rules

Figure 11 is the scatter plot between confidence and support of the generated rules. Following are the minimum characteristics.

Least Support: 0.001

Least Confidence: 0.54

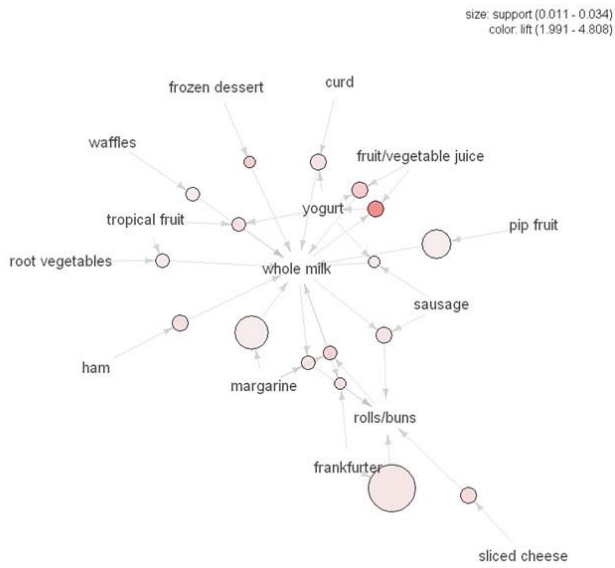

Figure 12 : Network graph between rules.

From the above network graph, it can be seen that mostly all the rules have one thing in common that is ‘whole milk’.

apriori function has been used from the *arules* library in R language. Passing the dataset and required parameters to the function yields us the rules as output.

Then rules were visualized in different type of graphs and bar plots for better understanding of the association between the rules.

5.3 Clustering

Rules generated are then clustered using K-means method. In order to decide K, which is the number of clusters that K-Means has to make we performed elbow analysis. Elbow analysis method is to run K-means clustering on the dataset for a range of values of 1 to *k* and for every value of *k* calculate the sum of squared errors. Then, plot a chart of the SSE for each value of *k*. If the line chart looks like an arm, then the "elbow" on the arm is the value of *k* that is the best. K value from the elbow analysis is used and performed clustering. (Han, Kamber 2012)

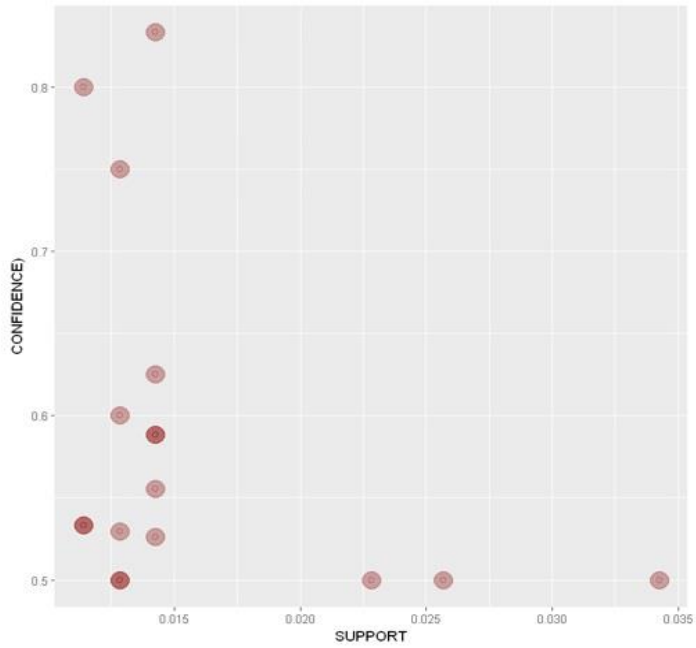

Figure 13 : Rules before running K-Means clustering algorithm

First after determining the number of clusters to make then those columns were selected which were to be used for clustering purpose. K-Mean algorithm was applied on the rules created by Apriori algorithm. The output is the rules divided into separate clusters.

The parameters for K-Mean clustering are support, lift and confidence. After applying the K-Means algorithm we can see that now the points are made into groups. The colours distinguish between the clusters

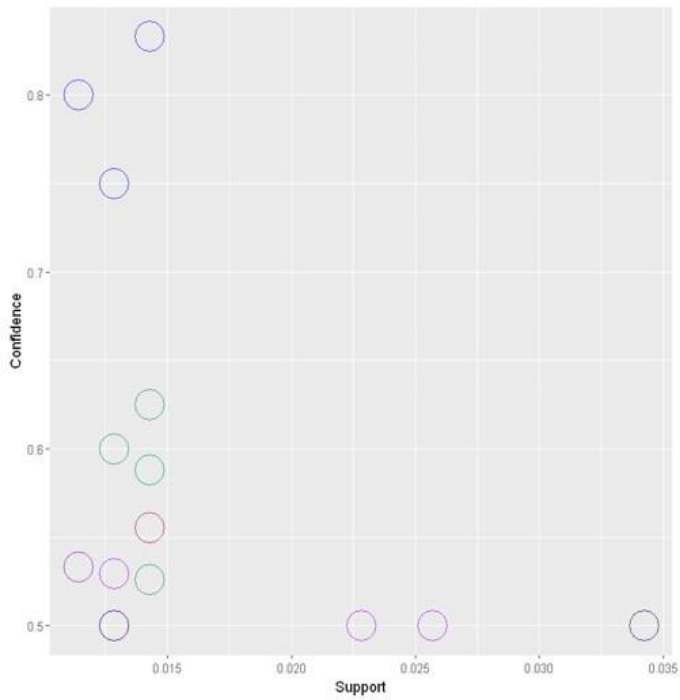

Figure 14 : Rules after running K-Means clustering algorithm

After applying Apriori algorithm now it's time to apply clustering on the rules, but before doing that we need to determine that how many clusters we have to make. By using a function from NbClust library, following results are obtained.

Among all the transactions:

- Best number of clusters is 3 suggested by 1
- Best number of clusters is 4 suggested by 1
- Best number of clusters is 6 suggested by 2*

In deduction, 6 clusters in totals, is the best number.

Table 1: Results after applying clustering on Association Rules

In the figure 16, it can be observed that currently the rules have been scattered in the form of clusters. The initial column expresses the index number of the rules. It can be observed that rule number 7 and 9 fit to cluster number 1 and rule 2, 10, 14, 15 and 17 fit to cluster number 3 and so on.

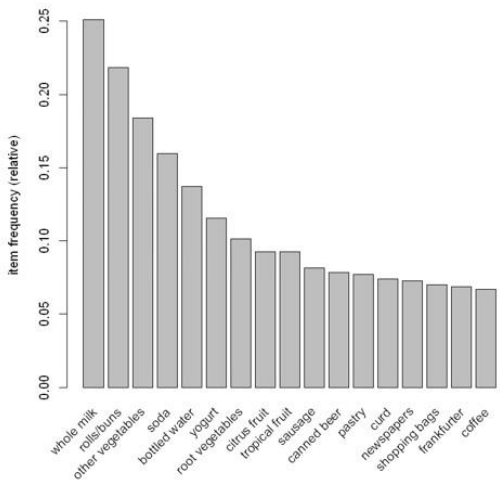

Figure 15 : Frequent Itemset Histogram

	support	confidence	lift	Cluster	rules
7	0.03423680	0.5000000	2.290850	1	{frankfurter} => {rolls/buns}
9	0.01283880	0.5000000	2.290850	1	{margarine,whole milk} => {rolls/buns}
12	0.01426534	0.5555556	4.807956	2	{fruit/vegetable juice,whole milk} => {yogurt}
2	0.01426534	0.6250000	2.489347	3	{ham} => {whole milk}
10	0.01141227	0.5333333	2.443573	3	{frankfurter,whole milk} => {rolls/buns}
14	0.01426534	0.5263158	2.411421	3	{sausage,whole milk} => {rolls/buns}
15	0.01426534	0.5882353	2.342914	3	{curd,yogurt} => {whole milk}
17	0.01283880	0.6000000	2.389773	3	{tropical fruit,yogurt} => {whole milk}
1	0.01141227	0.8000000	3.186364	4	{frozen dessert} => {whole milk}
8	0.01283880	0.7500000	2.987216	4	{margarine,rolls/buns} => {whole milk}
11	0.01426534	0.8333333	3.319129	4	{fruit/vegetable juice,yogurt} => {whole milk}
3	0.01426534	0.5882353	2.695117	5	{sliced cheese} => {rolls/buns}
4	0.01283880	0.5000000	1.991477	6	{waffles} => {whole milk}
5	0.02282454	0.5000000	1.991477	6	{pip fruit} => {whole milk}
6	0.02567760	0.5000000	1.991477	6	{margarine} => {whole milk}

Figure 16 : Clustering on Association Rules

6 Conclusion

There is the breakdown of the steps performed in order to achieve the combination of the two data mining technique. First the dataset was presented in the form of transactions in order to apply the algorithm to a find frequent items set. Then there was a need to find out the most frequent items in the dataset. On those frequent items normal association rules were performed. but the results were not satisfactory. These rules were identified but there was still uncertainty to find any pattern or trend in the rules.

Then there comes the implementation of K-Means clustering to the association rules. Choosing this approach wisely is very important. At first by studying the association mining rules generated on the dataset carefully and deciding to perform K-Mean clustering. Elbow method is used to decide number of clusters. In elbow method the main theme is that increasing clusters can help decrease sum of within-cluster variance of each cluster. This is due to the fact that more clusters help capture finer groups of data objects which have more similar characteristics.

The results were quite satisfactory as now it is quite possible to visualize it better and identify hidden patterns in the association rules. The rules based on their antecedents were perfectly clustered and gave a detailed view of the dataset which better explains the analysis task at hand. Throughout the process of writing and analysis, the main object was to get deep insight into the combination of both association analysis and clustering. Association analysis is very common in market basket type of analysis and the idea of giving input in terms of transactions is very straight-forward and easy to understand for the beginners or anyone who intends to do that sort of analysis. The problem becomes bigger when the rules generated during that process are too numerous and even if rules have been sorted according to the interestingness and relevancy it is still sometimes difficult to find the most relevant ones or the ones that make decision making easy for stakeholders. To solve this kind of limitations, clustering algorithms (K-Means in this study) provide a way to group the rules users are most interested in without scrolling through the results too much.
By using both the processes in parallel, reducing the number of rules for the scope of further analysis is much easier and faster instead of checking interestingness measures such as lift etc, for all the rules. Another approach that is worth exploring and has been under recent considerations by the researchers is to represent datasets on multiple levels depending on their product hierarchy even before the process of extracting, clustering and merging for association rules. In some situations, this can lead to understand

customer buying behaviour more thoroughly. A simple way to look at it, for example, will be, which dairy products is a likely customer to buy next and what can be done to make it easier. Another approach is to apply simultaneous clustering of rows and columns in the transactional data before applying the association rule mining. Row or column blocks created after clustering algorithms are mined separately to generate rules.

7 References:

Charu C. Aggarwal, Cecilia Procopiuc, Philip S. Yu. (2002). Yu. Finding localized associations in market basket data. *IEEE Transactions on Knowledge and Data Engineering* 14.1 51-62.

Claudio Carpineto, Giovanni Romano. 1993. Galois: An order-theoretic approach to conceptual clustering. *Proceedings of ICML*. Vol. 293.

Fukuda Takeshi, Yasukiko Morimoto, Shinichi Morishita, and Takeshi Tokuyama. (1996). Data mining using two-dimensional optimized association rules: Scheme, algorithms, and visualization. *ACM SIGMOD* Record 25, no. 2 13-23.

Fu, Huaiguo. (2008). Cluster analysis and association analysis for the same data. *Proc of the 7th WSEAS International Conference on Artificial intelligence, Knowledge engineering and Databases*. 576-581.

Hartigan, John A., and Manchek A. Wong. (1979). Algorithm AS 136: A k-means clustering algorithm. *Journal of the Royal Statistical Society*. Series C (Applied Statistics) 28.1 100-108.

Han, J.W., Kamber, M. and Pei, J. (2012). *Data Mining Concepts and Techniques*. 3rd Edition, Morgan Kaufmann Publishers, Waltham. 254-255.

Jain, Vikas, (2018) Associative Rule Mining/ Frequent Patterns Mining, accessed 13, March 2019, < http://www.studyml.in/ml/Association-rule-mining.html>.

Kusrini, Kusrini. (2015). Grouping of Retail items by using K-Means clustering. *The Third Information Systems International Conference Procedia Computer Science* 72. 495-502.

Libin Liu, Yee-Kin Ho, Stephen Yau. (2006). Clustering DNA sequence by feature vectors, *Molecular Phylogenetics and Evolution*. (pp 64-69).

Lent, Brian, Arun Swami, and Jennifer Widom. (1997). Clustering association rules. Data Engineering, 1997. *In Proc. 13th International Conference of. IEEE*. (pp. 220-231)

Mahesh Kr. Singh, Zaved Akhtar, Devesh Kr Sharma, (2012). Challenges and Research Issues in Association Rule Mining, *International Journal of Electronics and Computer Science Engineering* 1(2), 767-770.

Marie Plasse, Ndeye Niang, Gilbert Saporta, Alexandre Villeminot, Laurent Leblond. (2007). Combined use of association rules mining and clustering methods to find relevant links between binary rare attributes in a large data set. *Computational Statistics & Data Analysis* 52.1 596-613.

Mandeep Mittal, Sarla Pareek, and Reshu Agarwal. (2015). EOQ estimation for imperfect quality items using association rule mining with clustering. *Decision Science Letters* 4.4. 497-508.

Pang-Ning Tan, Michael Steinbach, Anuj Karpatne, Vipin Kumar (2015), Published by Pearson Education. *Introduction to Data Mining*, Chapter 6. Pages 328-338.

Pavel Berkhin. (2006). A survey of clustering data mining techniques. *Grouping Multidimensional Data* (pp. 25-71). Springer, Berlin, Heidelberg.

Rajak, Akash, and Mahendra Kumar Gupta. (2008). Association rule mining: applications in various areas. *Proc of International Conference on Data Management*, Ghaziabad, India. 5-6.

Padua, Renan de et. al. Pre-processing data sets for association rules using community detection and clustering: a comparative study, (2016) *Proceedings of the XIII National Meeting of Artificial and Computational Intelligence.* Porto Alegre: SBC Publisher, 554-555.

Sanatan Mishra, (2017), Unsupervised Learning and Data Clustering, accessed 13, March 2019, <https://towardsdatascience.com/unsupervised-learning-and-data-clustering-eeecb78b422a>.

Vijay Raghani, (2019) Recommendar Systems using Apriori, accessed 10, March 2019, < https://labs.sogeti.com/recommender-systems-using-apriori/>.

Schaeffer, S. E. (2007). Graph clustering. *Computer Science Review*, 1(1), 27-64.

Strehl, Alexander. (2002). Relationship-based clustering and cluster ensembles for high-dimensional data mining. The University of Texas at Austin. Diss.

Videla-Cavieres, Ivan F., and Sebastian A. Rios. (2014). Extending market basket analysis with graph mining techniques: A real case. *Expert Systems with Applications* 41.4 1928-1936.