Comparative Study of Popular Data Mining Algorithms

T. Venkat Narayana Rao  
Professor, Department of Computer Science and Eng.  
Sreenidhi Institute of Science and Technology(SNIST)  
Yammampet, Hyderabad, TS, India.

Harsh Goel  
Department of Computer Science and Engineering  
Sreenidhi Institute of Science and Technology(SNIST)  
Yammampet, Hyderabad, TS, India.  
Email: harshgoel1999@gmail.com

Abstract: Data Science is an appealing field, in the present world due to advancement of science as there is huge assortment of data which exist in numerous forms. Such data must be handled with care and store safely so that it can be retrieved as per needs. Some of the popular or commonly used algorithms are Apriori algorithm, K Means Clustering, Support Vector machines(SVM) and Association Rule Mining algorithms. This paper focuses on the above mentioned algorithms and a comparison is made in terms of Technique, Time Utilization Software taking real time data examples.

Keywords: Apriori, time, efficiency, Support Vector Machines(SVM), K Means clustering.

I. Introduction

Data science has been a term in the computing field since 1960. It is interdisciplinary, incorporating elements of statistics, data mining, and predictive analysis, and focusing on process and systems that extract knowledge from the data. Basically three methodologies are followed in data science namely classification, regression and similarity matching. A number of algorithms were built on statistical models that are available for data scientists and which algorithm is chosen is based on the goals that have been established prior to implementation. From the emergence of Big Data, data science began to be a fundamental requirement of any organization on working out how to analyze such massive amount of data. Comparative study tells us about various factors involved in algorithms such as system used, time utilization, memory usage, software needed etc[1][2].

II. Popular Data Mining Algorithms

A. Apriori algorithm:

Apriori is an algorithm for items which occur frequently over databases. It was proposed by Agrawal and Srikant in 1994. It starts by identifying ordinary things and extend them to bigger items as long they appear regularly. The common items set that are determined by Apriori can be used to find association rules which have application in domain such as market basket analysis and commercial transactions etc[10]. The Apriori algorithm functions on databases focusing on number of items that consumers buy. Apriori follows "bottom up" approach, where common subsets are extended one item at a time. The algorithm ends when no more successful extension is found. It uses breadth first search and a hash tree structure to calculate item sets efficiently[3][6].

Example of Apriori algorithm:

Suppose we want to find out what all items are commonly bought with other items from the given table. This can be found by following below steps with the aid of Table 1.

Table 1: Items bought

<table>
<thead>
<tr>
<th>Transaction Id</th>
<th>Items bought in Quantity</th>
</tr>
</thead>
<tbody>
<tr>
<td>T1</td>
<td>[QTY1,QTY2,QTY3,QTY4,QTY5, QTY6]</td>
</tr>
<tr>
<td>T2</td>
<td>[QTY7,QTY2,QTY3,QTY4,QTY5, QTY6]</td>
</tr>
<tr>
<td>T3</td>
<td>[QTY1,QTY8,QTY4,QTY5]</td>
</tr>
<tr>
<td>T4</td>
<td>[QTY1,QTY9,QTY10,QTY4,QTY6]</td>
</tr>
<tr>
<td>T5</td>
<td>[QTY10,QTY2,QTY2,QTY4,QTY11, QTY5]</td>
</tr>
</tbody>
</table>

Step 1: From the table 1, calculate number of occurrences of each item. QTY 2 occurs 4 times in total, but, it occurs in 3 transactions.

Table 2: Item Occurrences

<table>
<thead>
<tr>
<th>Items</th>
<th>No of Occurrences</th>
</tr>
</thead>
<tbody>
<tr>
<td>QTY 1</td>
<td>3</td>
</tr>
<tr>
<td>QTY 2</td>
<td>3</td>
</tr>
<tr>
<td>QTY 3</td>
<td>2</td>
</tr>
<tr>
<td>QTY 4</td>
<td>5</td>
</tr>
</tbody>
</table>
Step 2: The most frequently QTY is 3 times. From Table 2 remove all the items which occur less than 3 times and keep only items that are bought more than 3 times as shown in table 3.

Table 3: Item Occurrences which are more than 3

<table>
<thead>
<tr>
<th>Items</th>
<th>No of Occurrences</th>
</tr>
</thead>
<tbody>
<tr>
<td>QTY 1</td>
<td>3</td>
</tr>
<tr>
<td>QTY 2</td>
<td>3</td>
</tr>
<tr>
<td>QTY 4</td>
<td>5</td>
</tr>
<tr>
<td>QTY 5</td>
<td>4</td>
</tr>
<tr>
<td>QTY 6</td>
<td>3</td>
</tr>
</tbody>
</table>

Step 3: Make pairs for items, like QTY1 QTY 3, QTY 1 QTY 4, QTY 1 QTY 5, QTY 1 QTY 6 and then we start with the second item like QTY 2 QTY 4, QTY 2 QTY 5, QTY 2 QTY 6 as shown in table 4.

Table 4: Item Pairs

<table>
<thead>
<tr>
<th>Item Pairs</th>
<th>No of Occurrences</th>
</tr>
</thead>
<tbody>
<tr>
<td>QTY 1 - QTY 2</td>
<td>1</td>
</tr>
<tr>
<td>QTY 1 - QTY 4</td>
<td>3</td>
</tr>
<tr>
<td>QTY 1 - QTY 5</td>
<td>2</td>
</tr>
<tr>
<td>QTY 2 - QTY 6</td>
<td>2</td>
</tr>
<tr>
<td>QTY 2 - QTY 4</td>
<td>3</td>
</tr>
<tr>
<td>QTY 4 - QTY 5</td>
<td>3</td>
</tr>
<tr>
<td>QTY 4 - QTY 6</td>
<td>2</td>
</tr>
<tr>
<td>QTY 5 - QTY 6</td>
<td>1</td>
</tr>
</tbody>
</table>

Step 4: From table 1, Calculate no of times each pair occurs. As shown in table 5 we get support of all the pairs.

Table 5: Occurrences of item pairs

<table>
<thead>
<tr>
<th>Item Pairs</th>
<th>No of Occurrences</th>
</tr>
</thead>
<tbody>
<tr>
<td>QTY 1 QTY 2</td>
<td>1</td>
</tr>
<tr>
<td>QTY 1 QTY 4</td>
<td>3</td>
</tr>
<tr>
<td>QTY 1 QTY 5</td>
<td>2</td>
</tr>
<tr>
<td>QTY 1 QTY 6</td>
<td>2</td>
</tr>
<tr>
<td>QTY 2 QTY 4</td>
<td>3</td>
</tr>
<tr>
<td>QTY 2 QTY 5</td>
<td>3</td>
</tr>
<tr>
<td>QTY 2 QTY 6</td>
<td>2</td>
</tr>
<tr>
<td>QTY 4 QTY 5</td>
<td>4</td>
</tr>
<tr>
<td>QTY 4 QTY 6</td>
<td>3</td>
</tr>
<tr>
<td>QTY 5 QTY 6</td>
<td>2</td>
</tr>
</tbody>
</table>

Step 5: remove all the item pairs with occurrences less than three and we are left with items whose occurrences is more than 3 as shown in table 6.

Table 6: Item pair occurrences which are more than 3

<table>
<thead>
<tr>
<th>Item Pairs</th>
<th>No of Occurrences</th>
</tr>
</thead>
<tbody>
<tr>
<td>QTY 1 QTY 4</td>
<td>3</td>
</tr>
<tr>
<td>QTY 2 QTY 4</td>
<td>3</td>
</tr>
<tr>
<td>QTY 2 QTY 5</td>
<td>3</td>
</tr>
<tr>
<td>QTY 4 QTY 5</td>
<td>4</td>
</tr>
<tr>
<td>QTY 4 QTY 6</td>
<td>3</td>
</tr>
</tbody>
</table>

Pseudo code for Apriori algorithm:

 Algorithm 1 Apriori algorithm
 1: begin
 2:  \[ L_k \leftarrow \text{Frequent}_{k-1} - \text{itemset} \]
 3:  \[ k \leftarrow 2 \]
 4:  \[ \text{while } L_{k-1} \neq \emptyset \text{ do} \]
 5:    \[ \text{Temp } \leftarrow \text{candidateItemSet}(L_{k-1}) \]
 6:    \[ C_k \leftarrow \text{frequencyOfItemSet(Temp)} \]
 7:    \[ L_k \leftarrow \text{compareItemSetWithMinimumSupport}(C_k, \text{minsup}) \]
 8:    \[ k \leftarrow k + 1 \]
 9:  \[ \text{end while} \]
 10: return \( L_k \)
 11: end

Apriori algorithm advantages:
- Uses huge item set belongings.
- Easy to parallelize.
- Easy to apply

Disadvantages:
- Assume operation database is memory occupant.
- Requires a lot of database scan
- Final cluster pattern is dependent on initial.
B. K means clustering Algorithm

Clustering refers to minute group of objects which represents combinations into clusters. Clustering means separating data points into similar classes or clusters. When we have different objects, we put them into groups depending upon similarity.

Clustering Algorithms:

It analyzes data on the source of likeness. It identifies centroid of data points. For useful clustering it evaluates the distance amid each point from the centroid of cluster.

![Figure 1: Process of Clustering](image)

**K-means clustering** is a vector quantization method that comes from signal processing. This algorithm partition n observations into k clusters wherein each observations belongs to the cluster with the closest mean(figure 1).

K-means is a simple unsupervised learning algorithms which follows easy way to sort a given data set through certain number of clusters (k clusters). Primary purpose is to identify k centers, one of each cluster. The centers should be placed in right manner and some place them far from each other. Evaluate each point to a given data set and relate to the nearest center. If no point is left, then first step is completed and early group age is done. Then re-calculate k new centroids so that a fresh binding has to be completed between similar data set points and the adjacent new center so that loop gets formed. As a result of this loop it is noticed that k centers change their place step by step until no more changes exists. It also minimizes an objective function known as squared error function given by:

\[ J(V) = \sum_{i=1}^{c} \sum_{j=1}^{n} c_i \left| |x_j - v_j| \right|^2 \]

here,

\[ |x_j - v_j| \] is the Euclidean distance between \( x_j \) and \( v_j \), \( c_i \) is number of data points in \( c_i \) cluster and \( c \) is number of cluster centers.

**Example of Kmeans clustering:**

Suppose is a company desires to open petrol bunks in highways in a particular state. They want to open petrol outlets in such a way that it covers all highways. The test is to identify location of the petrol outlets so that the whole region is covered. To resolve this we can apply the concept of K means clustering.

**K-means Clustering Procedure:**

As shown in figure 2 we can tell if k is given, then K-means algorithm can be processed in following ways:

- Division of items into k non-empty items
- Resolve cluster centroids of the present partition.
- Transmission of every point to particular cluster
- Calculate distance from each point and allocate it where the distance from centroid is least.
- After re-allotting, identify the centroid of the new clusters generated.

![Figure 2: Step Wise Implementation of K means clustering algorithm](image)

**Pseudo-code for K Means Clustering**

Loop through K times
  current centroid = Randomly generate values for each attribute
  Done = False
  All instances cluster = none
  WHILE not Done
    Total distance = 0
    Done = true
    For each instance
      instance’s previous cluster = instance’s cluster
      measure eucliden distance to each centroid
      find smallest distance and assign instance to that cluster
      if new cluster != previous cluster
        Done = false
      add smallest distance to total distance
      Report total distance
    For each cluster
      loop through attributes
      loop through instances assigned to cluster
      update totals
      calculate average for attribute for cluster = producing new centroid
  END While
K-Means Advantages:
1) K-Means generates results faster than hierarchical clustering, if the k value is small.
2) It creates tighter clusters than hierarchical mainly if they are circular.

K-Means Demerits:
1) Difficult to foresee K-Value.
2) Diverse partitions initially can result in different clusters in final.
3) It do not work well with clusters of diverse dimension and diverse thickness.

C. Support vector machine (SVM):
The support vector machines clustering algorithm created by Hava Siegelmann and Vladimir Vapnik, which sorts unlabeled data and is one of the most extensively use clustering algorithms in industry applications.
This algorithm defines a hyperplane to separate data into two classes. A hyperplane is the line that divides a group but is based on a property or attribute rather than location. This algorithm can help to figure out an underlying separation mechanism between people who would buy a product and those who don’t. In order to maintain the computational load sensible, techniques involve dot products of input data pairs which may compute effortlessly by declaring them with respect to kernel function. The hyperplanes are defined as set of points whose dot product in that space is steady[4][8].

Example:
Imagine there is a machine learning (ML) course is offered at a university. The course instructors have observed that students get the most out the curriculum if they are good at Mathematics or Statistics related courses. Over time, they have recorded the scores of the enrolled students in these subjects. For each student, they have a label depicting their performance “Good” or “Bad”. Now the idea is to determine the relationship between Mathematics and Statistics scores and the performance in the ML course. We could draw a two-dimensional plot, where one axis represents scores in Mathematics, while the other represents scores in Statistics. A student with certain scores is shown as a point on the graph as shown in figure 3.
As shown in figure 3, the color of point green or red represents how he did on the ML course.

Figure 3: Relationship between Mathematics and Statistics scores and the performance in the ML course.

When a student requests for enrollment the instructors would ask her to supply Mathematics and Statistics scores. Based on the data they already have, they would make an informed guess about her performance in the ML course. In this case, finding a line that passes between the red and green clusters, and then determining which side of this line a score tuple lies on, is a good process to determine. For instance, based on the green side or the red side, a good indicator can be set for his/her to ascertain most likely chances to perform in the course.
As shown in figure 4 the line here is the separating boundary or classifier.

Figure 4: Finding of line that passes between the red and green clusters and to tell which side of this line a score tuple lies on.

Now how to identify good and bad classifiers. As shown in figure 5 the first line above seems a bit “skewed”. Near its lower half it seems to run too close to the red cluster, and in its upper half it runs too close to the green cluster. This ensure that the line separates the training data perfectly, but if it sees a test point that is farther out from the clusters, there is a good chance it would get a label wrong. The second line doesn’t have this issue. The second line stays as...
far away as possible from both the clusters while getting the training data separation right. By being right in the middle of the two clusters, it is less “risky” and gives the data distributions for each class some wiggle room and thus generalizes well on test data[9].

Let the plane back be projected to the original two-dimensional space as shown in figure 7

\[
X_1 = x_1^2 \\
X_2 = x_2^2 \\
X_3 = \sqrt{2}x_1 x_2
\]

Figure 5: Identification of good and bad clusters

**Non-linearly Separable Data**

We have seen how Support Vector Machines systematically handle linearly separable data. How does it handle the cases where the data is absolutely not linearly separable? After all, plenty of real-world data falls in this category only. As shown in figure 6, we have only 75% accuracy on the training data the best possible with a line.

And this line passes very close to some of the data. The best accuracy is not appreciable, and to get even there, the line nearly straddles a few points. We start with the dataset in the above figure, and project it into a three-dimensional space where the new coordinates are:

**III. The SVM PSEUDO CODE**

**ACO-SVM Algorithm**

Input: $k, m, q, C, \gamma$, and termination criterion

Output: Optimal value for SVM parameters and classification accuracy

Begin

Initialize $k$ solutions

Call SVM algorithm to evaluate $k$ solutions

$T = \text{Sort} (S_1, ..., S_k)$

while classification accuracy $\neq 100\%$ or number of iteration $\neq 10$

for $i = 1$ to $m$

select $S$ according to its weight

sample selected $S$

store newly generated solutions

Call SVM algorithm to evaluate newly generated solutions

end

$T = \text{Best} (\text{Sort} S_1, ..., S_k + m), k$

end

End

Advantages:
- SVM’s are useful when we don’t have that much knowledge on data.
- Works well with even unstructured and semi structured data like text, Images.
- The kernel trick is important part of SVM. With an appropriate kernel function, we can solve any complex problem.
- It scales relatively well with high dimensional data.
- SVM models have overview in practice, the risk of over fitting is less in SVM.
Disadvantages:
- Selecting a “good” kernel function is not easy.
- Much longer training time for large datasets.

### Table 3: Comparison of 3 algorithms used

<table>
<thead>
<tr>
<th>Parameter(s)</th>
<th>Apriori</th>
<th>K means clustering</th>
<th>SVM</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Technique</strong></td>
<td>Use Apriori property and join and prune property</td>
<td>The k-means clustering algorithm divides a given unknown data set into fixed clusters(k). The fixed number of k clusters are called centroids,</td>
<td>SVM is a supervised machine learning algorithm for classification or regression problems. A technique is used here called kernel trick to convert your data and finds out best boundary between possible outputs</td>
</tr>
<tr>
<td><strong>Memory utilization</strong></td>
<td>Since large number of candidate s are registered so large memory space is needed</td>
<td>Memory use by k-means is essentially the output data size only</td>
<td>The SVM choice function is prolonged into a polynomial form and consolidate into classification function with significant lesser memory footprint and computational cost</td>
</tr>
<tr>
<td><strong>Time needed</strong></td>
<td>Execution time is more</td>
<td>It has been Recently recognized as One of the best Algorithms for Clustering Unsupervised data</td>
<td>A fast and dependable classification algorithm that performs very well with a limited amount of data</td>
</tr>
<tr>
<td><strong>Time complexity</strong></td>
<td>O(2^d)</td>
<td>O (m)</td>
<td>O(n^3)</td>
</tr>
<tr>
<td><strong>Space complexity</strong></td>
<td>O(2^d)</td>
<td>O ((m+k)n)</td>
<td>O(n)</td>
</tr>
<tr>
<td><strong>Data mining Software</strong></td>
<td>Mahout machine learning library</td>
<td>Rapid Miner</td>
<td>LibSVM, WEKA</td>
</tr>
</tbody>
</table>

As shown in table 7 comparison between all 3 algorithms is drawn with various factors[3][4].

IV. Findings and observations

**Apriori Algorithm:**
Apriori algorithm assumes huge data set. For items which occur frequently over databases. It is used extensively in various online e–shopping platforms. It is useful in determining what all items are frequently bought together by customers with respective to an item and places that item accordingly. It is very useful and easy to implement[7].

**K means clustering algorithm:**
It is one among simple unsupervised learning algorithms that solves difficult clustering problems. It categorize given data set through number of clusters which refers to small group of objects. K means algorithm is widely used in order to determine various things like what all important areas are to be covered in order to construct or build any realistic entity[9].

**Support Vector Machines algorithm:**
It is supervised machine learning algorithm which helps to figure out an underlying separation mechanism between people/items who will buy/bought a product and those who won’t. This algorithm defines hyperplane to separate data into two classes. A hyperplane is the line that divides a group but is based on a property or attribute rather than location. The kernel trick is important part of SVM. With an appropriate kernel function, we can solve any complex problem.

**Conclusion**
This paper exhibits the comparisons between Apriori, K means clustering, and Support vector machines algorithms. It has described the functioning of each algorithm and has shown some practical or real time examples for each algorithm. Each algorithm discussed in this paper has its own applications and merits. In today’s world the growing size of data has huge impact on human in terms of taking day to day and business decisions. Thus, the study carried out has a good blend of comparison between three popular algorithms useful for real time implementations.

**References**


