Research Report on Clustering Algorithms: K-Means & AGNES

SELVAGANAPATHY KOODALINGAM, 12665925

KEVIN RAJ, 12661811

Master of Information Technology,

Course Code – CP5605,

Course Title – Advanced Data Mining and Knowledge Discovery

James Cook University of Australia, Singapore

Table of Contents

1. Abstract

In recent years, information industry is giving special attention to the data load that is alarmingly increasing and accumulated day-by-day. The real challenge faced by these industries lies in transforming this huge set of data into valuable information that will be of prime help in decision making process. To bridge the huge distance or gap between data and information, **Data Mining** is employed as a powerful tool that turns this bulky raw data into meaningful information. Having many numbers of data mining technologies, in this paper our focus is restricted on **two clustering methods** - K-Means and AGNES (Agglomerative Nesting). This paper gives an overview of how both the algorithms work on two different datasets and how the individual algorithms behave on the same dataset. Apart from comparison between the algorithms, the java implemented results of these algorithms is compared with the widely accepted data mining software tool, WEKA. Also we propose ways for improving the algorithms in such a way to obtain the best results and consequently the best useful information is obtained as a result of clustering process.

2. Introduction

As said earlier, data is getting added in huge loads each day making the size of the database to range in terabytes. The art of data mining lies in extracting the useful hidden information that is of more value to the organization and also representing the useful information in user interpretable manner. Data Mining is not a process that emerged all of a sudden. It is the result of evolution in the information processing. While the evolution led to many data mining algorithms, we are going to analyze one of the widely used method called as, ***clustering***.

Clustering works on the base principle of grouping similar items that resemble each other most closely. It is an unsupervised learning technique which does not need any labeling of the dataset or training dataset. The objective of clustering methods is to increase the intra-cluster similarity and minimize the inter-class similarity. Viewing the clustering in a broad picture as a tree is well rooted with many branches. Further narrowing down our focus, research is done on two of its sub-branches – K-Means and AGNES (Agglomerative Nesting).

K-Means algorithm

It falls under the category of **Partitioning Methods**. It is also one of the most widely used and well-known portioning method. It operates on a dataset for which the number of clusters to be formed needs to be given as an input. So, K-Means accepts two inputs –

i. The number of clusters to be formed.
ii. Dataset containing features with many instances.

The clusters that are formed will have

- More intra-cluster similarity.
- Less inter-cluster similarity.

Similarity is measured in terms of the mean value of the objects or instances that represent a cluster which can be thought of as the cluster's centroid or point of attraction that holds the similar data towards it and leaving out dissimilar objects/instances.

Agglomerative Nesting algorithm

This algorithm, popularly known as **AGNES** falls under the category of **Hierarchical clustering** approach. Agnes proceeds by a series of aggregation. In this method each instance in the given input represents a cluster by itself at the first level. When moving up the hierarchy to the next level we can see the most similar instances/objects are merged into one cluster. This series of aggregation proceeds till we have a single cluster which will be the top level. The level with the best clustering is taken as the best result.

3. Related work

Before delving into the improvements that can be made to these methods, in this section we describe a more detailed description of how the existing algorithm works?

K-Means

The algorithm works as follows.

- Let the number of clusters to be formed be represented as ***k***
- From the dataset, randomly *k* instances are chosen to be the initial cluster centroids.
- The remaining instances are assigned to each of the cluster based on the similarity of the object with the cluster centroids.
- Similarity measure can be computed as a distance function, which can be
    - Euclidean distance
    - Manhattan distance
    - Minkowski distance

- An instance/object is assigned to a cluster to which it has the minimum distance or in other words minimum dissimilarity.
- Now the new centroid is computed based on the newly assigned objects and again dissimilarity is measured for each instance with the newly computed centroids.
- The re-distribution of instances stops when the cluster centroid does not change.
- Also we can say that algorithm converges when the cluster centroid does not change.
- The convergence criterion is square-error-criterion.

$$E = \sum_{i=1}^{k} \sum_{p \in C_i} |p - m_i|^2$$

  - E is the sum of the square error for all objects in the data set;
  - p is the point in space representing a given object;
  - m_i is the mean of cluster C_i (both p and m_i are multidimensional).
  - For each object in each cluster, the distance from the object to its cluster center is squared, and the distances are summed.
  - This criterion tries to make the resulting k clusters as compact and as separate as possible
- In our research, we have used the Euclidean distance in our implementation to evaluate the dissimilarity measure between instances and cluster centroids. The distance metric is given by the below equation, where p and q represent two instances.

$$d(p, q) = \sqrt{\sum_{i=1}^{n} (p_i - q_i)^2}$$

Agglomerative Nesting

Working description of AGNES is as follows

- Uses the Single-Linkage method to determine the distance between two clusters.
- Initially, AGNES places each instance of the dataset as a single cluster.
  - Say if there are 10 instances in the dataset, at the initial stage, all 10 instances are 10 clusters.
- Next the clusters with minimum distance (i.e. Single Linkage) are determined and merged as single cluster.
- This merging proceeds till all the instances are placed in a single cluster.
- In our research, we have made use of the Euclidean distance to compute the dissimilarity between two clusters.

Both AGNES and K-Means are already implemented in the WEKA software. The next section involves a comparison of our implemented version of these algorithms with the WEKA version.

4. Dataset

To compare the efficiency of our implemented versions of K-Means & AGNES against the WEKA versions we have taken two sample datasets. Their descriptions are as follows.

Dataset 1 – Facial Data

Facial Expression (Palsy) Data is a dataset that contains instances of different features of facial expression.

Properties:

- No of instances: 66
- No of features/attributes: 66
- Data type: All attributes are numerical
- Has two sets of classes: +1 and -1
- Instances in class +1 : 22
- Instances in class -1 : 44

Dataset 2 – Diabetes Data

Diabetes Data is a dataset that contains instances of different features that are useful to determine whether a patient is tested positive or negative of diabetes test.

Properties:

- No of instances: 768
- No of features/attributes: 8
- Data type: All attributes are numerical
- Two sets of classes: tested_positive and tested_negative
- Instances in class tested_positive : 268
- Instances in class tested_negative : 500

5. Implementation Results

In this research paper we use two types of Data sets – Facial Expression (Palsy) data and Diabetes data.

- Both the datasets are given as input to both the algorithms and the result for each dataset is with one algorithm compared with the corresponding result of the other algorithm.
- Also the same dataset is run with the WEKA software to compare the efficiency of our implemented version.

Table 1.1 Result of K-Means on Palsy data-Our version

Sl.No	Seed value	Iterations	SSE	Time Taken in seconds	Clusters Formed			
					# of clusters	Cluster #	Class Label	# of instances
1	10	3	4105.066	0.29	2	1 - 73.0%	-1	30
							+1	18
						2 - 27.0%	-1	14
							+1	4
2	25	3	4035.174	0.31	2	1 - 56.0%	-1	21
							+1	16
						2 - 44.0%	-1	23
							+1	6
3	35	3	3952.055	0.35	2	1 - 59.0%	-1	30
							+1	09
						2 - 41.0%	-1	14
							+1	13
4	115	3	3727.365	0.29	2	1 - 26.0%	-1	7
							+1	10
						2 - 74.0%	-1	37
							+1	12
5	230	3	3625.418	0.28	2	1 - 82.0%	-1	40
							+1	14
						2 - 18.0%	-1	4
							+1	8
6	715	3	3625.418	0.28	2	1 - 82.0%	-1	40
							+1	14
						2 - 18.0%	-1	4
							+1	8

Table 1.2 Result of K-Means on Diabetes data-Our version

Sl.No	Seed value	Iterations	SSE	Time Taken in seconds	Clusters Formed			
					# of clusters	Cluster #	Class Label	# of instances
1	10	14	1.1372374481E7	1.13	2	1 - 64.0%	tested_positive	122
							tested_negative	371
						2 - 36.0%	tested_positive	146
							tested_negative	129
2	25	14	1.1372374481E7	1.13	2	1 - 64.0%	tested_positive	122
							tested_negative	371
						2 - 36.0%	tested_positive	146
							tested_negative	129
3	35	9	1.0534472353E7	0.82	2	1 - 42.0%	tested_positive	186
							tested_negative	136
						2 - 58.0%	tested_positive	82
							tested_negative	136
4	115	13	9896081.665	1.06	2	1 - 56.0%	tested_positive	76
							tested_negative	355
						2 - 44.0%	tested_positive	192
							tested_negative	145
5	230	20	8110684.037	1.32	2	1 - 60.0%	tested_positive	110
							tested_negative	352
						2 - 40.0%	tested_positive	158
							tested_negative	148
6	735	9	8054465.523	0.79	2	1 - 39.0%	tested_positive	155
							tested_negative	147
						2 - 61.0%	tested_positive	113
							tested_negative	353

Table 1.3 Result of AGNES on K-Means data-Our version

Sl.No	Time Taken In Seconds	Level #	Clusters Formed		
			# of clusters	Class Label	# of instances
6	0.77	766	2	+1	21
				-1	44
				+1	1
				-1	0

Table 1.4 Result of AGNES on Diabetes data-Our version

Sl.No	Time Taken In Seconds	Level #	Clusters Formed		
			# of clusters	Class Label	# of instances
6	7.082	766	2	tested_positive	267
				tested_negative	500
				tested_positive	1
				tested_negative	0

6. Weka Results

Weka is software tool that in-houses many data mining algorithms in it.To have a better comparative study we would like to compare our implementation results with the widely accepted standard data mining software tool WEKA's results.

So the dataset is given as input to the WEKA implementation and results are observed.Below depicted tabular results are weka results for the dataset which we taken in our research

Table 1.5 Result of K-Means on Palsy data-Weka version

Sl.No	Seed value	Iterations	SSE	Time Taken in seconds	Clusters Formed			
					# of clusters	Cluster #	Class Label	# of instances
1	10	2	147.997	0.06	2	1 – 36%	+1	8
							-1	16
						2 – 64%	+1	14
							-1	28

2	25	2	147.61	0.03	2	1 – 48%	+1	7
							-1	25
						2 – 52%	+1	15
							-1	19
3	35	2	147.19	0.02	2	1 – 85%	+1	16
							-1	40
						2 – 15%	+1	6
							-1	4
4	115	3	146.97	0.03	2	1 – 29%	+1	11
							-1	8
						2 – 71%	+1	11
							-1	36
5	230	3	147.29	0.01	2	1 – 33%	+1	11
							-1	11
						2 – 67%	+1	11
							-1	33
6	715	3	147.22	0.01	2	1 – 47%	+1	15
							-1	16
						2 – 53%	+1	7
							-1	28

Table 1.6 Result of K-Means on Diabetes data-Weka version

Sl.No	Seed value	Iterations	SSE	Time Taken in seconds	Clusters Formed			
					# of clusters	Cluster #	Class Label	# of instances
1	10	7	121.25	0.19	2	1 – 67.0%	tested_positive	135
							tested_negative	380
						2 - 33.0%	tested_positive	133

							tested_negative	120
2	25	14	121.25	0.05	2	1 - 67.0%	tested_positive	136
							tested_negative	381
						2 - 33.0%	tested_positive	132
							tested_negative	119
3	35	11	121.25	0.04	2	1 – 33%	tested_positive	133
							tested_negative	120
						2 - 67%	tested_positive	135
							tested_negative	380
4	115	15	121.25	0.05	2	1 - 67.0%	tested_positive	135
							tested_negative	380
						2 - 33.0%	tested_positive	133
							tested_negative	120
5	230	15	121.25	0.05	2	1 - 67.0%	tested_positive	135
							tested_negative	380
						2 - 33.0%	tested_positive	133
							tested_negative	120
6	735	9	121.25	0.04	2	1 – 33%	tested_positive	133
							tested_negative	120
						2 - 67%	tested_positive	135
							tested_negative	380

Table 1.7 Result of AGNES on Palsy data-Weka version

Sl.No	Time Taken In Seconds	Clusters Formed		
		# of clusters	Class Label	# of instances
6	0.4	2	+1	21
			-1	44
			+1	1
			-1	0

Table 1.8 Result of AGNES on Diabetes data-Weka version

Sl.No	Time Taken In Seconds	Clusters Formed		
		# of clusters	Class Label	# of instances
6	7.39	2	tested_positive	267
			tested_negative	500
			tested_positive	1
			tested_negative	0

7. Comparison

Having the results of our implementation, now we can have a comparative study of both the algorithms and also against the WEKA version.

K-Means vs. AGNES

1. **Palsy Dataset**
   - From Table 1.1 and Table 1.3 we can see the results of K-Means and AGNES on palsy data.
   - We can see that K-Means resulted in 2 cluster formation with 23 incorrect instances while AGNES resulted in 2 clusters with 45 incorrect instances.
   - Incorrect Instances:
     - K-Means – 23/66
     - AGNES - 45/66

2. **Diabetes Dataset**
   - From Table 1.2 and Table 1.4 we can see the results of K-Means and AGNES on diabetes data.
   - We can see that K-Means resulted in 2 cluster formation with 260 incorrect instances while AGNES resulted in 2 clusters with 267 incorrect instances.
   - Incorrect instances:
     - K-Means – 260/768
     - AGNES – 267/768

From this comparison, we can state that **K-Means performs well** with these two datasets compared to the AGNES performance.

Our Implementation vs. Weka

1. K-Means

a) Palsy Dataset

	Seed value	Iterations	Time Taken in seconds	Clusters Formed			
				# of clusters	Cluster #	Class Label	# of instances
Our Implentation	715	3	0.79	2	1 - 82.0%	-1	40
						+1	14
					2 - 18.0%	-1	4
						+1	8
WEKA	715	3	0.01	2	1 – 47%	+1	15
						-1	16
					2 – 53%	+1	7
						-1	28

Comparing both the implementation versions, we have that for the same seed value; our implementation gave 18 incorrect instances while the WEKA produced clusters have 23 incorrect instances.

b) Diabetes Dataset

	Seed value	Iterations	Time Taken in seconds	Clusters Formed			
				# of clusters	Cluster #	Class Label	# of instances
Our Implentation	715	9	0.28	2	1 - 39.0%	tested_positive	155
						tested_negative	147
					2 - 61.0%	tested_positive	113
						tested_negative	353
WEKA	735	9	0.04	2	1 – 33%	tested_positive	133
						tested_negative	120
					2 - 67%	tested_positive	135
						tested_negative	380

From the above table, we see that our version has 260 incorrect instances and WEKA version has 255 instances.

Overall we can see that our version produced results that are similar to the results got through the WEKA standards.

2. AGNES

a) Palsy Dataset

- From Table 1.3 and Table 1.7, we can clearly infer that both the versions produced the same results.
- Both produced two clusters with 45 incorrect instances.

a) Diabetes Dataset

- From Table 1.4 and Table 1.8, we can clearly infer that both the versions produced the same results.
- Both the versions produced two clusters with 267 incorrect instances.

8. Discussion

- **K-Means**
  - As earlier stated, K-Means works by pulling instances towards the cluster centroids.
  - K-Means is only efficient in finding clusters of only convex shape.
  - Not suitable for identifying clusters with non-convex shape or others.
  - More sensitive to outlier and noise as it can have a very great impact on the centroid calculation.
  - K-Means effectiveness depends on the initial value selected as the seeds.
  - Very often, the cluster result gets stuck at local optimum and is based on the initial seed value.
  - The initial value should be chosen in such a way that it attains global optimum.

- **AGNES**
  - AGNES suffers a difficulty regarding the selection of merge points.
  - Once a decision to merge two points is done, it is impossible to undo the merge done in the previous step.
  - The new step operates on the new clusters merged in the previous step.
  - If a previous merge decision is not chosen well, then it will lead to a cluster that is of low-quality.

- Also the scalability of the algorithm is not that good, as it involves the examination and evaluation of large number of clusters/instances in each step.

9. Improvements

After a detailed analysis on the algorithms, we have proposed the following improvements that can be made on the algorithms.

- **K-Means**
  - **Normalization**
    - K-Means is highly sensitive to outliers.
    - Outliers can have a huge impact on the effect of cluster centroids.
    - It can make the centroids to move away and pull data from other clusters or form entirely a different cluster.
    - Objective here is to make the outliers remain detectable.
    - It has been experimentally proved that normalizing the dataset can reduce the impact of outliers and it is implemented in our version.
  - **Run the algorithm N times with different seed values**
    - K-Means often stops at local optimum value.
    - The square error criterion converges at local optimal value.
    - To reach the global optimum value, the initial value should be chosen accordingly.
    - Run the algorithm N number of times with different initial cluster centroids and choose the best cluster results.
  - **Simulated Annealing**
    - Choosing the best result by running the result "N" number of times may not guarantee that the square error criterion converges to the global optimum.
    - We refer here a method called Simulated Annealing.
    - Simulated Annealing is a concept used in metallurgy
      1. Heating up of metals such that it shoots up and down.
    - Choose a set of initial seed values
    - Shoot up and down the initial values and proceed with the step to the successive values.
    - Run the algorithm with resulting set of seed values.

- This ensures that there is a high probability of determining the initial seed value that helps in attaining the global optimum
- Another advantage is that it is a heuristic approach rather than an exhaustive search of initial cluster centroids.

- **Employing Principal Component Analysis**
  - It is always important that our algorithm should be scalable when dimensionality increases. Let us assume we have a dataset of dimension 'N'
  - Proposal includes the following steps:

    1. ***Decrease the dimensionality/features of the dataset by the following steps:***
       i. Use principal component analysis do reduce the dimensionality which yields in a dimension d such that d<N.
       ii. Principal components are determined by employing the covariance matrix method.
       iii. Select the foremost principal component as the principal axis for partitioning and arrange them in increasing order.
    2. ***Determine the initial centroids***
       i. Divide the set got through the above step into k steps, where k being the number of clusters to be determined.
       ii. Find the median of the each subset.
       iii. The median will be the initial cluster centroids.

       (or)

       i. The above method of ***simulated annealing*** can be used in combination to determine the initial seed values.
    3. ***Perform the clustering as usual K-Means algorithm***
       i. Using iterative relocation of instances by its similarity with cluster centroids and updating the cluster centroids with the newly assigned instances to the cluster in each iteration.
       ii. Similarity is measured in terms of distance function such as using Euclidean distance.
       iii. Iteration stops when the square error criterion converges to an minimum value.

- **Agnes**
  - **Hybrid Clustering**
    - It is better to run AGNES in combination with other clustering algorithms so that it yields better results.
    - Also normalizing the input dataset can improve the efficiency of the algorithm.

10. Conclusion

In this research paper we have analyzed and shown the implementation results of two clustering algorithms: K-Means and AGNES. The algorithms are compared against each other by running it on tow datasets. They are compared in terms of its efficiency and effectiveness. Also the implemented versions of algorithms are compared with a standardized data mining software too, WEKA and the results are similar to those of WEKA standards. Also a brief idea on improvisation of the algorithm is suggested in this research report. Implementation of referred simulated annealing on K-Means is not done and is on progress. The comparative study also showed that K-means performs and scales well with large databases and also yields commendable results.

11. References

[1] Facial Expression Dataset - http://kdd.newscope.org/

[2] Diabetes Dataset - http://www.cs.waikato.ac.nz/ml/weka/

[3] Data Mining Concepts and Techniques, Jiawei Han and Micheline Kamber.

[4] "Performance analysis of k-means with different initialization methods for high dimensional data", Tajunisha, Sri Ramakrishna College of Arts and Science and Saravanan Dr.N.G.P. Institute of Technology, India

[5] Abstract of "K-Means Optimization Algorithm for solving clustering problem", Jinxin Dong; Minyong Qi; College of Computer Science, Liaocheng University, Liaocheng.

[6] "Simulated Annealing For Selecting Optimal Initial Seeds In The K-Means Algorithm", G.Phanendra Babu And M.Narasimha Murty, Department Of Computer Science and Automation, Indian Institute of Science, Bangalore 560 012

[7] "Consensus-Based K-Means algorithm for Distributed Learning Using wireless sensor networks", Pedro A.Forero, Alfonso Cano and Georgios B.Giannikis; Department of ECE, University of Minnesota, Minneapolis, MN 55455

[8] K-Means clustering via Principal Component Analysis, Chris Deng, Xiaofeng He; Computational Research Division, Lawrence Berkeley National Laboratory, Berkeley