Comparing the Efficiency of Two Clustering Techniques

A Case-Study Using Tweets

Submitted as part of Masters of Science program requirement
At University of Maryland
By

Srividya Ramaswamy

Introduction:

Clustering is the method of analyzing and organizing data such that data which share similar characteristics are grouped together. Clustering is used in various fields including Data Mining [19], [20], [21] and Machine Learning [28], [29], [30]. Much research has been done in the past to efficiently cluster data. Various algorithms and methods have been devised for the same. Hierarchical clustering [22], [23], partitional clustering [24], nearest neighbor clustering [26], [27], fuzzy clustering [25] are some popular techniques used for clustering [6]. This paper conducts a survey of two different methods of clustering. While both the algorithms are basically hierarchical in nature, the difference comes in their implementation. The first algorithm incorporates techniques from association rule problems [16]. On the other hand, the second one incorporates techniques from partitional clustering methods [31], [32]. This paper provides a summary of the implementation of both the algorithms and does a comparison of their behavior.

Clustering:

Clustering is a multi-step process [6]. Given two elements, we have to first define what we mean when we say “they are similar”. This would enable us to decide if these elements should belong to the same cluster or not. Next, we have to define a “distance” function which helps us identify how close any two elements are to each other. Thus two elements whose distance from each other is very small (say less than some alpha > 0) would be put in the same cluster. A common distance function used for numeric data is the Euclidean distance [34]. For example, in a 1-dimensional space, the Euclidean distance between two points x and y is |x-y|. One of the common distance measures for text is Levelshtein distance [35]. The main difference between the algorithms discussed in this paper is the choice of distance function. We will discuss this in detail in later sections.

The final step is to perform the actual clustering process by using any of the various existing algorithms. Numerous techniques have been devised to efficiently cluster data. The algorithms discussed in this paper are Hierarchical in nature. Hierarchical clustering algorithms fall into one of the following two categories - We could start with classifying everything as a single large cluster and then systematically break it down into smaller clusters based on the distance [23], [33]. Alternatively we could start with having many clusters and then progressively merge them based on their distances [23], [33]. The method used varies on the application and the type of data. Both the algorithms used here are of the second kind. We start with clusters having one element each and then progressively build larger clusters.

Twitter:

Twitter is an information network that lets you know “what is happening now”. In other words, Twitter is a micro-blogging service which allows its users to read and write messages which are called *Tweets*. Every minute thousands of people write about what’s new in their lives in the form of tweets. Twitter recently changed its motto from "What are you doing?" to "What's happening?” and this certainly increased its application in various other fields ranging from traffic updates to emergency alerts [3], [2]. Obviously since the volume of data flowing into Twitter is huge it could be used put to use in various interesting and important research projects including sentiment analysis [7]. Data Wrangling uses Twitter data for their ‘R-project’ [40]. Jennifer Golbeck et al have done some research on the usage of Twitter by the U.S. Congress [42]. The Enterprise Data World Conference; Business Intelligence, Data Mining & Machine Learning; The Database Journal are some of the data miners of Twitter [39]. Realizing the value of information stored in Twitter in the form of Tweets, the Library of Congress is archiving now all the Tweets [36]. However the problem with Twitter is that the tweets are neither categorized nor prioritized based on user preference. Whenever a search is run a whole chunk of data is thrown at us with no classification which makes it difficult for us to derive anything useful from it. This is where clustering fits in. The algorithms presented here help group the tweets into clusters which help the user parse the results better. When a search keyword is entered, only clusters containing that keyword need to be returned.

Related Work:

One common complaint that users of Twitter clients have is its lack of clustering [8]. A few attempts have been made in the recent past to extract data from Twitter and use it for data mining purposes [9]. Some have tried to organize the tweets in a more readable format [10]. One such interesting site is *www.TopicScoop.com* which extracts all the relevant tweets based on a search keyword and separates them into different categories. It also indicates which topics are popular amongst other users. In addition we are provided with the option of filtering data based on topics of interest [10].

Various studies have been conducted using Twitter data. Earlier in June 2009, Heil and Piskorsi [11] tried to analyze the usage of Twitter and came up an estimate of the number of tweets an average user makes every day. They also tried to analyze the number of male and female followers for men and women.

There are various Twitter research tools which are available including TweetStats, Trendrr, Tweetmeme, Twitturly, WeFollow, Twellow, and Xefer [12]. TweetStats gives an account of your usage of Twitter. Trendrr is a very interesting tool which gives a popularity trend chart based on a keyword search. As the name stands, WeFollow and Twellow keep track of Twitter users and categorize them based on various categories.

One notable project on Twitter is that by Kalafatis [9] who recently tried to analyze the thoughts of thousands of Twitter users. He conducted an interesting experiment where he extracted all the posts/tweets which contained the phrase “I don’t want to” and after some processing came up with a list of activities that Twitter users didn’t want to engage in. Obviously the activity that topped the list was going to school/work the next day! Of course, Kalafatis had to do a lot of processing before he could come up this information. He first had to analyze the data, discarded useless information, group relevant information them into clusters and then finally assess the output. However, he gave a new dimension to the usage of Twitter data. Without any doubt more research work in this area can be expected to be performed in the future considering the importance Twitter data is gaining. The amount of useful information hidden beneath those cluttered tweets is innumerous and invaluable.

Goal of this paper:

The goal of this paper is to see how the choice of distance function affects the behavior of hierarchical clustering algorithms. The original algorithm is from a paper by Ward [22]. The algorithm by Ward aims at grouping a set into mutually exclusive subsets such that the elements in a subset are very similar to each other. Johnson [23] performed further research about Hierarchical Clustering Schemes and describes the above algorithm as follows:

Step 1: We are initially given k elements and consider each of them to be in k different clusters.
Step 2: Consider all the (k choose 2) different pairs of clusters and merge the two clusters whose distance from each other is the least.
Step 3: Repeat the above step until we get one large cluster.

The above algorithm is certainly the starting point for the algorithm used in this study. However it cannot be used in its entirety as we are not trying to form mutually exclusive subsets here. The purpose of clustering tweets is to form clusters of words such that words that appear in a cluster appear in as many same tweets as possible. For example, the words “Spain”, “Barcelona”, “Picasso” might be common words that appear in tweets related to Spain tourism. Similarly, “Spain”, “Soccer” could be common words that appear in tweets regarding soccer world cup. Thus, we would like to have a cluster containing the words “Spain”, “Barcelona”, and “Picasso” and another cluster containing the words “Spain” and “Soccer”. Even though “Spain” is a common word between the two clusters, they cannot be combined. Also, the fact that “Spain” and “Soccer” appear in the same cluster doesn’t necessarily mean that every cluster that contains “Spain” contains “Soccer” as well. It just means that a reasonable amount of tweets containing “Spain” contain “Soccer”. Thus in the context of Twitter, we consider two words to be closer if they appear in more tweets together. We do not want clusters of large size, especially considering that each tweet itself has only 140 characters. We are looking for meaningful clusters which are not too big in size. Thus we fix the maximum size (MAX_LENGTH) of clusters we are allowed to form. For our purposes we modify Ward’s algorithm as follows:

Step 1: We are initially given k elements and consider each of them to be in k different clusters.
Step 2: For each element ‘e’ (among the k given elements) repeat Steps 3 and 4
Step 3: For each cluster ‘c’ repeat Step 4
Step 4: Determine if the element ‘e’ is close enough to the cluster ‘c’. If so, add ‘e’ to ‘c’
Step 5: Repeat Steps 2, 3, and 4 until we have reached the maximum size of the clusters desired or until no more new clusters can be formed

As mentioned earlier, two words are “close” when they appear in more tweets together. In Step 4 above we have to determine how close an element is to a cluster. For this we have to define the notion of the distance of an element (word) from a cluster. Algorithm 1 uses concepts from association rule problems [41], [16] – Support and Confidence. We look at the proportion of tweets containing the cluster ‘c’ that contain the element ‘e’ as well. If the amount is reasonable, then we conclude that they are close enough and combine them. We talk more about this in later sections.

Algorithm 2 is a slight modification of Algorithm 1. In Step 4 above, Algorithm 2 looks at the average distance of the element ‘e’ from each element of the cluster ‘c’. If the average distance is small enough, then we add the word/element ‘e’ to the cluster ‘c’ to form a larger cluster.

Finally, we compare the effect of each of these distance functions on the clustering algorithm. We calculate the percentage of 1-Clusters, 2-Clusters,…, 5-Clusters formed and see if they vary for both the algorithms for different values of Confidence level and Support level.

Preprocessing:

Thousands of tweets are added every day to Twitter and not all the information is useful. Thus it might be helpful to come up with heuristics to efficiently and quickly cluster data.

(a) Certain words do not contribute much in forming clusters. These include some of the most frequently occurring terms in Literature. Such words are called “*Stop words*” [43]. These words include: the, to, an, a, and, for, if, then, and many more. For purposes of this study these words are called “**insignificant words**”. Insignificant words are removed from the tweets before forming clusters.

(b) Twitter has a restriction of 140 characters. Considering that the average length of an English word is five letters [14], the maximum number of words in a tweet is expected to be 28. However, a recent survey indicated that the average number of words in a tweet is around 15 [15]. Ignoring the insignificant words, we could assume that the number of non-insignificant words would be around 10. Thus when 7 or more words match in two tweets (after removing the insignificant words), then we assume that they are retweets and merge them together.

Definitions:

Before proceeding any further let us define a few terms and clarify some notations:

Insignificant Word – *Stop words* that do not contribute much to clusters (example: ‘of’, ‘in’, etc.)

Token – Any word that appears in a tweet and is not insignificant

Cluster – A group of tokens

K-Cluster – A cluster of size K

[t1, t2, …, tk] – A cluster of size k containing the tokens t1, t2, …, tk. In other words, it is a K-Cluster of the tokens t1, t2, …, tk

We say that a K-Cluster [t1, t2, …, tk] appears/occurs in a tweet iff each of t1, t2, …, tk appears in it.

a([t]) – total number of tweets that the cluster [t] appears in. Equivalently, total number of tweets that the token t appears in

a([t1, t2, …, tk]) – total number of tweets that the cluster [t1, t2, …, tk] appears in

C([t1, t2, …, tk], t) –
$$\frac{a([t1, t2, \ldots, tk, t])}{a([t1, t2, \ldots, tk])}$$

Thus, $0 <= C([t1, t2, \ldots, tk], t) <= 1$

C([t1, t2, …, tk], t) determines how close t is to the cluster [t1, t2, …, tk] by looking at the proportion of the number of tweets that t appears in amongst the tweets that [t1, t2, …, tk] appears in. The higher the value of C([t1, t2, …, tk], t), the closer t is to [t1, t2, …, tk].

C’([t1, t2, …, tk], t) –
$$\frac{C([t1], t) + C([t2], t) + \ldots + C([tk], t)}{K}$$

Thus, $0 <= C'([t1, t2, \ldots, tk], t) <= 1$

C’([t1, t2, …, tk], t) determines how close t is to cluster [t1, t2, …, tk] by looking at the average distance of t from each of the clusters [t1], [t2], …, [tk]. The higher the value of C’([t1, t2, …, tk], t), the closer t is to [t1, t2, …, tk].

C_Table_K – A table which is created to store the values of C([t1, t2, … tk], t), where [t1, t2, …, tk] is a k-Cluster and t is a token.

The following table denotes a general C_Table_K:

	token-1	**token-2**	**…**	**…**	**token-n**
k-Cluster-1	C(k-Cluster-1, token-1)	C(k-Cluster-1, token-2)			C(k-Cluster-1, token-n)
k-Cluster-2	C(k-Cluster-2, token-1)	C(k-Cluster-2, token-2)			C(k-Cluster-2, token-n)
.					
.					
.					
k-Cluster-m	C(k-Cluster-m, token-1)	C(k-Cluster-m, token-2)	…	…	C(k-Cluster-m, token-n)

In the above table,

- token-1, token-2, …, token-n are ‘n’ different tokens
- k-Cluster-1, k-Cluster-2, …, k-Cluster-m are ‘m’ different k-Clusters. Thus each k-Cluster-j is of the form [tj1, tj2, …, tjk].

C'_Table_K – A table which is created to store the values of C'([t1, t2, … tk], t), where [t1, t2, …, tk] is a k-Cluster and t is a token.

Parameters:

There are a few parameters that we define in this section. The values for these parameters are found after performing several experiments. The support and confidence values are as defined in the book by Hellerstein and Stonebraker [18].

Support level - Some tokens appear in very few tweets, while others appear in many tweets. For example, considering a token which appears in only one tweet will not significantly affect the clustering. Support level defines the minimum number of tweets that a cluster should occur in before it can be considered to form bigger clusters.

Confidence level – This is the minimum value that C([t1, t2, …, tk], t) should have before t1, t2, … tk, t can be clustered together.

Max Cluster – This is the maximum number of words that a cluster can have. In our case, since we treat tweets containing 7 or more similar words to be retweets, we certainly cannot have Max Cluster to be greater than 7. We set this value to be 5.

Recap of our two algorithms:

Algorithm-I performs clustering based on **C** value, i.e., a cluster [t1, t2, …, tk] and token t are clustered if the value of C([t1, t2, …, tk], t) is higher than the confidence level and the cluster [t1, t2, …, tk, t] appears in atleast support level number of tweets. C([t1, t2, …, tk], t) looks for the **ratio of tweets** that the cluster [t1, t2, …, tk, t] appears in amongst the tweets that [t1, t2, …, tk] appears in.

Algorithm-II performs clustering based on **C'** value, i.e., a cluster [t1, t2, …, tk] and token t are clustered if the value of C'([t1, t2, …, tk], t) is higher than the confidence level and the cluster [t1, t2, …, tk, t] appears in atleast support level number of tweets. C'([t1, t2, …, tk], t) looks for the **average distance of the token** t from each of the clusters [t1], [t2], …, [tk].

Example trace of the clustering algorithms:

Given a set of tweets, this section traces an example of the algorithms being used to form clusters of tweets. The example below is used through the rest of this section to analyze each step of the algorithms.

For illustration purposes we set Support level = 1 and Confidence level = 0.5 in our example. We will see later why a support level of 1 makes no sense. We will reset this value to a better value at a later stage.

Tweet ID	Tweet	Tokens
1	I love the rain	love, rain
2	Is it going to rain tomorrow?	going, rain, tomorrow
3	I am going to LA tomorrow	going, LA, tomorrow
4	Looks like it might rain	looks, like, might, rain

Figure1: Example

Here are a few preliminary steps we perform before getting to the core of the clustering algorithm.

(a) Each tweet is given a unique id.
(b) The insignificant words are removed from the tweets and a list containing the tokens is formed.
(c) For each token a mapping between the tokens and the list of id of the tweets that the token appears in is maintained.

The following figure shows how a token is mapped to the list of id of the tweets that it appears in:

Token	Id
love	1
rain	1, 2, 4
going	2, 3
tomorrow	2, 3
LA	3
looks	4
like	4
might	4

Figure 2: Mapping between the tokens and the list of id of the tweets

Clustering is performed iteratively. We start with 1-Clusters and iteratively build clusters of larger size in each step. More specifically, in the kth step of the iteration process, we compare the k-Clusters created in the previous step with each of the existing tokens to see if any of them could be clustered together to form a (k+1)-Cluster.

Note that,

C([t1], t2) = C'([t1], t2).

Thus the tables C_Table_1 and C'_Table_1 are the same.

	love	rain	going	tomorrow	LA	looks	Like	might
[love]	1	1	0	0	0	0	0	0
[rain]	1/3	1	1/3	1/3	0	1/3	1/3	1/3
[going]	0	1/2	1	1	1/2	0	0	0
[tomorrow]	0	1/2	1	1	1/2	0	0	0
[LA]	0	0	1	1	1	0	0	0
[looks]	0	1	0	0	0	1	1	1
[like]	0	1	0	0	0	1	1	1
[might]	0	1	0	0	0	1	1	1

🟥	**The corresponding cluster and token can be clustered, i.e., C value >= Confidence level**
⬜	**The corresponding token is already part of the cluster**

Figure 3: C_Table_1 / C'_Table_1

Every step of Algorithm-I could be viewed as creating C_Table_K in Step k using the k-Clusters created in the previous step. Figure 3 gives a snapshot of C_Table_1. The above table gives an idea about the tokens which can be combined to form 2-Clusters. When the C value is greater than or equal to the Confidence level, the corresponding cluster and token are clustered to form a new cluster. Some cells are marked in grey. The corresponding cluster and token of such a cell is ignored as the token is already a part of the cluster. After the first iteration, we have the following list of 2-Clusters:

[love, rain]
[going, rain]
[going, tomorrow]
[going, LA]
[tomorrow, rain]
[tomorrow, LA]
[looks, rain]
[looks, like]
[looks, might]
[like, rain]
[like, might]
[might, rain]

Figure 4: 2-Clusters formed after the first iteration

Notice that since we set the Support level to be just one, we end up with several clusters which occur in just one tweet. When we have only one tweet in which the cluster occurs, the corresponding cells of that cluster have only 1 or 0 as the C value in the C_Table. For example, consider the cluster [looks, might]. In the C_Table_2 we see that all the columns are either 1 or 0 for this cluster, i.e., given any token t, we find that C([looks,might], t) = 1 or 0. This shows that Support level has to be greater than 1.

We now reset our Support level to be 2. Thus we ignore the clusters which occur in only one tweet. After resetting the value of Support level we are left with only one cluster [going, tomorrow]. Note that we have set Support level to be 2 for this example only. The actual value for this parameter is found after conducting several experiments.

In the second iteration, we try to form 3-Clusters based on the 2-Clusters formed in the previous step. After the second iteration, we arrive at the following clusters:

[going, tomorrow, rain]
[going, tomorrow, LA]

Figure 5: Clusters formed after Step 2

The corresponding C_Table_2 is shown in Figure 6. We continue this iterative process until all the clusters with *Max_Cluster* number of tokens are formed or until no more clustering could be done. Finally, duplicate clusters are deleted. Thus we get various clusters each of varying length.

We now illustrate how Algorithm-II works for the same example. Note that Algorithm-II works in a similar fashion as that of Algorithm-I except that we create a C’_Table_k table instead of C_Table_k table at every step.

As in Algorithm-I, every step of Algorithm-II could be viewed as creating C’_Table_K in Step k using the k-Clusters created in the previous step. Figure 3 gives a snapshot of C’_Table_1. It gives an idea about the tokens which can be combined to form 2-Clusters. When the C’ value is greater than or equal to the Confidence level, the corresponding cluster and token are clustered to form a new cluster.

For illustration purposes we have set **Support level = 1 and Confidence level = 0.5**. Thus, after the first iteration the list of 2-Clusters formed would be the same as in Figure 4.

In the next iteration we look for 3-Clusters that can be formed using the 2-Clusters formed in the previous step. For this we populate C’_Table_2 (shown in Figure 7) and then decide which elements could be clustered further. Notice that C’_Table_2 is different from C_Table_2.

If we change the Support value to 2 as we did for Algorithm-I then the list of clusters formed would be the same as that in Figure 5. However we continue the process for a Support value of 1 for illustration purposes.

	love	rain	going	tomorrow	LA	looks	like	might
[love, rain]			0	0	0	0	0	0
[going, rain]	0			1	0	0	0	0
[going, tomorrow]	0	1/2			1/2	0	0	0
[going, LA]	0	0		1		0	0	0
[tomorrow, rain]	0		1		0	0	0	0
[tomorrow, LA]	0		1			0	0	0
[looks, rain]	0		0	0	0		1	1
[looks, like]	0	1	0	0	0			1
[looks, might]	0	1	0	0	0		1	
[like, rain]	0		0	0	0	1		1
[like, might]	0	1	0	0	0	1		
[might, rain]	0		0	0	0	1	1	

🟧	The number of tweets that this cluster occurs = 1
▨	The tokens are already part of the cluster

Figure 6: C_Table_2

	love	rain	going	tomorrow	LA	looks	like	might
[love, rain]			1/6	1/6	0	1/6	1/6	1/6
[going, rain]	1/6			2/3	1/4	1/6	1/6	1/6
[going, tomorrow]	0	1/2			1/2	0	0	0
[going, LA]	0	1/4		1		0	0	0
[tomorrow, rain]	1/6		1		1/4	1/6	1/6	1/6
[tomorrow, LA]	0	1/4	1			0	0	0
[looks, rain]	1/6		1/6	1/6	0		2/3	2/3
[looks, like]	0	1	0	0	0			1
[looks, might]	0	1	0	0	0		1	
[like, rain]	1/6		1/6	1/6	0	2/3		2/3
[like, might]	0	1	0	0	0	1		
[might, rain]	1/6		1/6	1/6	0	2/3	2/3	

🟥	**C' value >= Confidence level**
▨	**The corresponding token is already part of the cluster**

Figure 7: C'_Table_2

The table below gives a list of clusters formed after the second iteration assuming that the Support value is 1.

[going, rain, tomorrow]
[going, tomorrow, LA]
[looks, like, rain]
[looks, might, rain]
[like, might, rain]

Figure 8: Clusters formed after Step 2

We continue this iterative process until all the clusters with *Max_Cluster* number of tokens are formed or until no more clustering could be done. Finally, duplicate clusters are deleted. Thus we get various clusters each of varying length.

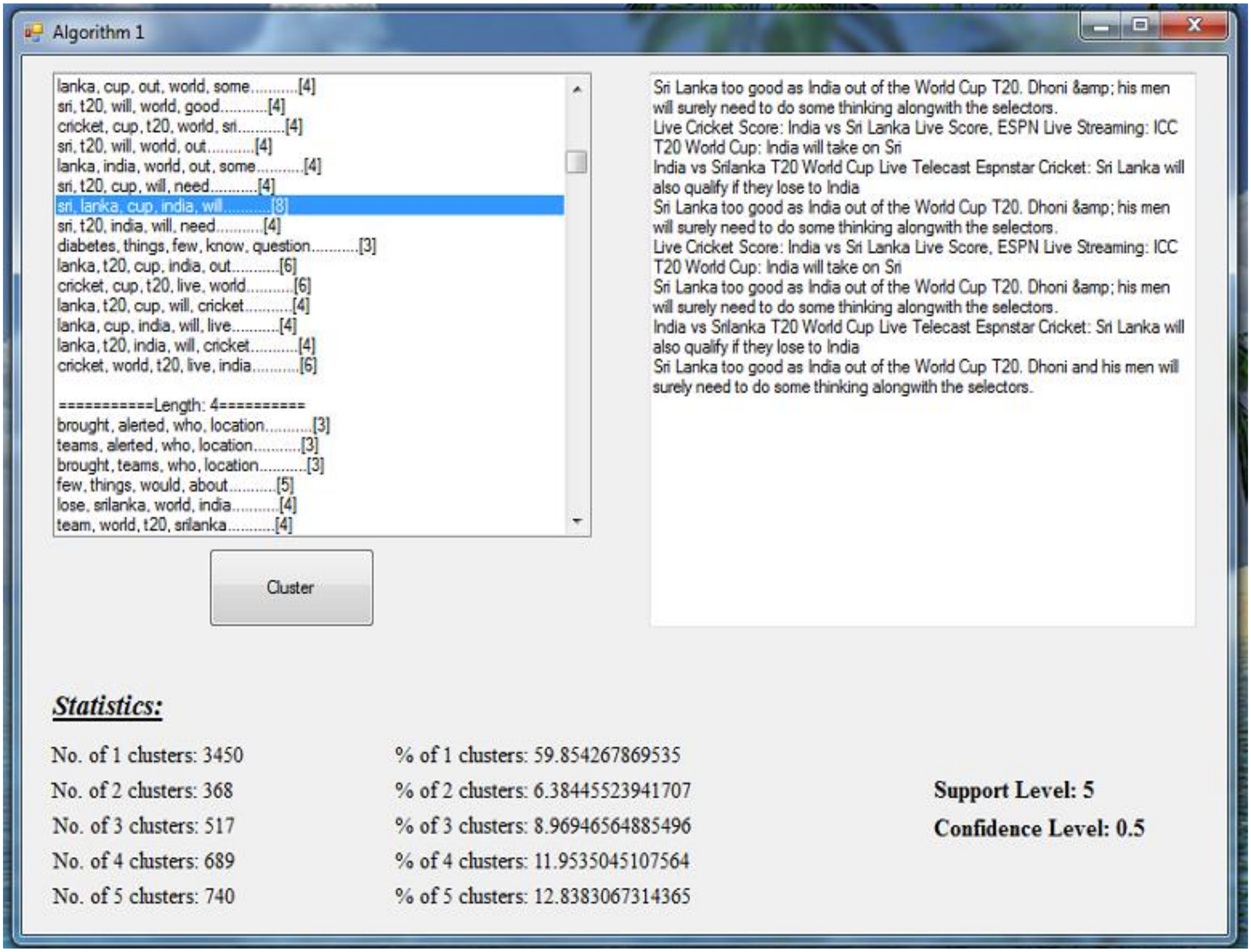
Software Implementation

Microsoft Visual C # 2008 Express Edition has been installed on a Windows 7 machine (Pentium Dual Core, 4GB Ram) to aid in programming. SQL Management Studio 2008 has been installed and has been used as the SQL client.

A total of 925 tweets have been used as a sample. Sample data collected contain tweets comprising of various topics with common keywords. The above algorithms have been fully implemented using C#. Core functionality includes – reading the tweets, tokenizing them, clustering them using the algorithm and returning clustered output.

A simple front end GUI has been developed as part of the implementation. On the left, you would see all the tweets read from a file. When the user clicks on the “Cluster” button, the clustering process starts and upon completion, the clustered output of tweets is displayed in a text box in the descending order of cluster length. Ex: Clusters of length 5 are displayed followed by clusters of length 4 followed by clusters of length 3 and so on…

Figure 9 gives a snapshot of an execution of Algorithm-I. The results shown are for a Confidence level of 0.5 and Support level of 5. The ListBox on the left lists the various clusters formed organized according to their length. Whenever a cluster is selected from the ListBox on the left, the corresponding Tweets that the cluster occurs in are displayed in the TextBox on the right. In the figure, the Tweets that are displayed on the TextBox on the right correspond to the cluster that has been selected from the ListBox on the left.



```
lanka, cup, out, world, some...........[4]
sri, t20, will, world, good...........[4]
cricket, cup, t20, world, sri...........[4]
sri, t20, will, world, out...........[4]
lanka, india, world, out, some...........[4]
sri, t20, cup, will, need...........[4]
sri, lanka, cup, india, will...........[8]
sri, t20, india, will, need...........[4]
diabetes, things, few, know, question...........[3]
lanka, t20, cup, india, out...........[6]
cricket, cup, t20, live, world...........[6]
lanka, t20, cup, will, cricket...........[4]
lanka, cup, india, will, live...........[4]
lanka, t20, india, will, cricket...........[4]
cricket, world, t20, live, india...........[6]

===========Length: 4==========
brought, alerted, who, location...........[3]
teams, alerted, who, location...........[3]
brought, teams, who, location...........[3]
few, things, would, about...........[5]
lose, srilanka, world, india...........[4]
team, world, t20, srilanka...........[4]
```

Sri Lanka too good as India out of the World Cup T20. Dhoni &amp; his men will surely need to do some thinking alongwith the selectors.
Live Cricket Score: India vs Sri Lanka Live Score, ESPN Live Streaming: ICC T20 World Cup: India will take on Sri
India vs Srilanka T20 World Cup Live Telecast Espnstar Cricket: Sri Lanka will also qualify if they lose to India
Sri Lanka too good as India out of the World Cup T20. Dhoni &amp; his men will surely need to do some thinking alongwith the selectors.
Live Cricket Score: India vs Sri Lanka Live Score, ESPN Live Streaming: ICC T20 World Cup: India will take on Sri
Sri Lanka too good as India out of the World Cup T20. Dhoni &amp; his men will surely need to do some thinking alongwith the selectors.
India vs Srilanka T20 World Cup Live Telecast Espnstar Cricket: Sri Lanka will also qualify if they lose to India
Sri Lanka too good as India out of the World Cup T20. Dhoni and his men will surely need to do some thinking alongwith the selectors.

Statistics:

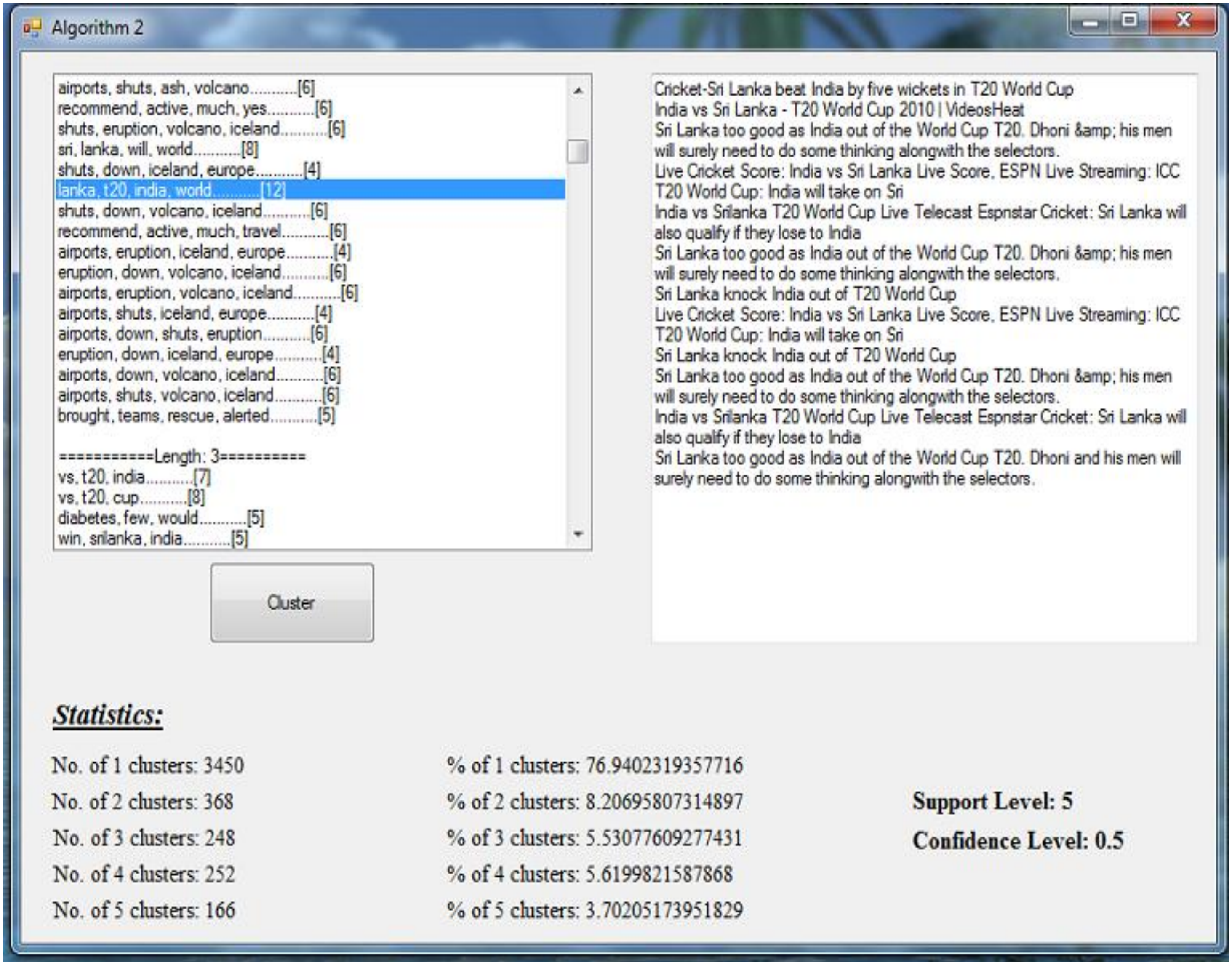
No. of 1 clusters: 3450
No. of 2 clusters: 368
No. of 3 clusters: 517
No. of 4 clusters: 689
No. of 5 clusters: 740

% of 1 clusters: 59.854267869535
% of 2 clusters: 6.38445523941707
% of 3 clusters: 8.96946564885496
% of 4 clusters: 11.9535045107564
% of 5 clusters: 12.8383067314365

Support Level: 5
Confidence Level: 0.5

Figure 9: Algorithm-I

Figure 10 gives a snapshot of an execution of Algorithm-II. The results shown are for a Confidence level of 0.5 and Support level of 5. The ListBox on the left lists the various clusters formed organized according to their length. Whenever a cluster is selected from the ListBox on the left, the corresponding Tweets that the cluster occurs in are displayed in the TextBox on the right. In the figure, the Tweets that are displayed on the TextBox on the right correspond to the cluster that has been selected from the ListBox on the left.



```
airports, shuts, ash, volcano...........[6]
recommend, active, much, yes...........[6]
shuts, eruption, volcano, iceland...........[6]
sri, lanka, will, world...........[8]
shuts, down, iceland, europe...........[4]
lanka, t20, india, world...........[12]
shuts, down, volcano, iceland...........[6]
recommend, active, much, travel...........[6]
airports, eruption, iceland, europe...........[4]
eruption, down, volcano, iceland...........[6]
airports, eruption, volcano, iceland...........[6]
airports, shuts, iceland, europe...........[4]
airports, down, shuts, eruption...........[6]
eruption, down, iceland, europe...........[4]
airports, down, volcano, iceland...........[6]
airports, shuts, volcano, iceland...........[6]
brought, teams, rescue, alerted...........[5]

===========Length: 3==========
vs, t20, india...........[7]
vs, t20, cup...........[8]
diabetes, few, would...........[5]
win, srilanka, india...........[5]
```

```
Cricket-Sri Lanka beat India by five wickets in T20 World Cup
India vs Sri Lanka - T20 World Cup 2010 | VideosHeat
Sri Lanka too good as India out of the World Cup T20. Dhoni &amp; his men
will surely need to do some thinking alongwith the selectors.
Live Cricket Score: India vs Sri Lanka Live Score, ESPN Live Streaming: ICC
T20 World Cup: India will take on Sri
India vs Srilanka T20 World Cup Live Telecast Espnstar Cricket: Sri Lanka will
also qualify if they lose to India
Sri Lanka too good as India out of the World Cup T20. Dhoni &amp; his men
will surely need to do some thinking alongwith the selectors.
Sri Lanka knock India out of T20 World Cup
Live Cricket Score: India vs Sri Lanka Live Score, ESPN Live Streaming: ICC
T20 World Cup: India will take on Sri
Sri Lanka knock India out of T20 World Cup
Sri Lanka too good as India out of the World Cup T20. Dhoni &amp; his men
will surely need to do some thinking alongwith the selectors.
India vs Srilanka T20 World Cup Live Telecast Espnstar Cricket: Sri Lanka will
also qualify if they lose to India
Sri Lanka too good as India out of the World Cup T20. Dhoni and his men will
surely need to do some thinking alongwith the selectors.
```

Statistics:

No. of 1 clusters: 3450 | % of 1 clusters: 76.9402319357716

No. of 2 clusters: 368 | % of 2 clusters: 8.20695807314897

No. of 3 clusters: 248 | % of 3 clusters: 5.53077609277431

No. of 4 clusters: 252 | % of 4 clusters: 5.6199821587868

No. of 5 clusters: 166 | % of 5 clusters: 3.70205173951829

Support Level: 5

Confidence Level: 0.5

Figure 10: Algorithm-II

Results:

Of the total clusters formed, the percentage of 1-Clusters, percentage of 2-Clusters,…, percentage of 5-Clusters formed were compared for various values of Confidence level and Support level.

In general, the percentage of 5-Clusters decreased as the Support level value increased, i.e., as the Support level increased, the number of larger length clusters reduced. This is due to the fact that the newly formed clusters were required to be part of a higher number of tweets thus making it less plausible to be clustered further.

On the other hand, as the value of Confidence level varied, it seemed to have different effects on clusters of different length for different Support levels.

The following two graphs show how the *Support Level* impacts the percentage of *K-Clusters* formed for both the algorithms for a Confidence level of 0.3. Of the total number of clusters formed, the percentage of 1-Clusters, 2-Clusters, …, 5-Clusters is shown for a Support level of 3, 4, and 5.

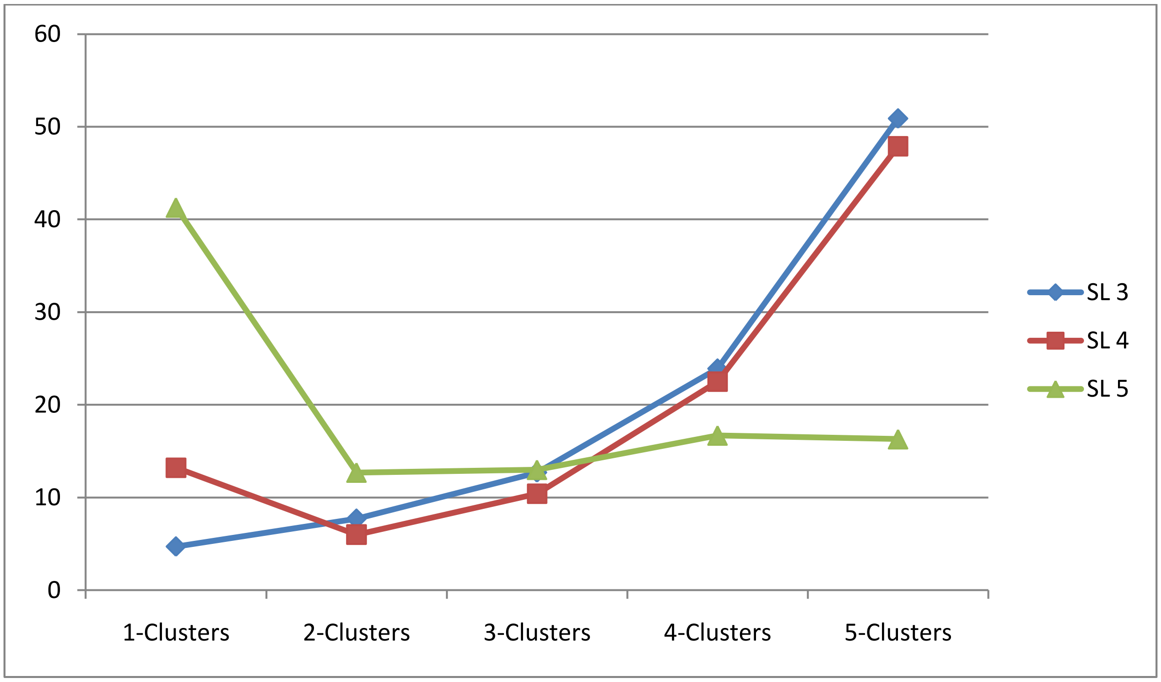


Figure 11: Algorithm-I with Confidence level 0.3

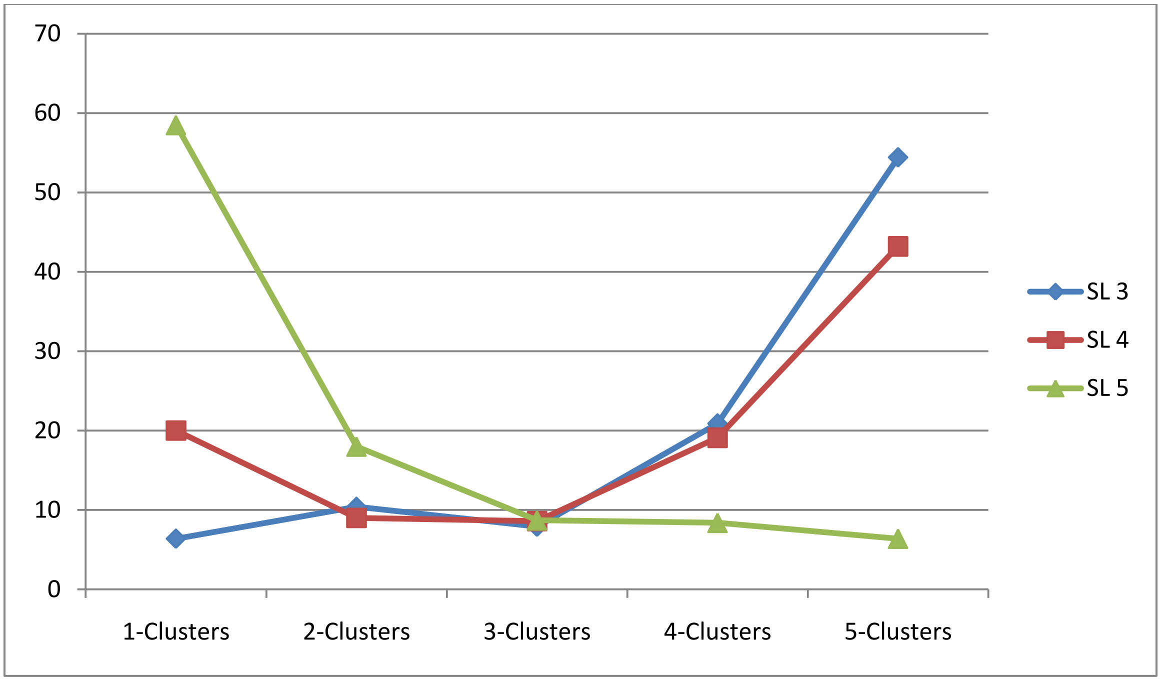


Figure 12: Algorithm-II with Confidence level 0.3

The following two graphs show how the *Support Level* impacts the percentage of *K-Clusters* formed for both the algorithms for a Confidence level of 0.4. Of the total number of clusters formed, the percentage of 1-Clusters, 2-Clusters, …, 5-Clusters is shown for a Support level of 3, 4, and 5.

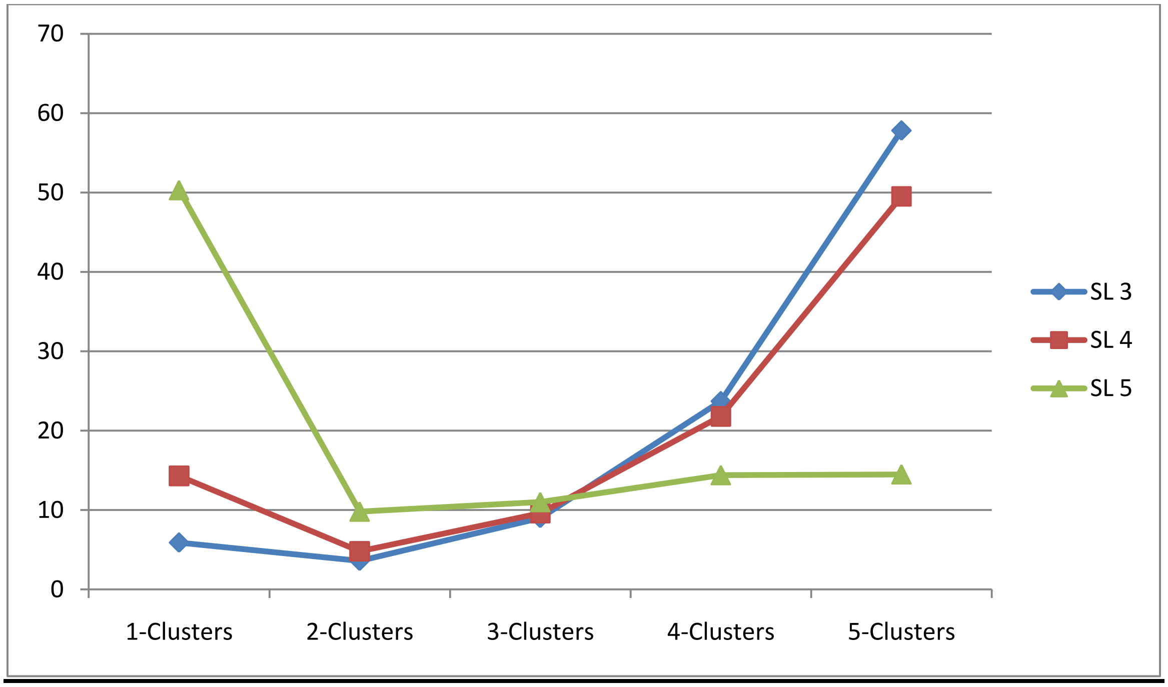


Figure 13: Algorithm-I with Confidence level 0.4

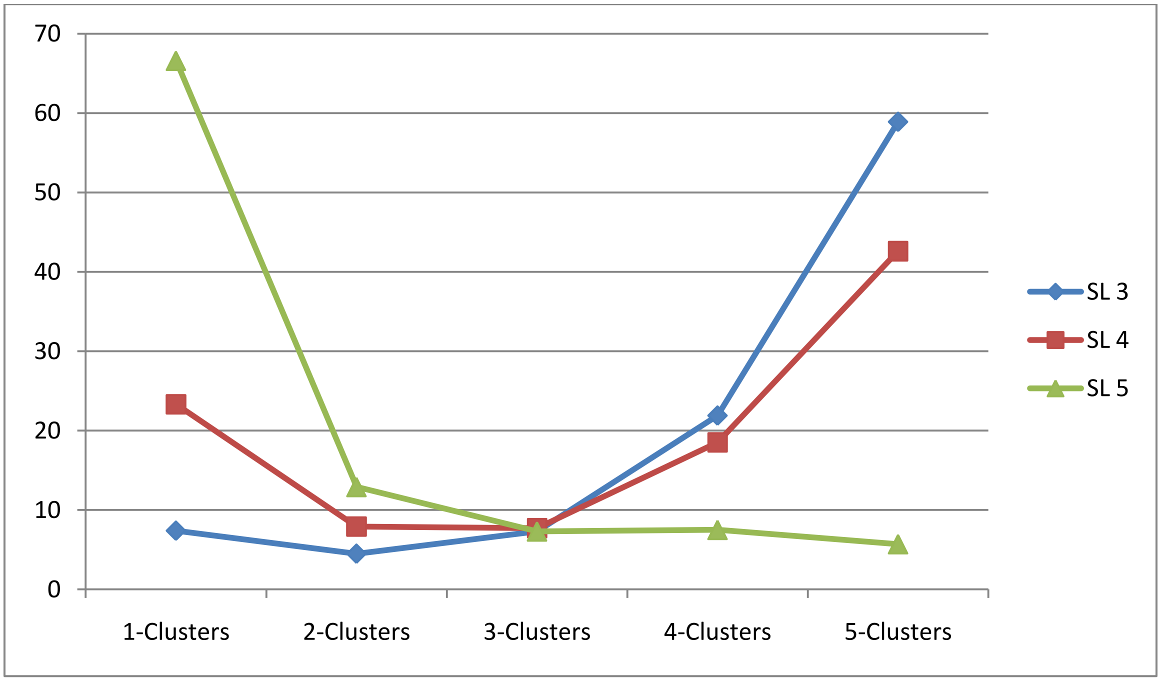


Figure 14: Algorithm-II with Confidence level 0.4

The following two graphs show how the *Support Level* impacts the percentage of *K-Clusters* formed for both the algorithms for a Confidence level of 0.5. Of the total number of clusters formed, the percentage of 1-Clusters, 2-Clusters, …, 5-Clusters is shown for a Support level of 3, 4, and 5.

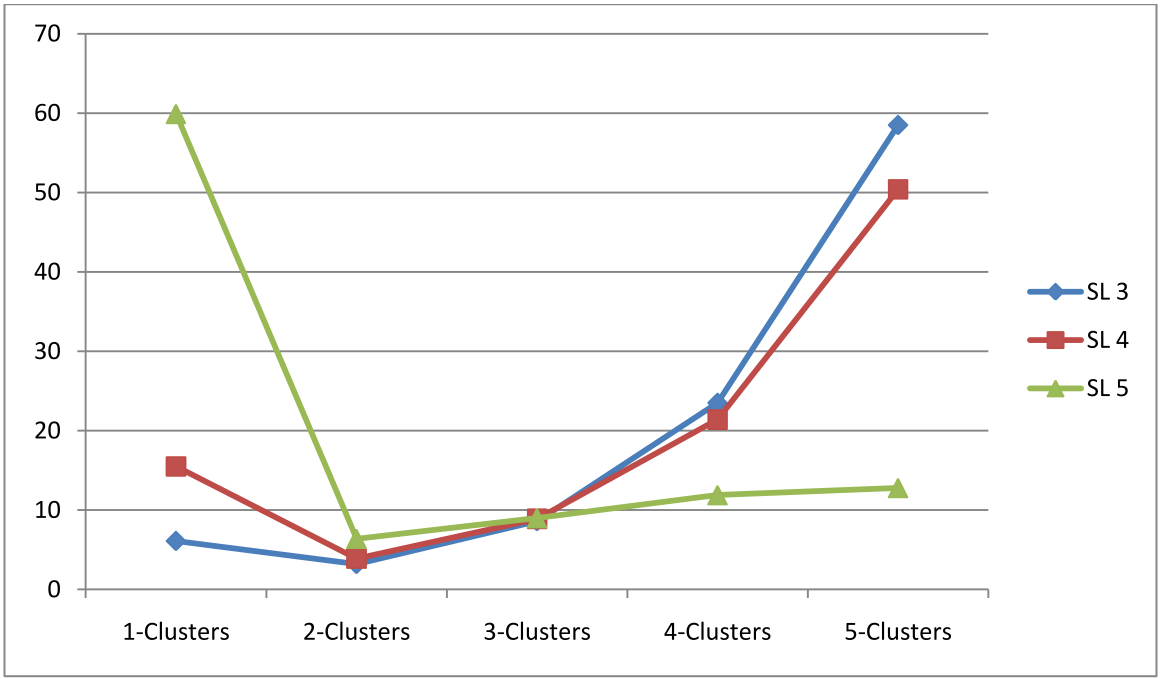


Figure 15: Algorithm-I with Confidence level 0.5

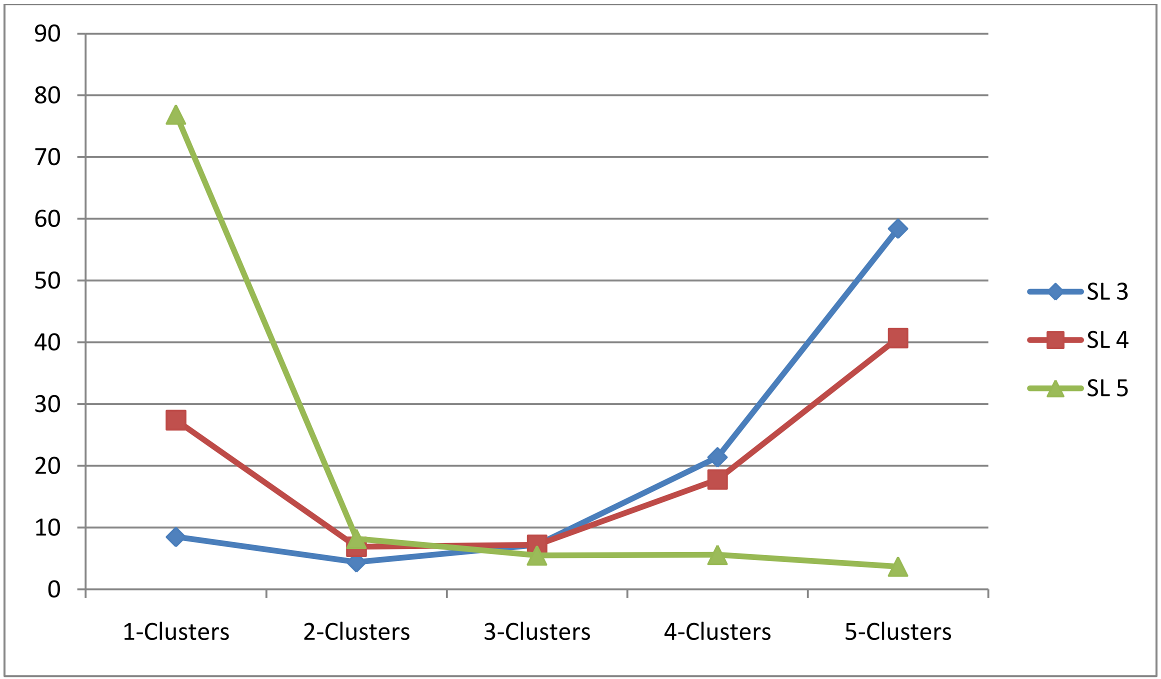
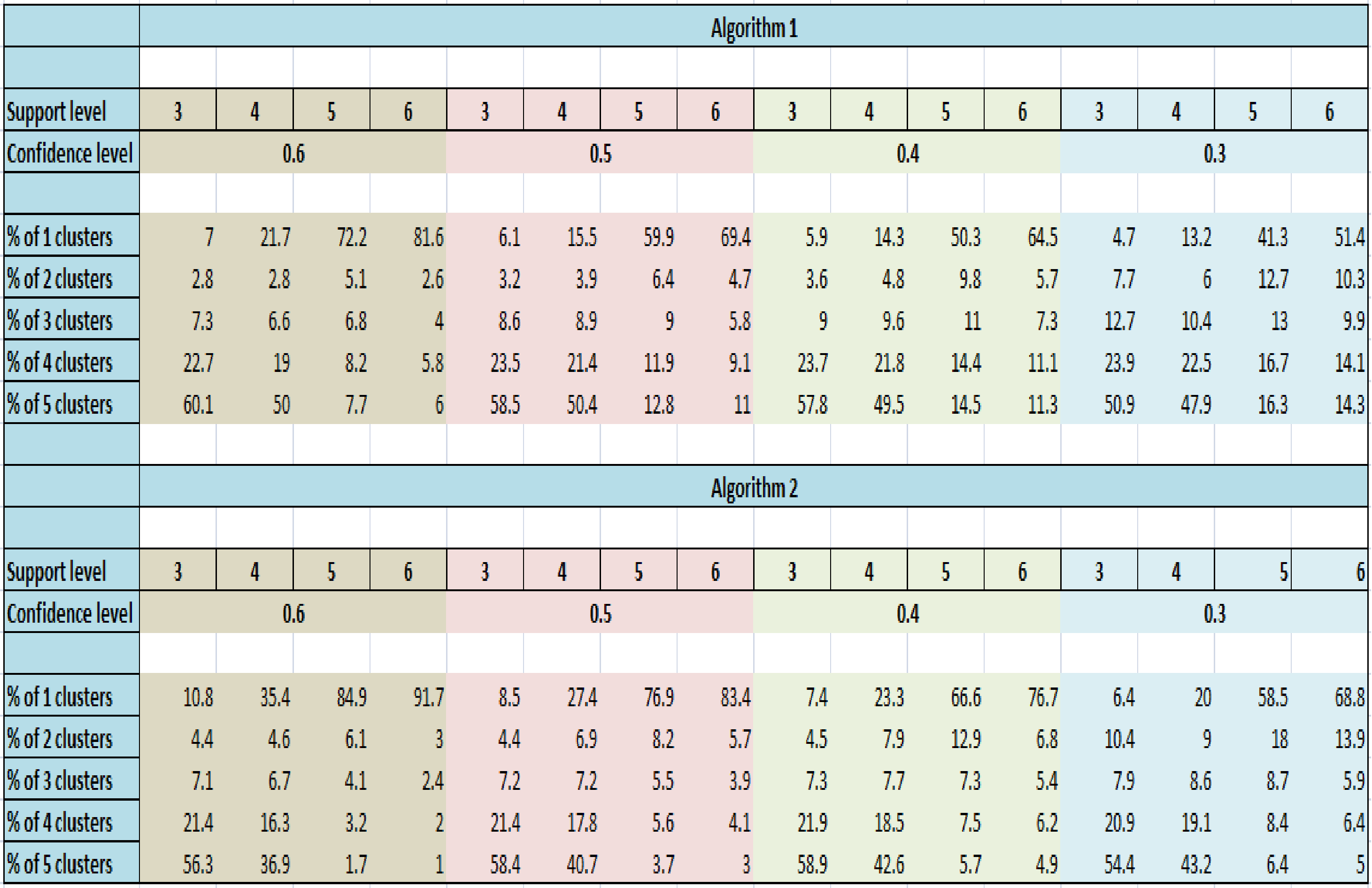


Figure 16: Algorithm-II with Confidence level 0.5

Note that according to the construction of our algorithms, the number of 1-Clusters formed is the same for both the algorithms irrespective of the Support level and Confidence level chosen. Thus finding the percentage of k-Clusters gives us an idea about how the number of larger clusters grows for each value of Confidence level and Support level. We do not want too many large clusters. Instead we would like to have fewer clusters which are tightly packed. Thus we do not want the number of larger clusters formed to exceed the number of 1-Clusters.

The figure below displays the percentage of k-Clusters formed for our sample data for various Confidence level and Support level values for both the Algorithms.



	Algorithm 1															
Support level	3	4	5	6	3	4	5	6	3	4	5	6	3	4	5	6
Confidence level	0.6				0.5				0.4				0.3			
% of 1 clusters	7	21.7	72.2	81.6	6.1	15.5	59.9	69.4	5.9	14.3	50.3	64.5	4.7	13.2	41.3	51.4
% of 2 clusters	2.8	2.8	5.1	2.6	3.2	3.9	6.4	4.7	3.6	4.8	9.8	5.7	7.7	6	12.7	10.3
% of 3 clusters	7.3	6.6	6.8	4	8.6	8.9	9	5.8	9	9.6	11	7.3	12.7	10.4	13	9.9
% of 4 clusters	22.7	19	8.2	5.8	23.5	21.4	11.9	9.1	23.7	21.8	14.4	11.1	23.9	22.5	16.7	14.1
% of 5 clusters	60.1	50	7.7	6	58.5	50.4	12.8	11	57.8	49.5	14.5	11.3	50.9	47.9	16.3	14.3

	Algorithm 2															
Support level	3	4	5	6	3	4	5	6	3	4	5	6	3	4	5	6
Confidence level	0.6				0.5				0.4				0.3			
% of 1 clusters	10.8	35.4	84.9	91.7	8.5	27.4	76.9	83.4	7.4	23.3	66.6	76.7	6.4	20	58.5	68.8
% of 2 clusters	4.4	4.6	6.1	3	4.4	6.9	8.2	5.7	4.5	7.9	12.9	6.8	10.4	9	18	13.9
% of 3 clusters	7.1	6.7	4.1	2.4	7.2	7.2	5.5	3.9	7.3	7.7	7.3	5.4	7.9	8.6	8.7	5.9
% of 4 clusters	21.4	16.3	3.2	2	21.4	17.8	5.6	4.1	21.9	18.5	7.5	6.2	20.9	19.1	8.4	6.4
% of 5 clusters	56.3	36.9	1.7	1	58.4	40.7	3.7	3	58.9	42.6	5.7	4.9	54.4	43.2	6.4	5

Figure 17: Percentage of k-Clusters formed for various Confidence level and Support level values

Many experiments were conducted with varying values of the parameters – *Confidence level* and *Support level.* Algorithm-II always seemed to have fewer larger clusters than Algorithm-I. The least value of Confidence level and Support level that seemed to give reasonable results was found to be as follows for both the algorithms for our sample data:

- Confidence Level = 0.4
- Support Level = 5

Conclusion:

A survey of two techniques of clustering was conducted. Two different distance functions were used in a hierarchical clustering algorithm and the effect of the distance function on the algorithm was analyzed. The percentage of 1-Clusters, 2-Clusters,…, 5-Clusters formed was calculated to see if they varied for both the algorithms for different values of Confidence level and Support level. Since the number of 1-Clusters formed is the same for both the algorithms irrespective of the Support level and Confidence level chosen, finding the percentage of k-Clusters gives us an idea about how the number of larger clusters grows for each value of Confidence level and Support level.

The algorithms presented here help group the tweets into clusters which help the user parse the results better. Thus we do not want too many large clusters. Instead we are looking for fewer clusters that are tightly packed. The percentage of 1-clusters formed was found to be higher for Algorithm-II than Algorithm-I for all values of Support level and Confidence level for our sample data. The percentage of larger clusters gradually increased for Support level 3 and 4 for both the algorithms. Although the overall behavior was found to be similar for both the algorithms, Algorithm-II always seemed to fare better for each of the Confidence level and Support level values for our sample data.

Future work:

In the original algorithm given by Ward, he initially considers ‘k’ different clusters each containing one of the ‘k’ given elements. He then iteratively merges clusters based on their distance from each other. Consider the following two variations of Ward’s algorithm:

Variation-I:
Step 1: Given ‘k’ different elements, run ‘n’ iterations of Algorithm-I
Step 2: Run Ward’s algorithm by considering the clusters formed in the above step as the initial clusters. (Note that Ward originally considered ‘k’ different clusters each containing one of the ‘k’ different elements as the initial clusters)

Variation-II:
Step 1: Given ‘k’ different elements, run ‘n’ iterations of Algorithm-II
Step 2: Run Ward’s algorithm by considering the clusters formed in the above step as the initial clusters. (Note that Ward originally considered ‘k’ different clusters each containing one of the ‘k’ different elements as the initial clusters)

In the future, it would be interesting to analyze the behavior of the above two variations of Ward’s algorithm. Note that the number ‘n’ has not been specified. We could see how the behavior changes for various values of ‘n’.

Acknowledgment

I owe my deepest gratitude to Dr Evan Golub for his encouragement, guidance and support which was crucial in the completion of this paper. I would also like to thank Dr Nick Roussopoulos for his inspiration and guidance which helped me gain a better understanding of the subject.

Finally I thank my family and friends for supporting me through the completion of the paper.

References:

[1] "Twitter – Wikipedia, the free encyclopedia", Feb. 20, 2010. [Online]. Available: http://en.wikipedia.org/wiki/Twitter. [Accessed: Feb. 23, 2010]

[2] Matt Williams, "Governments use Twitter for Emergency Alerts, Traffic Notices and More", *Government Technology*, Jan. 7, 2009. [Online]. Available: http://www.govtech.com/gt/579338. [Accessed Feb. 23, 2010]

[3] Biz Stone, "Twitter Blog - What's Happening?", *Twitter Blog*, Nov. 19, 2009. [Online] Available: http://blog.twitter.com/2009/11/whats-happening.html. [Accessed: Feb. 23, 2010]

[4] PearAnalytics, "Twitter Study – August 2009" [Online] Available: http://www.pearanalytics.com/wp-content/uploads/2009/08/Twitter-Study-August-2009.pdf. [Accessed: Feb. 23, 2010]

[5] Rashmi Sinha, "Clustering comes to Flickr", R*ashmi's Blog,* Aug. 4, 2005. [Online] Available: http://rashmisinha.com/2005/08/04/clustering-comes-to-flickr [Accessed: Feb. 23, 2010]

[6] A.K Jain, M.N.Murty and P.J.Flynn, "Data clustering: A Review", *ACM Computing Surveys, Vol. 31, No. 3*, September 1999. Available: http://citeseerx.ist.psu.edu/viewdoc/download;jsessionid=346683B81CA7EEB2CD9159240DE816E2?doi=10.1.1.18.2720&rep=rep1&type=pdf

[7] Themos Kalafatis, "Emotions, Beliefs and Analytics", *Life Analytics Blog*. [Online] Available: http://lifeanalytics.blogspot.com/2009/01/emotions-beliefs-and-analytics.html. [Accessed: Mar. 19, 2010]

[8] Alessio Signorini, "Why Twitter Clients still lack Classification, Clustering and Ranking?", *Tech Research and Life Blog*, Feb 7, 2010. [Online] Available: http://blog.alessiosignorini.com/2010/02/why-twitter-clients-still-lack-classification-clustering-and-ranking/. [Accessed: Mar. 19, 2010]

[9] Themos Kalafatis, "Emotions, Beliefs and Analytics", *Life Analytics Blog*. [Online] Available: http://lifeanalytics.blogspot.com/2009/01/clustering-thoughts-of-twitter-users.html. [Accessed: Mar. 19, 2010]

[10] Paul, "TopicScoop: Twitter Search with real-time topic clustering", *Tech Combo Blog*, Sept. 28, 2009. [Online] Available: http://techcombo.com/2009/09/28/topicscoop-twitter-search-with-real-time-topic-clustering-123/ [Accessed: Mar. 19, 2010]

[11] Bill Heil and Mikolaj Piskorski, " New Twitter Research: Men Follow Men and Nobody Tweets", *Harvard Business Review Blog*, Jun 1, 2009. [Online] Available: http://blogs.hbr.org/cs/2009/06/new_twitter_research_men_follo.html. [Accessed: Mar. 19, 2010]

[12] Ben Parr, "5 Terrific Twitter Research Tools", *Mashtable*. [Online] Available: http://mashable.com/2009/05/03/twitter-research-tools/. [Accessed: Mar. 19, 2010]

[13] "HubSpot's Inbound Internet Marketing Blog", Mar 2, 2009. [Online] Available: http://blog.hubspot.com/blog/tabid/6307/bid/4594/Is-22-Tweets-Per-Day-the-Optimum.aspx [Accessed: Apr 9, 2010]

[14] "Languages by Average word length". [Online] Available: http://blogamundo.net/lab/wordlengths/ [Accessed: Apr 9, 2010]

[15] "Twitter facts from the Oxford English Corpus". [Online] Available: http://www.askoxford.com/pressroom/archive/twitter_facts.pdf [Accessed: Apr 9, 2010]

[16] Rakesh Agrawal, Ramakrishnan Srikant, "Fast Algorithms for Mining Association Rules", *Proceedings of the 20th VLDB Conference Santiago, Chile,* 1994

[17] Tian Zhang, Raghu Ramakrishnan, Miron Livny, "BIRCH: An Efficient Data Clustering Method for Very Large Databases", *ACM* 1996

[18] Joseph M. Hellerstein, Michael Stonebraker, "Readings in Database Systems", Fourth Edition

[19] Peter Doyle and John Saunders, "Market Segmentation and Positioning in Specialized Industrial Markets", *The Journal of Marketing, Vol. 49, No. 2* (Spring, 1985), pp. 24-32

[20] G Punj, DW Stewart, "Cluster analysis in marketing research: review and suggestions for application", *Journal of marketing research*, 1983 - JSTOR

[21] Y Wind, "Issues and advances in segmentation research", *Journal of Marketing Research*, 1978 - JSTOR

[22] Ward, J. H., Jr., "Hierarchical grouping to optimize an objective function", *Journal of the American Statistical Association*, 1963, 58, 236-244.

[23] Stephen c. Johnson, "Hierarchical clustering schemes", Psychometrika vol 32; no. 3; Sept 1967

[24] Weili Wu, Hui Xiong, Shashi Shekhar , "Clustering and information retrieval", 2004

[25] Frank Höppner , Rudolf Kruse , Frank Klawonn , Thomas Runkler, "Fuzzy Cluster Analysis: Methods for Classification, Data Analysis and Image Recognition", 1999

[26] Ulrike von Luxburg, Sébastien Bubeck, Stefanie Jegelka, Michael Kaufmann, “Nearest Neighbor Clustering”

[27] Bubeck S., U. von Luxburg , “Nearest Neighbor Clustering: A Baseline Method for Consistent Clustering with Arbitrary Objective Functions”, *Journal of Machine Learning Research*, 2009.

[28] Peter Cheeseman, James Kelly, Matthew Self, et al., “Auto Class : A Bayesian Classification System”, *Proc. of the 5th Int’l Conf. on Machine Learning, Morgan Kaufmau*, Jun. 1988.

[29] Douglas H. Fisher, “Knowledge Acquisition via Incremental Conceptual Clustering”, *Machine Learning, 2(2)*, 1987

[30] Michael Lebowitz, “Experiments with Incremental Concept Formation: UNIMEM”, *Machine Learning*, 1987.

[31] MacKay, David, "Information Theory, Inference and Learning Algorithms", *Cambridge University Press*, 2003.

[32] MacQueen, J. B., "Some Methods for classification and Analysis of Multivariate Observations", *Proceedings of 5th Berkeley Symposium on Mathematical Statistics and Probability*, 1967

[33] “Cluster analysis – Wikipedia, the free encyclopedia”, [Online]. Available: http://en.wikipedia.org/wiki/Cluster_analysis . [Accessed: Feb. 23, 2010]

[34] “Euclidean Distance – Wikipedia, the free encyclopedia”, [Online]. Available: http://en.wikipedia.org/wiki/Euclidean_distance . [Accessed: Jul. 09, 2010]

[35] “Levenshtein Distance – Wikipedia, the free encyclopedia”, [Online]. Available: http://en.wikipedia.org/wiki/Levenshtein_distance . [Accessed: Jul. 09, 2010]

[36] "The tweets of your life, now archived at the Library of Congress", *Reuters*, [Online]. Available: http://blogs.reuters.com/frontrow/2010/04/14/the-tweets-of-your-life-now-archived-at-the-library-of-Congress/ , Apr 2010. [Accessed: Jul 2010]

[37] Girish J. Gulati, Christine B. Williams, "Communicating with Constituents in 140 Characters or Less: Twitter and the Diffusion of Technology Innovation in the United States Congress", *JEL Classification*, Apr 2010

[38] Jennifer Golbeck, “Trust and Nuanced Profile Similarity in Online Social Networks”, *ACM Transactions on the Web*

[39] Sandro Saitta, "Data Miners on Twitter", Feb 2009, [Online]. Available: http://www.dataminingblog.com/data-miners-on-twitter/ , [Accessed: Jul 2010]

[40] "R-Project", [Online]. Available: http://www.r-project.org/ , [Accessed: Jul 2010]

[41] “Association rule learning – Wikipedia, the free encyclopedia”, [Online]. Available: http://en.wikipedia.org/wiki/Association_rule_learning . [Accessed: Jul. 09, 2010]

[42] Jennifer Golbeck, Justin M. Grimes, Anthony Rogers, “Twitter use by the U.S. Congress”, *Journal of the American Society for Information Science and Technology,* May 2010

[43] Christopher Fox, "A stop list for general text", *ACM*, 1989