Research Report on Clustering Algorithm K-Means

**Contents**

Summary 2

Introduction 3

Reasons for Choosing Twitter 3

Data Collection 4

Data Clustering 4

How *k*-means Works 6

Mined Twitter Data 8

Improvements 9

Conclusion 10

**Summary**

Data mining is the process of analyzing data from different viewpoints and summarizing it into very important and useful information for an organization. This information can be used to increase revenue as well as cuts costs. In recent years, the data mining industry is more focused on the information overload is increasing at an alarming rate.

After much deliberation, we planned to create an algorithm where our program fetches user information from the online social networking and micro-blogging service, Twitter. Since its creation in March 2006, Twitter has exploded to over 500 million registered users as of 2012 and is a hive of knowledge waiting to be mined. Information that our collection algorithm mines includes the age of the user’s Twitter account, number of followers, number of tweets, number of friends, and the timing of the last Tweet, among other data**.**

We plan to use the *k*-means clustering algorithm to assist in data analysis. It involves the partitioning of *n* observations into *k* clusters, in which each observation belongs to the cluster with the nearest mean. We have also used a tool, Waikato Environment for Knowledge Analysis (WEKA) as a comparison for our own algorithm. It is a workbench that contains a collection of visualization tools and algorithms for data analysis and predictive modeling, together with graphical user interfaces for easy accesses to this functionality. The Cluster section of WEKA includes the simple *k*-means algorithm, which can be applied to various data types. The main purpose of this project is to efficiently collect data from Twitter as well as re-create the *k*-means algorithm such that it produces similar results, and consequently, the most valuable and useful knowledge is obtained as a result of the clustering process.

**Introduction**

As highlighted earlier, information available on the internet is growing at a rapid rate on a daily basis. With over 340 million Tweets per day, the massive rise of Twitter can be viewed as an analogy for the growth of information on the internet. Using the Twitter API, we can gain access to the following user information:

* **Age of Twitter Account:** Fetches the ages of a Twitter account, using the date of its creation.
* **Number of Followers:** Fetches the Follower count of a specified user.
* **Number of Friends:** Fetches the Friend count of a specified user.
* **Number of Tweets:** Fetches the Tweet count of a specified user.
* **Number of Favorited Tweets:** Fetches the count of Tweets a specified user has favorited.
* **Listed Count:** Fetches number of lists a specified user appears on.
* **Last Tweets:** Fetches the last Tweets of a specified user.

**Reasons for Choosing Twitter**

Twitter is excellent source of information, as evidenced by over 340 million Tweets per day. It is open and a real-time communication of information among individuals and groups. Yet, under the surface, it is a treasure trove of knowledge about the behavior of its users, and trends at the local and global levels. People go to Twitter to share what they know and learn in return. Twitter users are hungry for new ideas, opportunities, information, services, and products. Businesses of all sizes use Twitter for a variety of reasons, from marketing to customer service. If your business is not part of this exchange, you're leaving two huge opportunities untouched: growing your business, as well as improving it. Using the mined data, companies better understand the customer demographic, thereby improving their marketing content. Companies can use their money wisely by appropriately tailoring their advertising to this knowledge. What most businesses tend to do is to hire a young, bright intern whose assignment will be to create a Twitter account to perform data mining the old-fashioned manual way. This manual approach is slow and very inefficient and a large amount of valuable information could be missed due to human error. Our motivation was to create an inexpensive solution to mine data out of the social networking site, which can be useful for any startup business. We have decided on Twitter as it is narrowed by its 140 character user status updates and not as complex as larger social media services, such as Facebook.

**Data collection**

To collect data from Twitter we have written a ruby program, which will collect data for us. This program makes use of the Twitter API which gives us a diverse array of functionalities in accessing user information. The created program takes in a single argument which specifies which user account to access. It then analyses the specified account and writes the abovementioned data to an output .CSV file type. This output file is then used as input for our data clustering process.

Data Miningis the computational process of discovering patterns in large data sets with an aim of extracting information from the data set and transforming it into an understandable structure for further use. Information, that is very valuable to businesses and organizations on a global scale. Data Mining did not emerged out of the blue. It arose from the evolution in information processing. This evolution led to a diverse range of data mining algorithms. In our project, we analyzed all the information mined from Twitter, by using data clustering.

**Data Clustering**

Clusteringis the process of grouping a set of objects in such a way that objects in the same group, or cluster, are more similar to each other than to those in other groups. It is a major part of the data mining process and is a common process in for statistical data analysis. It analyses data objects without consulting class labels, which may or may not exist. It generates labels for data as objects clustered based on a principle of maximizing intraclass similarity and minimizing interclass similarity. In our project, we used the *k*-means clustering algorithm which we explain below.

***k*-means Clustering**

The ***k*-means** algorithm is a distance-based clustering algorithm, which partitions all the data into a predetermined or fixed number of clusters (provided there are enough distinct cases). In our case, we have designed and programmed it such that it is able to discern the activity, or inactivity, of the followers of a specified Twitter account. Recreating the *k*-means algorithm to suit our needs is the main task of the project and provides the basis of our research as it groups the mined data and provides useful information. The *k*-means algorithm is one of many partitioning algorithms but is the one of the best and most common clustering methods. It operates in such a way that the number of clusters, *k*, is user-defined, while the dataset is also provided by the user. In summary, the algorithmaccepts only two inputs which are as follows:

1. The number of clusters (*k*): Simply, the number of partitions.
2. The input data set: The data to be partitioned.

These Similarities are measured in relations of the mean value of the objects, which represent cluster, and that can be consider of as the cluster’s centroid or point of attraction, that holds the similar data towards it and leaving out dissimilar objects/instances.

**How *k*-means Works**

This section goes into the details of how the *k*-means algorithm works:

* Clusters, which are formed, were represent by the **K.**
* Randomly the particular kingdom instances usually are been recently chosen for being original cluster centroids from your data set.
* All of those other circumstances tend to be given to each other of the cluster, which in turn is dependent on identical thing with all the cluster centroids.
* All the data which are similar and measured can compute and calculate as a distance functionality, which is Euclidean distance.
* An instance/object is assigned to a cluster to which it has the minimum distance or in other words minimum dissimilarity.
* Centroid is actually calculated in line with the not too long ago assigned physical objects as well as once more dissimilarity is actually calculated for each and every illustration while using the freshly calculated centroids.
* When the cluster centroid does not changes, then the re-distribution of instances get stopped
* Square-error-criterion has the convergence criterion.



Explanation of the above:

* E is known as error of the sum of the square in entire objects in the data set
* P defines the point in distance which is representing a given object;
* Mi isknown as cluster Ci(both p and miare multidimensional).
* Each and every single object inside each cluster, the space from the object to its cluster center is  squared, and the distances are summed.
* This whole principle put effort to make the result of K cluster as a compact and separate.
* For this project as per our research we have made used of Euclidean distance in our program, because we need to evaluate the dissimilarity measurement between the cluster centroids and instances. *Below is given the formula where P and Q represent as 2 instances :-*



**Mined Twitter Data**

Using data collection program, followed by our data clustering algorithm, the mined data is as follows:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No. of Instances** | **No. of Attributes** | **Instances in Class tested positive** | **Instances in Class tested negative** | **Data Type** | **Sets of Classes** |
| 192 | 7 | 98 | 94 | Numeric | Active / Inactive |

*k*-means is also implemented in the WEKA software. This section depicts a comparison of our implemented version of the algorithm versus the algorithm provided with WEKA:

Our Algorithm

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **SL. No** | **Seed value** | **Iteration** | **Time taken in second** | **NO. of cluster** | **Cluster #** | **No. of instances** |
| 1 | 10 | 3 | 0.138 | 2 | 24.47% | 47 |
| 75.52% | 145 |

WEKA

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **SL. No** | **Seed value** | **Iteration** | **Time taken in second** | **NO. of cluster** | **Cluster #** | **No. of instances** |
| 1 | 10 | 2 | 0.02 | 2 | 23% | 45 |
| 77% | 147 |

In summary, our algorithm is marginally less efficient with 3 iterations, as opposed to WEKA, which has 2 iterations. Due to this, our algorithm also clusters as a marginally slower pace. This indicates that WEKA is slightly more efficient than our algorithm. However, in terms of accuracy, the results are equal, with a deviation of less than 2%.

**Improvements**

Due to the nature of Twitter, outliers proved to be very common, with many users having extreme values for various attributes. For example, one user would have absolutely zero followers, but another user would have a massive following, that numbered in the thousands. We found that the *k*-means algorithm was very sensitive to these extreme values, or outliers. Initially, when we clustered the raw data, we got clusters which were affected by these outliers:

* Cluster 1: 96.3%
* Cluster 2: 3.6%

To solve this problem, one option was to ignore the outliers completely. However, that would render a lot amount of the data useless. Instead, we normalized the raw data before clustering them, which provided better results, and more in line with the results by WEKA:

* Cluster 1: 24.7%
* Cluster 2: 75.52 %

The nature of the *k*-means algorithm is that it is sensitive to the initial input value. For different initial values, there may be different clusters generated. To solve this, the first step is to get an initial seed points. To do so, randomly choose some subset within equal number of samples from large data sets. Then apply the partitional algorithm to each subset to get the center sets of the subset. Next, gather these centers and again apply partitional algorithm to obtain the most proper center set.

To find the optimum cluster, the optimum *k* value should be found. We can do this by using the encoding method. The value of *k* is the learning object to the genetic algorithm, the encoding it encoding to *k* value.

**Conclusion**

We would like to have used a more comprehensive version of the Twitter API. Unfortunately, the one we made use of had limited functionality in terms of accessing user data. In future, with more time, we consider making our own API to access this data. Moreover, we have also managed to solve the problem of outliers that plague our Twitter dataset, using the data normalization technique. As per our research, we have analyzed and implemented the *k*-means clustering algorithm, along with the normalization. In comparison to the WEKA algorithm we have managed to get similar results.