News Aggregator: Using K-means Clustering

Saurabh Pujar
Dept. of Computer Science, NYU
New York
ssp437@nyu.edu

Suruchi Sharma
Dept. of Computer Science, NYU
New York
sss665@nyu.edu

Abstract—

K-means clustering is a commonly used data clustering for unsupervised learning tasks. This method partitions a data set by minimizing a sum-of-squares cost function. A co-ordinate descent method is then used to find the local minima. However, successful use of k-means requires a carefully chosen distance measure most suitable for the task. Here we use Euclidean distance measure to cluster similar news stories gathered from different news sites.

Keywords **–** K – means clustering, crawling, news

I. INTRODUCTION

Clustering is an important data mining task employed in data set exploration and in other settings where one wishes to partition sets into related groups. Among the algorithms typically used for clustering, k-means is arguably one of the most widely used and effective clustering methods.

K-Means is based on the minimization of the average squared Euclidean distance between the data items and the cluster's center (called centroid). The results of the algorithm are influenced by the initial centroids. Different initial configurations might lead to different final clusters. Here, we randomly choose the initial configuration. The cluster's center is defined as the mean of the items in a cluster.

The internet has given rise to thousands of news websites each updated daily with latest events. Clustering of similar news articles poses a unique challenge in information retrieval. Different clustering methods are employed to cluster different forms of data. Our project is to build a news aggregator, which displays the top stories from multiple websites, using k-means clustering. Finding the appropriate number of clusters for a given data set is generally a trial-and-error process made more difficult by the subjective nature of deciding what constitutes 'correct' clustering.

II. RELATED WORK

A lot of successful news clustering websites are already in place. The most popular among them, Google News, uses a combination of different algorithms which have not been made public. For the purposes of this project, we found that k – means provides satisfactory results. Plenty of academic data is available on k-means clustering, many of them have been quoted in the references. We apply a basic k-means clustering algorithm and try and optimize it by modifying the value of k and the number of iterations.

III. ARCHITECTURE

The project is essentially a Dynamic web project in Java. The main components are front end, business logic and database. The business logic consists of crawling and clustering algorithms. A more detailed description of each component is given below. A war file of this project is deployed on sever.

1. <u>*CRAWLING:*</u>

a) Implementation:

Crawling is a part of the dynamic web project in java which includes clustering [5]. The crawling component of the project executes independently every three hours and begins executions as soon as the server starts. It refreshes the database tables with the latest news articles from seed links of news websites. In order to be selective about the links to be considered, we made a list of valid and invalid links. This was mostly based on observation by going through different news websites and trying to find a pattern in the links of news stories.

Crawling is primarily performed for two sets of seed links for two regional categories, India and World. We started out planning to retrieve news closest to the location of the user, but faced a lot of issues while crawling (highlighted below) which forced us to narrow down to just two regions. Each region has its own set of seed links which are crawled through alternately. Because the crawling functionality runs periodically as soon as the server starts, we show latest news stories from each region.

b) Issues:

Because of the unstructured nature of the news websites data, we faced a lot of challenges while crawling through our seed links.

1. *Avoiding click-bait articles:*
   A recent trend in online journalism is the popularity of click-bait news stories, which are

not really news stories. They tend to be in the form of slideshows, so we avoided all slideshows in the URL.

2. *Avoiding link to a list of news stories:*
   Many sections of a news website contain a collection of news stories belonging to a particular category. For example, the sports section of a website will contain links of different sports stories. We avoided such pages by not considering links which ended with terms like "sports/" or "technology/" or had word like "/articlelist/" in the URL.

3. *Avoiding links to comments:*
   All news websites we considered allowed users to post comments and had links to comments below the articles. Such links were avoided by not considering their URL pattern.

4. *Avoiding RSS feed/multimedia/other content:*
   To limit the scope of the project, we avoided RSS feeds and multimedia content.

5. *Fetching dynamically generated anchor tags:*
   The external library we are using to extract links, JSoup, fetches anchor tags that are statically present in the webpage and cannot fetch tags that are dynamically generated. This limits our choice of news websites and articles.

c) Scaling:

It turned out to be very difficult for us to scale our crawling algorithm, compared to our clustering algorithm. This was largely because of the esoteric nature on unstructured data on different news sites. This caused considerable limitations on the project. We had to restrict the number of news links, news websites and the regions of the news websites.

2. <u>CLUSTERING:</u>

a) Implementation:

We implemented the code for k-means clustering from scratch in java. Bag of words with Euclidean distance similarity measure was used. The Euclidean distance and Cosine similarity measures are given below.

Euclidean distance [10],

$$\mathrm{d}(\mathbf{p},\mathbf{q}) = \mathrm{d}(\mathbf{q},\mathbf{p}) = \sqrt{(q_1-p_1)^2 + (q_2-p_2)^2 + \cdots + (q_n-p_n)^2} = \sqrt{\sum_{i=1}^{n}(q_i-p_i)^2}.$$

Cosine Similarity [11],

$$\text{similarity} = \cos(\theta) = \frac{A \cdot B}{\|A\|\|B\|} = \frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n}(A_i)^2} \times \sqrt{\sum_{i=1}^{n}(B_i)^2}}$$

The data provided by the crawler consisted of links and page content, which included the title and keywords. The stop words were eliminated and dictionary was constructed for each link. We then created a global word list of all the links in the database. This global word list was then used to create a sparse dictionary of each individual links.

This sparse dictionary was the vector used for calculating the similarity measure. We initially assigned k random sparse dictionary values as our centroids and then formed clusters based on these random centroids. Then the centroid of each cluster was calculated and clusters reformed. Centroid formula is given below.

Centroid of k points [12],

$$\mathbf{C} = \frac{\mathbf{x}_1 + \mathbf{x}_2 + \cdots + \mathbf{x}_k}{k}$$

The algorithm is given below of k – means if given below.

```
K-means-cluster (in S : set of vectors : k : integer)
{  let C[1] ... C[k] be a random partition of S into k parts;
   repeat {
         for i := 1 to k {
            X[i] := centroid of C[i];
            C[i] := empty
           }
         for j := 1 to N {
            X[q] := the closest to S[j] of X[1] ... X[k]
            add S[j] to C[q]
           } }
    until the change to C (or the change to X) is small enough.
}
```

a) Experiments:

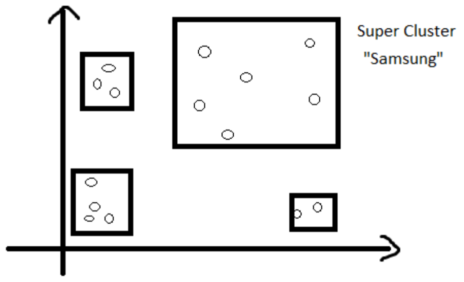
1. *Similarity Measure:*

We tried out two similarity measures, Euclidean distance and Cosine similarity. Euclidean distance provided better results consistently. Cosine similarity caused a majority of clusters to be empty and grouped many unrelated news stories in same clusters. For the same value of K, Euclidean distance led to less number of empty clusters and more well-formed clusters.

2. *Value of K:*

K corresponds to the number of different news stories which are currently present in the database. When some major news event takes place, like the Nepal Earthquake, most news outlets would cover this story and the ideal K goes down. On a slow news day, different news outlets will cover different stories which pushes the up value of ideal K.

As expected, for high volume of data, low value of K led to ill formed clusters. High value of K relative to the data, led to similar elements forming their own individual clusters with one element each. One problem we faced for relatively lower values of K was the formation of super clusters. Figure given below.



A super cluster was a combination of different un-related elements, which did not belong to any cluster, grouped together to form one very large cluster. Since we were naming the clusters based on whatever keyword appeared most frequently, these clusters were named after some random words which appeared very frequently in one news story. For e.g. a cluster in which one news article mentioned the word "Samsung" frequently led to a cluster named "Samsung" although the cluster contained 10 different news items which did not feature this word. To break this super cluster, we had to increase the value of K.

3. *Number of iterations:*

We are initializing the centroids with K random vectors from the dataset and then forming clusters. So object creation leads to rudimentary cluster formation.

Consecutive iterations led to better clustering of elements. But after a point, vectors did not move between clusters. So as per our observations, increased iterations give minor improvement in performance and then the improvement flattens off. This could be because of a local minimum [1]. This local minimum could be because of the initial points which are chosen at random. By choosing better initial points, we can arrive at a better local minimum. Much work has gone into refining initial points so that the best local minimum can be arrived at [2] but such techniques are beyond the scope of this project.

After many trials we arrived at an optimum number of iterations for data of given size and k of a particular value.

4. *Selection of words:*

As mentioned earlier, the crawler captured the keywords (metatags), News headline and the description of the news. We initially decided to consider only keywords for our vector formation since it seemed like the obvious thing to do. This led to ill-formed clusters because we discovered that following a major news event, like the Nepal Earthquake, many articles had "Nepal" and "Earthquake" in their keywords list, even though the article barely talked about these topics. We also found out that google does not recommend using keywords metatag [4].

We decided to then consider a combination of the news headline, description and keyword to form the bag of words vector. From this combination we deleted the stop words [3] and used the resultant vector for clustering. So if a news articles has Nepal Earthquake in its Headline, description and keyword metatag, it is more likely to be placed in the "Nepal Earthquake" cluster.

b) *Additional Data Structures:*

Because we did not apply any Natural Language Processing technique, other than bag of words, we ended up with a lot of similar clusters. For example, output was clusters like "Nepal", "Nepal Earthquake", "Earthquake", "Earthquake Nepal", "Nepal quake" etc. Since all these were news stories were from Nepal, they belonged in one cluster. We employed some string manipulation techniques and reformed the clusters to group such stories together.

We had to further reform the clusters because we had an over whelming amount of one news item and relatively fewer links of other news stories. We decided to restrict the number of links of one cluster to five while displaying on the user interface.

c) Final Output:

For the final output, we decided to display at the most twenty clusters (default five) with at the most five links in each. The user will be given an option to choose the number of clusters. Number of iterations was set at fifteen since beyond fifteen there was only a marginal impact on performance and high impact on response time.
Also, based on our trials, we received the best results for high values of k (k $\approx$ number of news links). We also restricted the size of the dataset. The code is such that all there variables can be manipulated with relative ease.

b) Database:

a. Tables:

1. *Seeds: India, World*
   This contains the seed links which will be used for crawling. The tables are static in nature. Since seed links are fixed, an update is not required.

2. *Records: India, World*
   This contains the links accumulated by the crawler and the relevant information for each link. This too gets truncated and updated every time crawling starts.

3. *Ip2_location_db1: Not used*
   This table is used to map ip-address to the location of the user [6]. It is static and is not updated. This was the part of the original functionality to get the location. We now use online services to get the location. This can be used later to precise location based news.

b. Connectivity:

We used MySQL database and connected to it using java JDBC. [5] Database was connected to while storing links during crawling and for getting links during clustering.

c) User Interface:

a. Framework:
HTML, jQuery, CSS, JavaScript, Bootstrap

b. Design:
1. jQuery accordion is used to display news links dynamically. The first section of the accordion is open by default displaying the contents of the first cluster. We can open any section (news cluster) by clicking on its name. This closes the current open section. At any point only the first section is open.
2. We have provided an input box at the upper left hand side to allow users to select the number of clusters to be displayed. This is not the value of K. The value of K is fixed and cannot be altered from the front end. This will only limit the number of clusters to be shown on the front end. This value has to be between 1 and 20. We have made introduced such as restriction as our parameters are set for these values.
3. Region selection option is provided at the top - center of the page below the web page header. The default region is based on the location of the user. If the JavaScript function is unable to decipher the region, or the user refuses location to the user, the default region is India.

IV. Results

We were able to obtain high precision in classification of documents. Around two or three out of about one hundred links tend to be misclassified on average. Time taken to load the links is a concern. It takes about eight to ten seconds to load the links. This is directly proportional to the number of iterations of the k – means clustering algorithm.

V. Future Work

1. Ranking Clusters: As of now, we are ordering the news clusters largest to smallest. We would like to incorporate some better ranking mechanism which would place latest or most trending news on top.
2. Ranking News links within clusters: Currently we are ranking links based on Euclidean distance to the centroid. We would like to rank links based on in links and out links.
3. Scalable Frameworks: Frameworks like hibernate would make the project more portable and structured while improving performance.
4. Improved Crawling: One of the biggest bottle necks in scaling a project like this is crawling. We would like to come up with better techniques for crawling that would improve scalability and performance.
5. Instead of deleting old news records, we plan to save it to an archive table. We can then add a search functionality which would allow users to search indexed data. We explored Apache Lucene to implement this in the current project but decided to not pursue it due to lack of time.

VI. Conclusion

The project was an excellent learning experience and allowed us to implement many concepts that we learnt in class. We realized the practical difficulties faced in implementing an idea for a web application from scratch. Apart from academic knowledge, a lot of practical knowledge was required in implementing the project. Creating a UI, using databases and servers is very useful to know. Creating a full stack web application can give one confidence to undertake similar projects in the future. Some of conclusions we drew by experimenting in this project were as follows.

1. Difficult to fix the value of k. Small k will cause vectors which are far apart to be grouped together. Large k will cause each vector to be grouped separately.

2. Subsequent iterations cause vectors to change groups but this eventually stops and we observe no change in cluster elements. This could be due to a local minimum that is attained.

ACKNOWLEDGMENTS

[1] Professor Ernest Davis

[2] The Open – Source community

[3] CIMS Support Team

REFERENCES

[1] http://ranger.uta.edu/~chqding/papers/Zha-Kmeans.pdf

[2] P. S. Bradley and Usama M. Fayyad. (1998). Refiningning Initial Points for K-Means Clustering. International Conf. on Machine Learning.

[3] Java api examples on stopwords at programcreek.com

[4] http://googlewebmastercentral.blogspot.com/2009/09/google-does-not-use-keywords-meta-tag.html

[5] http://www.programcreek.com/2012/12/how-to-make-a-web-crawler-using-java/

[6] http://lite.ip2location.com/database-ip-country#ipv4-mysql

[7] http://www.ee.columbia.edu/~dpwe/papers/PhamDN05-kmeans.pdf

[8] http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.218.5784&rep=rep1&type=pdf

[9] https://fedcsis.org/proceedings/2014/pliks/258.pdf

[10] http://en.wikipedia.org/wiki/Euclidean_distance

[11] http://en.wikipedia.org/wiki/Cosine_similarity

[12] http://en.wikipedia.org/wiki/Centroid

[13] http://ws.geonames.org/