CATEGORIZATION OF THE DOCUMENTS BY USING MACHINE LEARNING

Amol Jagtap
ME Computer Engineering, AISSMS COE
Pune, India
Email:[1]amol.jagtap55@gmail.com

***Abstract*—Machine learning is a scientific discipline that explores the construction and study of algorithms that can learn from data. Clustering is one of the applications of Machine Learning. Clustering is nothing but grouping data items together into classes as per the similarity among themselves. Data items within the class have high similarity in comparison to one another but are very dissimilar to data items in other class. The K-means algorithm is one of the widely used clustering algorithms. It is easy to implement and can be applied to wide variety of problems. But use of K-means was restricted to small datasets. Hadoop system supports Mapreduce which divides input into small inputs and operations are executed in distributed manner. It is inexpensive, scalable, free and open source which is why a promising technology for data intensive problems. Aim of this work is to use K-means algorithm for large dataset on Hadoop and then checking the performance of the same by increasing data and number of nodes.**

Keywords— Machine Learning, Document Clustering, K-means, Stemming, Term Frequency, Mapreduce, Hadoop

I. INTRODUCTION

Machine Learning is a subfield of computer science. It has strong ties to artificial intelligence. Machine Learning algorithms operate on building a model based on inputs and use that model for making predictions or decisions rather than following only explicitly programmed instructions.

Machine Learning can be divided into two parts:

1) *Supervised Learning*
2) *Unsupervised Learning*

In Supervised Learning training data is given from which model is built whereas in Unsupervised Learning model is built from Input Data only and no training is given prior. Machine Learning can be used for Classification, Clustering, and Recommendation. Classification is supervised learning whereas Clustering is example of unsupervised learning.

Document Clustering is a useful tool that can help the MNC's for processing large amount of documents. Unlabeled documents are grouped together into clusters as per their similarity. Document clustering can also be used for automatic topic extraction and for fast information retrieval or for filtering. Some interesting approaches to document clustering can be found in [2-5]. Survey on Document clustering applied to web engines can be found in [6].

Traditional approaches used for document clustering were applicable for only small and controlled datasets. In reality documents set is large and noisy. For example, Wikipedia contains set of documents acquiring 10 TB of storage. Internet also causing huge generation of documents. We can get benefit from these documents only when they are very well organized.

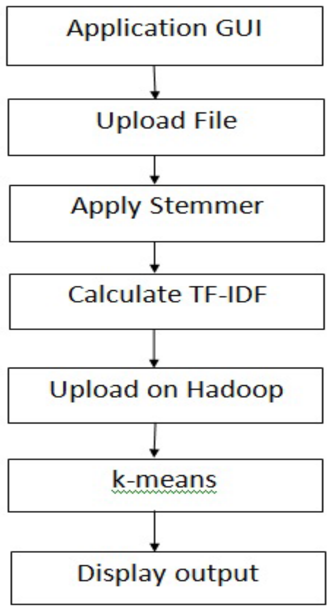
Clustering is divided into two methods. Hierarchical methods and Partitional methods. First group combines elements in sequence forming hierarchy structure whereas second group adds samples into group without creating any hierarchy. A deeper explanation is given in [7-8]. K-means is partitional method of clustering which is easy to implement and widely used.

As per the paper published by Abbas [9], partitional algorithms are best suited for large data sets whereas hierarchical algorithms are best option for small, data sets. Yadav, Sing [10] have found out that partitional algorithms complexity is less than the hierarchical one. Zhao [11] proved the same thing in the paper.

The K-means clustering algorithm is simple and has been used for solving different domain problems. In past use was restricted to small data sets only however in reality we find large amount of data to be processed which can not fit into main memory.

Apache Hadoop can be solution for this problem. However less work is done on performance of Hadoop framework. With this aim is to run K-means algorithm on large amount of documents on Hadoop and then studying the gain in performance caused by this solution. K-means algorithm is chosen here because it is most popular and easy algorithm.

II. SYSTEM ARCHITECTURE



III. THOERY

A. *Text Mining*

For clustering problems input samples should be vector format. Each part of the vector represents a feature or an attribute. Clustering algorithms then by use some distance measures find out distance between those vectors and the centroids (center of the cluster)

In case of document clustering, it is necessary to convert the text into numerical attribute vector format. For documents vector format is represented by using several dimensions. Dimension here considered as number of times certain word occurs in the document. It's obvious that dimensions of a single document can be very large. So we have to reduce them which is called as dimensionality reduction. One technique is removing the stop words which does not contribute as a document dimension or which does not specify document characteristics.

In document vector we find that weight reflects the feature distribution in the document. We should separate out words which are having more importance regarding the characterization of the document from words which are less important for characterization. A popular weighting scheme is the Term frequency – inverse document frequency (TF-IDF).

In Term Frequency we calculate the frequency of the word in the document comparing with total number of words in the same document. Whereas in IDF we calculate frequency of the word compared with same word in all documents. IDF is used for improvement.

TF(t) = (Number of times term t appears in a document) / (Total number of terms in the document).

IDF(t)= $\log_e$(Total number of documents / Number of documents with term t in it).

For Example, consider one document contains 100 words where word cricket appears 3 times. Therefore the term frequency (i.e., tf) of the word cricket is then (3 / 100) = 0.03.

Now, assume we have 1 billion documents and the word cricket appears in one thousand of these. Then, the inverse document frequency (i.e., idf) is calculated as log(10,000,000 / 1,000) = 4. Thus, the Tf-idf weight is the product of these quantities: 0.03 * 4 = 0.12.

There are also some more methods that can be for dimensionality reduction. They are as follows:

1. Filtering
2. Stopword removal
3. Tokenization
4. Stemming
5. Pruning

Stemming is nothing but keeping the word in base form and removing extra words. We stem or reduce words to base form. After applying stemming we will have less number of words (dimensions) to which we can assign TF-IDF values to find distance between centroid and samples. A stemming algorithm reduces the words "fishing", "fished", and "fisher" to the root word, "fish".

B. *K-means*

K-means algorithm partitions the samples into clusters by minimizing a distance between samples and centroid point of the cluster. There are many different distance measure available out of which Euclidean distance measure is widely used.

The K-means algorithm steps is given as:

1. Decide number of clusters k.
2. Initialize centroids (center of clusters) randomly c1, c2…..,ck.
3. For each point find closest centroid (by some distance measure) and assign the data point to that cluster.
4. Re-compute the centroids by taking means of all the point in cluster.
5. Stop when there is no change observed in the cluster (i.e. when it converges)

K-means clustering is having high complexity when data sets are large. Single machine memory can be a barrier for the performance. As a result k-means was not used for large data set in the past.

C. *Apache Hadoop Solution*

The Apache Hadoop software library is a framework that allows for the distributed processing of large data sets across clusters of computers using simple programming models. It is designed to scale up from single server to thousands of machines, each offering local computation and storage.

Hadoop uses Mapreduce which is a distributed programming model. Mapreduce has two user defined functions: map function and reduce function, specified on a job. Input job (here in our case document clustering) is split into multiple small jobs and stored in independent blocks which are then parallely processed by the map functions. Then Hadoop sorts those outputs generated by map tasks and then gives it as a input to reduce tasks.

Hadoop also uses distributed file system called as Hadoop Distributed File System (HDFS). In HDFS, data is stored in multiple copies which is decided by replication factor. By default replication factor is 3. This property is useful for availability of the data. Replicas are not stored on the same node which enable reliable, extremely rapid computations. Hadoop has two types of nodes: Namenode and Datanode. Namenode keeps track of data stored on the datanode. Without namenode Hadoop infrastructure is useless as there is no way to find which data is stored on which node. Data is stored on datanode with replication factor. Actual calculation is done on data node locally to reduce network uses.

However there is trade-off between overhead of map reduce and performance gain. As if data is less, then their will be less CPU utilization and more network utilization. Performance gain is less (even zero) in that case. Splitting job into multiple jobs and combining all outputs to from final output can be difficult for complex data sets.Hadoop need input file in specific format called as Sequence files. Sequence files are flat files consisting of binary key/value pairs. There are three types of the sequence files: uncompressed, record-compressed, block compressed. So documents need to be converted into sequence file format.

While executing K-means on the Hadoop cluster we have to provide following parameters:

-Sequence file containing input vectors.

-Sequence file containing initial cluster centers.

-Similarity measure to be used.

-Maximum iterations for longer jobs

-Number of reduce jobs

User gets output as centroid coordinates and the samples attributed to each cluster. Output is again stored in sequence file format.

There are three stages in a k-means job (given in figure):

1. Initial Stage:

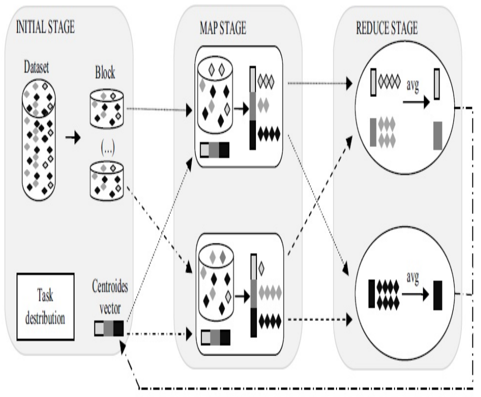
   Input data is segmented into HDFS blocks (each having size 64 MB), replicated and transferred to other machines (Datanodes)

2. Map stage:

   Distance between samples and centroid is calculated. Then data samples are assigned to nearest centroid and to cluster.

3. Reduce Stage:

   By taking mean of all the data samples in the cluster new centroid is calculated and again given these values to the mapper stage as a feedback. This loop ends when the centroid converges.



Once job is done we can find out time required to execute and quality of clusters.

IV. EXPERIMENT

Aim of the experiment is to find out time required to run K-means algorithm on large data set on Hadoop framework (Distributed), checking the performance while increasing the data and number of nodes and checking the trade-off effect between Mapreduce overhead and performance.

A. *Dataset*

News articles downloaded from the internet (news websites) up to 30 GB uncompressed. Then it is given for preprocessing (stemming, TF-IDF calculation). For the preprocessing we are using Java software.

B. *Nodes*

Initially just a single node and then number of nodes to compare the performance.

C. *Jobs*

To study how the performance changes with the scalability, jobs should be setup with the guaranty of convergence. Output depends on the centroids those are chosen initially. To avoid that effect same centroids should be used for single node setup and for multiple node setup.

V. EXPECTED RESULTS

1) As we increase the number of nodes time per iteration can be reduced.

2) Gain in the performance increases as we increase more number of nodes.

3) When we keep same amount of data and increase number of nodes, CPU utilization is reduced whereas network utilization is increased.

REFERENCES

[1] Rui Maximo Esteves, Chunming Rong, Rui Pais: ‘K-means clustering in the cloud- a Mahout test’, IEEE International Conference on Advanced Information Networking and Applications, 2011.

[2] T. F. Gharib, M. M. Fouad, and M. M. Aref, “Fuzzy Document Clustering Approach using WordNet Lexical Categories,” in Advanced Techniques in Computing Sciences and Software Engineering, Springer Netherlands, 2010, pp. 181-186.

[3] N. Kumar, V. Vemula, Vinay Babu, K. Srinathan, and V. Varma, “Exploiting N-Gram Importance And Additional Knowedge Based On Wikipedia For Improvements in Gaac Based Document Clustering,” *KDIR- 2010*, 2010.

[4] S. Banerjee, K. Ramanathan, and A. Gupta, “Clustering short texts using wikipedia,” in Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval, Amsterdam, The Netherlands, 2007, pp. 787-788.

[5] Pu Wang, C. Domeniconi, and Jian Hu, "Using Wikipedia for Coclustering Based Cross-Domain Text Classification," in Data Mining, 2008. ICDM '08. Eighth IEEE International Conference on, 2008, pp. 1085-1090.

[6] C. Carpineto, S. Osinski, G. Romano, and D. Weiss, "A survey of Web clustering engines," ACM Comput. Surv., vol. 41, no. 3, pp. 1-38, 2009.

[7] R. M. Esteves, R. Pais, and C. Rong, "K-means Clustering in the Cloud -- A Mahout Test," in Proceedings of the 2011 IEEE Workshops of International Conference on Advanced Information Networking and Applications, 2011, pp. 514-519.

[8] S. B. Kotsiantis , P.E.P.: 'Recent Advances in Clustering: A Brief Survey', WSEAS Transactions on Information Science and Applications, 2004, 1, pp. 73 – 81.

[9] Abbas, O.A.: 'Comparisons Between Data Clustering Algorithms', The International Arab Journal of Information Technology, Vol 5., No. 3, July 2008, 2008, 5, (3), pp. 320-325.

[10] Singla, B., Yadav, K., and Singh, J.: 'Comparison and analysis of clustering techniques', in Editor (Ed.)^(Eds.): 'Book Comparison and analysis of clustering techniques' (2008, edn.), pp. 1-3.

[11]YING ZHAO, G.K.: 'Hierarchical Clustering Algorithms for Document Datasets', Data Mining and Knowledge Discovery, 2005, 10, pp. 141-168.

[12] Tom White: 'Hadoop Definitive Guide' (O'Reilly Media, 2012,3rd ed.)