An Analysis of Data cleaning and Bisecting K-Means Algorithm for Clustering

Abstract: K-means algorithm is an iterative technique that attempts to split a dataset into K unique pre-defined non-overlapping subgroups (clusters), where each data point belongs to only one group. Bisecting k-means algorithm is a slight modification of k-means algorithm. The purpose of this research paper is to illustrate the Bisecting k-means algorithm and explain how it helps in performing Cluster analysis for Text clustering with News records for different cluster sizes and the steps required to clean the data before passing the data through the algorithm. Explore the Internal metrics that are utilised to evaluate accuracy of the mining algorithm.

I. Introduction

1.1 Definition

Cluster analysis is defined as finding groups of objects such that the objects in a group will be similar(or related) to one another and different from (or unrelated to) the objects in other groups. It is an unsupervised Machine learning algorithm that helps to understand the structure of the data set which acts as a tool to get better insight into the data distribution.

The main goal of clustering is to minimize Intra-cluster distances which elucidate that objects present inside the cluster are similar to one another and maximize Inter-cluster distances which elucidate that each cluster has objects which are not similar to one another.

1.2 k-means clustering

k-means is a partitional clustering algorithm where the number of clusters(k value) must be specified before starting cluster analysis. Each cluster is associated with a centroid(center/object) and each point in the cluster is assigned to the cluster with the closest centroid. But k-means suffers from some limitations and performs poorly when clusters are of differing sizes, densities and are of non-globular shapes. It also suffers if the data contains outliers.

1.3 Bisecting k-means clustering

Bisecting k-means is an improved and extended version of the k-means algorithm where the initial centroid problem can be avoided as it is less susceptible to initialization issues. This algorithm performs well k value is large. For the K-means algorithm, the computation involves every data point of the data set and k centroids. On the other hand, in each Bisecting step of Bisecting k-means, only the data points of one cluster and two centroids are involved in the computation. Thus, the computation time is reduced [2].

Bisecting k-means produce clusters of similar sizes, while k-means is known to produce clusters of widely different sizes.

I have performed clustering on input data News records. Input data (provided as training data) consists of 8580 text records in sparse format. It is a simple CSR sparse matrix containing the features associated with different feature ids and their counts in the input file. (Each line represents a document. Each pair of values within a line represent the term id and its count in that document). I have to assign each of the instances in the input data to K clusters identified from 1 to K.

II. Preprocessing techniques (Data Cleaning)

Preprocessing the data is highly important in data mining tasks because quality of the data effects the performance of the algorithm. In simple words, a data mining task can be divided into two parts - Data preprocessing(Data cleaning) and applying the mining algorithm on the clean data. Data cleaning involves various techniques such as removing null/missing values, reducing the dimensions, matrix factorization, normalization , binarization and many more.Following are some techniques which I have applied in order to maintain high quality data.

Before passing the data through the algorithm it needs to be preprocessed in order for the algorithm to produce accurate clusters and result in high clustering accuracy. The following are the steps that are performed :

1. Converting a list of documents in the dataset into sparse matrices.
2. Using TF-IDF methods to scale and normalize those matrices.
3. PCA technique for dimensionality reduction.

The data gets cleaned by the above techniques before performing clustering. All these steps are categorized into ‘Feature Reduction’ techniques.

2.1 Sparse Matrices(CSR matrix)

A matrix is sparse if many of its coefficients are zero. The interest in sparsity arises because its exploitation can lead to enormous computational savings and because many large matrix problems that occur in practice are sparse [3]. CSR stands for ‘Compressed Sparse Row’.

Building sparse matrix from a list of documents, where each of which is a list of words/terms in the document. Since only the non-zero values (or presence of data) are counted, and zero-values can be ignored in sparse matrices, the time taken to process sparse matrices is fast compared to dense matrices. This is the main reason for converting the News records into sparse matrices.

2.2 TF_IDF

TF-IDF stands for '**Term frequency**' and '**Inverse document frequency** which is a technique used to determine how relevant a word is to a document in a collection of documents. It quantifies words in a set of documents. It is formulated as:

TF-IDF = Term Frequency (TF) * Inverse Document Frequency (IDF)

This is used to define the importance of words in a set of documents. For text clustering, the obtained CSR matrix from the above step is scaled by IDF.

Normalization is a method of reducing the range of data dimensions to a narrower range, such as -1.0 to 1.0 or 0.0 to 1.0. This helps in processing the data effectively which increases the performance and takes less computing power. Here the CSR matrices are normalized by their L2-norm which is used to calculate the length of the obtained vectors after scaling. L2-norm which is also known as Euclidean-norm is calculated as the square root of the sum of the squared vector values.

2.3 Dimensionality Reduction

Dimensionality reduction is a technique employed in data mining preprocessing steps that reduce the number of attributes that describe the data points in order to eliminate irrelevant features and reduce noise. This process helps in reducing the amount of time and memory required by the data mining algorithms and makes data more visualized.There have been many techniques to perform dimensionality reduction and I have chosen to perform '**Principal Component Analysis**(PCA).

PCA projects a set of dimensions or attributes which describe the data points into a new set of attributes that is smaller and they do that by measuring the variability across different vectors and those vectors become a new set of dimensions.

The main reason for choosing PCA is because it takes the 'covariance' factor into consideration while reducing the dimensions in order to capture the intrinsic variability in the data. Covariance tells us how two variables vary together. Not only that, it can retain most of the information from the original dataset (variance) even after the reduction. The resulting principal components are orthogonal to each other which results in having unique information in each of the principal components.

III. Internal Evaluation Metric

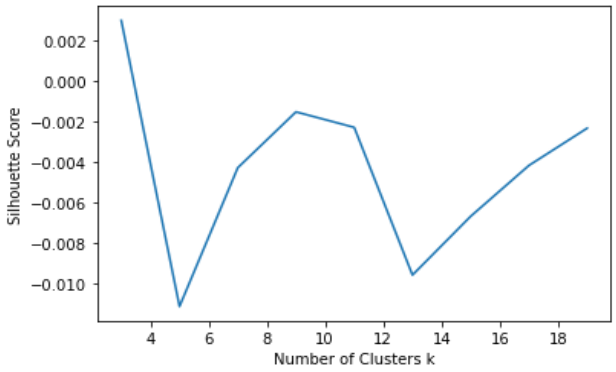
An internal evaluation metric is required to measure the cluster validity. It is a numerical measure applied to judge various aspects of cluster validity. One such measure is an 'Internal

Index' which is used to measure the goodness of a clustering structure without respect to external information.

3.1 Silhoutte coefficient

As an evaluation metric, Silhouette Coefficient is used. It is used to evaluate the quality of clusters that are created using bisecting k-means. It calculates the goodness of a clustering technique. Its value ranges from -1 to 1. The closer the value is to 1, the better the silhouette score. It combines the ideas of 'cohesion' and 'seperation' for both individual points and for clusterings. Cohesion measures how closely related are objects in a cluster. Separation measures how distinct or well-seperated a cluster is from other clusters.

Calculated Silhouette Coefficient score for number of clusters ranging from k = 3 to 21.



Plotted a graph with Silhouette Coefficient on x-axis and number of clusters on y-axis.

3.2 Sum of squared error(SSE)

I have also used '**Sum of squared error(SSE)'** as an internal evaluation metric. SSE is good for comparing two clusterings or clusters. Since, Bisecting k-means is being used in this usecase, SSE acts as an optimal evaluation metric technique here. SSE is sum of squared differences between each observation and its group's mean. It is used as a measure of variation within the cluster. The higher the SSE value, the greater is the variation within the cluster. By calculating the SSE we know the distance between the points in the cluster and know at which point to divide the cluster.

$$SSE = \sum_{i=1}^{n} (x_i - \bar{x})^2$$

n- is the number of observations , x_i is the value of the ith observation, $\bar{x}$ is mean of all observations.

IV. Bisecting k-means algorithm

Bisecting k-means is a special type of k-means algorithm which is used to find for cluster analysis. It is a type of partitional clustering algorithm which bisects the cluster into two clusters.

In this algorithm, we can start with choosing our own k (number of clusters) value. I have chosen k value as 7. We first take the whole cluster as one cluster and calculate the SSE value.

SSE calculates distance between every point in the cluster and returns the id value of a point which represents the point at which the bisecting should happen.

K-means divides the cluster into two clusters at that point by calculating centroids. Centroid is the mean of the values of the points of data in the cluster. The cluster with the highest SSE value is again calculated for SSE value and performed k-means for further clustering and the process continues till the specified k value.

4.1 Algorithm

1. **Initialize** a list of clusters to contain the cluster containing all points.
2. **repeat**
   select a cluster from the list of clusters
   > for i:=1 to number_of_iterations do
   Bisect the selected cluster using basic k-means
   end for
   > Add the two clusters from the bisection with the lowest SSE to the list of clusters.
   until Until the list of clusters contains k clusters

4.2 Logic

k -> number of clusters n_iter -> number of iterations

Bisecting k-means program start

> Initialize initial_cluster = list()

> Add initial_cluster to clusters_list

> Start while loop(condition : length of clusters < k)

clusterId_that_should_be_removed = calculate SSE(matrix, clusters_list) and find the point with highest SSE value

clusterId_that_is_removed = clusters_list[clusterId_that_should_be_removed]

Using k-means the cluster is divided into two clusters - Cluster 1 and Cluster 2

Cluster1, Cluster2 = k-means(matrix[from clusterId_that_is_removed till last index], n_iter)

Delete clusterId_that_should_be_removed from clusters_list

Two clusters returned by k-means is appended to the real(original) cluster.

Examine index in Cluster1 and

Append clusterId_that_is_removed[index] to real_cluster1

Examine index in Cluster2 and

Append clusterId_that_is_removed[index] to real_cluster2

Append real_cluster1 and real_cluster2 to clusters_list
(continue bisecting cluster until the k value specified)

Here, we can write our own k-means or use an external library to perform k-means.

The following are some silhouette scores for different cluster sizes produced by the algorithm:
For K= 3 silhouette_coefficient Score is 0.002913
For K= 5 silhouette_coefficient Score is -0.011231.

We can see that for k=3 the silhouette score is greater making it the best the optimal number of clusters for cluster analysis.

Accuracy of the algorithm can be evaluated through external index metric like Normalized Mutual Information(NMI) which produced a 0.6080 accuracy for the algorithm. The closer the accuracy is to 1, the better.

V. References

[1] C. Aggarwal C. and C. Reddy K., *Data Clustering*. CRC Press, 2018.

[2]. Ristoski, Petar, Christian Bizer, and Heiko Paulheim. "Mining the web of linked data with rapidminer." Web Semantics: Science, Services and Agents on the World Wide Web 35 (2015): 142-151.

[3]. I. Duff S., A. Erisman M., and J. Reid K., *Direct Methods for Sparse Matrices*. Oxford University Press, 2017.