PR1 Report for CMPE 255 - Text Clustering using Bisecting K-means

Name: Sandesh Gupta
SJSU ID: 015649036

Overview and Assignment Goals

The objectives of this assignment are the following:

- Implement the Bisecting K-Means algorithm.
- Deal with text data (news records) in document-term sparse matrix format.
- Design a proximity function for text data.
  - Think about the Curse of Dimensionality.
- Think about best metrics for evaluating clustering solutions

Input

train.clabel: Contains the unique labels used in train.dat file. Each line indicates the word and the index(line number) association.

train.dat: 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)

Output

output.dat: Contains the cluster labels alloted to each document in a new line

Approach

- Loading CSR matrix:
  - The train.data consists of the 8580 documents. There are 126373 distinct words in train.clabel which are used in the documents. We can create a matrix of 8580*126373 size with each cell representin the frequency of the word in each document.

- Pre-processing:
  - Reduce columns of the matrix:
    - One observation is that not all 126373 words present in clabel file are used in the documents. There are only 27673 unique words present in the overall actual data. Hence, our CSR matrix can be reduced to size 8580 * 27673

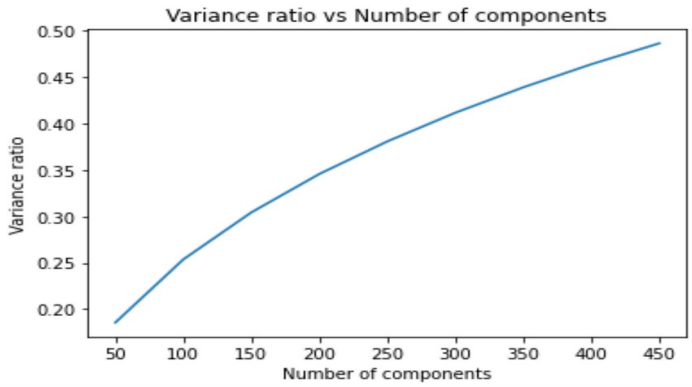
  - Removing words with size <= 3:
    - Words with character length 3 and below are removed from consideration as they can influence the mining because of it's high frequency.

  - Stop-words removal:
    - Stop-words like 'a', 'an', 'the' which don't provide much value to the document's content are filtered out. NLTK's english stop-word library is used to perform this.

  - Tf-idf transformation:
    - Tf-idf is the product of TF(Term Frequency) and IDF(Inverse Document Frequency). TF is basically the importance of the words in document, where IDF is the importance of words in the whole corpus of documents.
    - Tf-idf transformation is applied to the matrix using the sklearn TfidfTransformer to achieve normalisation.

After applying various pre-processing techniques, we got the tf-idf normalised matrix with size 8580 * 26237

- Curse of dimensionality
    - We have 26237 dimensions, which is very large. To reduce the dimensionality**,** **Truncated SVD** is applied, also known as Latent Semantic Analysis(LSA). PCA was not chosen as it has to center the data before computing SVD, which would be not be efficient with sparse matrices.
    - Scikit learn recommends the number of components be 100 for LSA.
    - The variance ratio sum for different number of components is as shown as below:



    - I checked for multiple values of components from range 100 - 8000. The accuracy on the CLP portal were around 50-60% NMI for components < 100. When components > 2000 is chosen along with good K no of clusters, the accuracy improved with max being 67.87% NMI.

- Bisecting K-means:
    - Bisecting K-means is a hybrid approach of K-means and Hierarchical Clustering.
    - In each iteration of the algorithm, it splits the data into 2 clusters. The data with higher SSE is further split into 2 clusters. This process is repeated until desired number of K is reached.
    - K-means method call is used from the scikit library.
    - **Pseudo-code:**

```
#Runs Bisecting K-means for different values of K in range k_start, k_end with increments of step
BisectingKMeans(matrix, k_start, k_end, step):
        for k in range(k_start, k_end+1, step):
                current_clusters = 1
                while current_clusters != k:
                        kmeans(k=2)
                        cluster_1 = cluster with minimum SSE
                        cluster_2 = cluster with maximum SSE
                        Assign label(current_cluster) to points in  cluster_1
                        matrix = cluster_2 #run next iteration on this cluster
                        current_clusters++
```

- Metrics

Since there were limited number of submissions on CLP portal, metrics helped me to identify the best cluster combination. Both the metrics are used which doesn't need true labels to score.

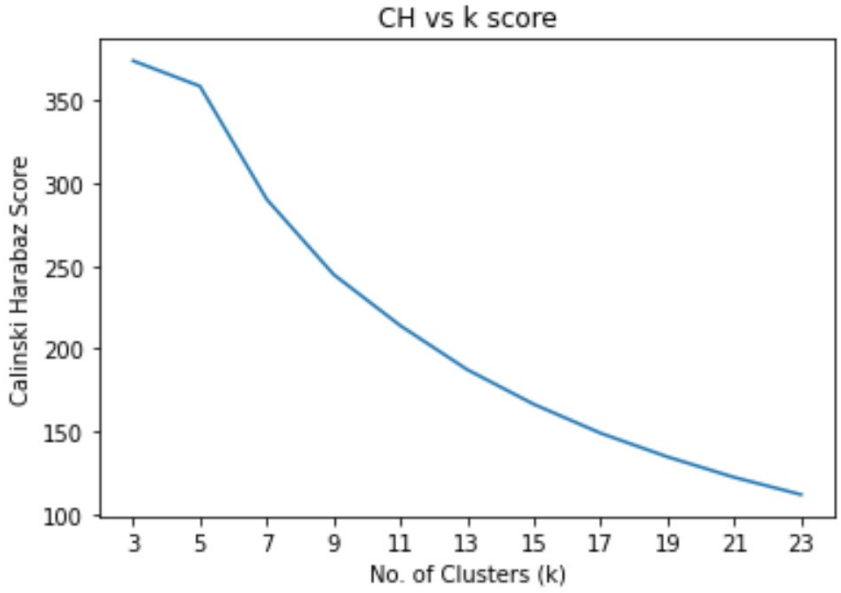
  - Silhoutte Score: This score measures the seperation distance between clusters. It has a range of [-1, 1].

  - Calinski Harabasz Score: This score is the ratio between within-cluster and between-cluster dispersion.

For different values of K [2,22] and Components(SVD) [100 - 8000], I found scores in range [-0.06, 0.02]

- Plotting graphs

Graph for Calinski Harabasz Score with k [3, 23] with step 2



- Improving the accuracy:

I tried to improve the accuracy by changing different values for K [2,22] and Components(SVD) [100 – 8000]. Some of the analysis of multiple values can be found at this Pastebin link.

Learnings and future opportunities:

- The documents could have been stemmed to merge words like realise, realised, realising to realise
- Should have stuck to less number to components for SVD (under 200s)
- Should try to verify output with other algorithms
- More plots. Visualisation helps

References:

https://en.wikipedia.org/wiki/Tf%E2%80%93idf
https://towardsdatascience.com/performance-metrics-in-machine-learning-part-3-clustering-d69550662dc6
https://scikit-learn.org/stable/index.html
https://link.springer.com/article/10.1007/s10586-018-2084-4