Natural Language Processing Project: Text Clustering

Sudat Tuladhar
Electrical Engineering

I. Introduction

Through this project, we get familiar with text clustering, cluster comparison, and evaluation. Two clustering algorithms K-Means and DBSCAN are implemented and compared with reference to NSF Abstract dataset.

Fig. 1 depicts the overall process of the project.

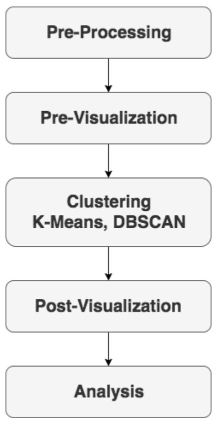


Fig. 1: Overall Process

II. Pre-Processing

A. Data Extraction

Unique identifier 'AwardNo' and abstract content is extracted from every file of NSF grant dataset: Part-1.

B. Document Filtering

All documents with empty/null/Not-Available abstracts are filtered out. Finally 51715 number of documents are chosen for processing.

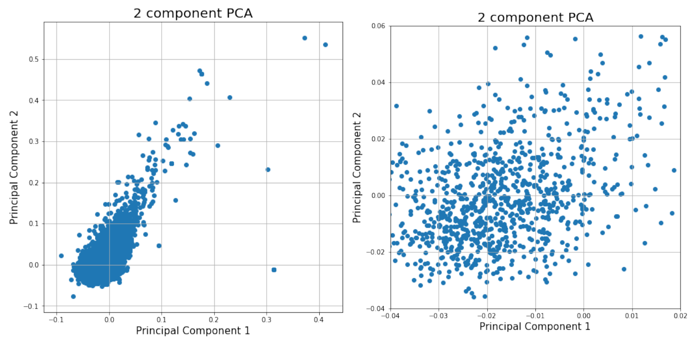
C. Document Vector

Word2Vec library is utilized to convert each word to a vector. Vector corresponding to a document is the average Word2Vec of all words that are present in the document and Word2Vec library. Each document vector is a 300 dimension.

'docs_vector.csv' is a pre-processed file containing 51715 documents each of 300 dimension.

III. Pre-Visualization

Fig. 2 shows the all data n = 51,715 and sample data n = 1000 after 2-component principal component decomposition. First principal component retains 64.4% and second principal component retains 6.1% variation of total data.



(a) All Data, n = 51715 (b) Samples, n = 1000

Fig. 2: Data Visualization with PCA

IV. Clustering: K-Means

K-Means clustering is implemented with different values of K = 5, 7, 10, 12, 15. Python pandas data-frame is used as a data structure to hold the document vectors and perform grouping and vectorized operations.

In the implementation, initial cluster centers are initialized with random data points. The label of each data point is updated based on the cosine similarity between the point and cluster centers. Cluster centers are calculated by grouping the data points based on the label and then performing vectorized averaged operation. Finally Sum of Squared Error (SSE) is calculated using euclidean distance between final centers and documents in the cluster.

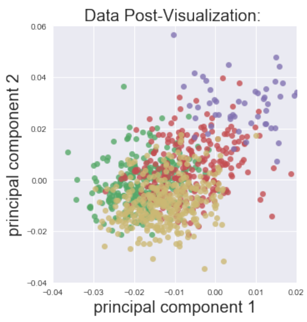
The distribution of documents in different clusters is shown in Fig. 3. SSE for different values of K is shown in Fig. 4. The SSE decreases with increase in number of clusters as expected. The distribution of data with annotated unique document identifier 'awardNo' is shown in Fig. 5.

Documents with awardNo = 9000157, 9000154 and 9000130 seem to be categorized into same cluster from Fig. 5. The meta-data for these documents were found to be from same field application: Oceanography.

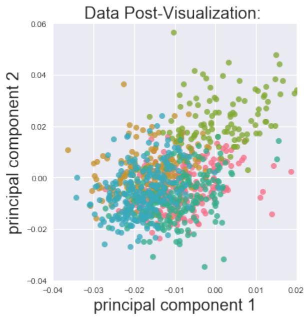
V. Clustering: DBSCAN

The optimized parameter of minPts = 4 and eps = 0.05 for DBSCAN was found using K-distance graph in Fig. 6. Python data-frame is used as a data structure to hold the document vectors and calculate euclidean distance between data points.

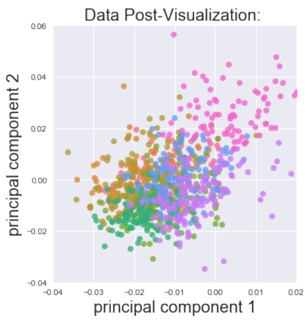
In the implementation, all documents are first labeled 0 for unmarked status. For unmarked points, the points which have



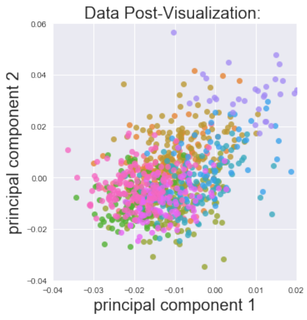
(a) Data, K = 5



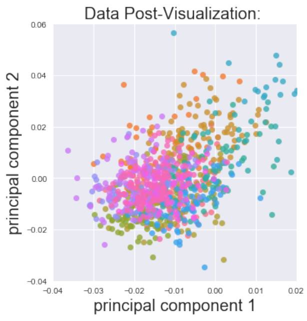
(b) Data, K = 7



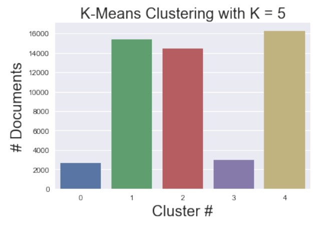
(c) Data, K = 10



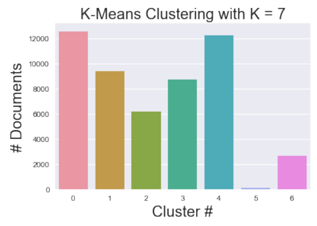
(d) Data, K = 12



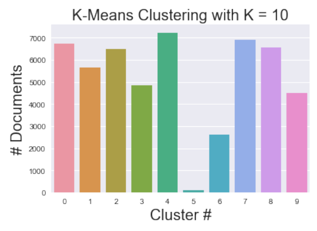
(e) Data, K = 15



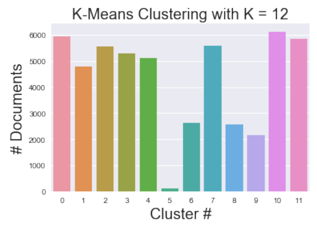
(f) Data Count, K = 5



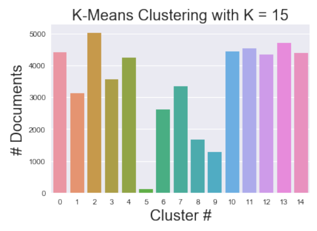
(g) Data Count, K = 7



(h) Data Count, K = 10



(i) Data Count, K = 12



(j) Data Count, K = 15

Fig. 3: Post-Visualization of K-Means

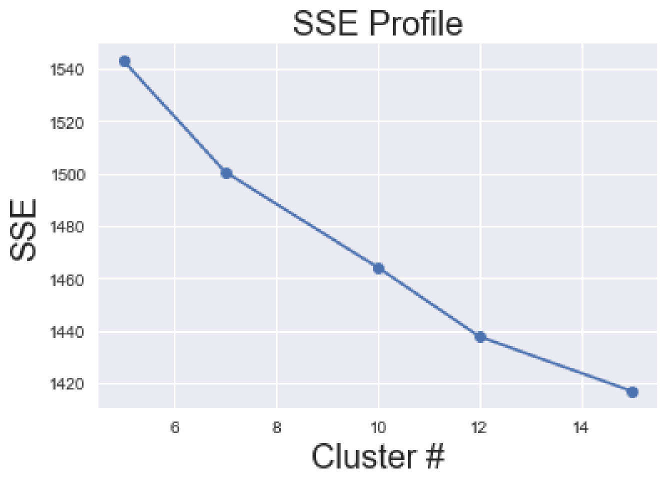


Fig. 4: K-Means SSE Profile

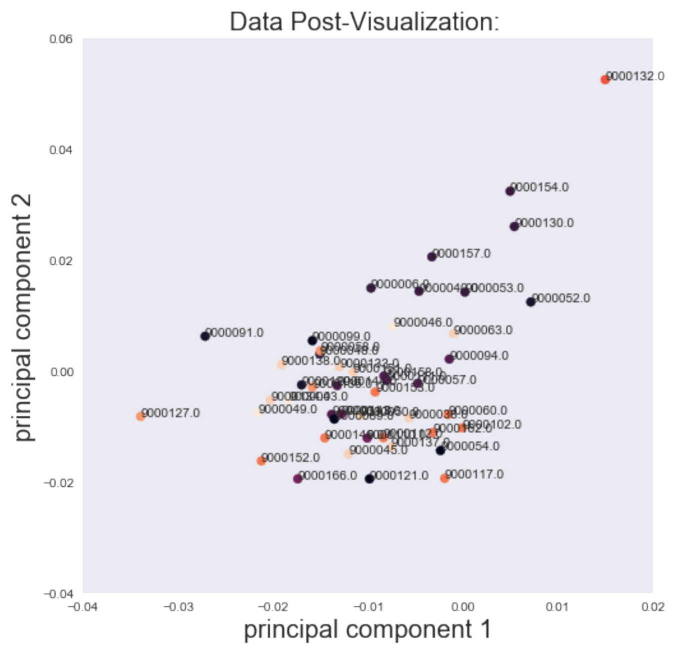


Fig. 5: Annotated Data , K = 15

insufficient data points within specified radius determined by minPts and eps parameter of DBSCAN are marked as noise points. For other points, a depth-first search is carried out at every point reachable from the given point determined by minPts and eps parameter. A new cluster is created whenever core point cannot be reached from other core point.

The current implementation is simple and thus does not use accelerating index structure for neighborhood query. Because of this the time complexity of the implementation is $O(n^2)$ instead of $Nlog(N)$. Since the data points N = 51715 is large, the computation for DBSCAN algorithm under different parameter settings were carried out using MCSR resources.

DBSCAN clustering with the parameters (minPts, eps) = [(4, 0.03), (4, 0.05)] resulted in clustering as shown in Fig. 7.

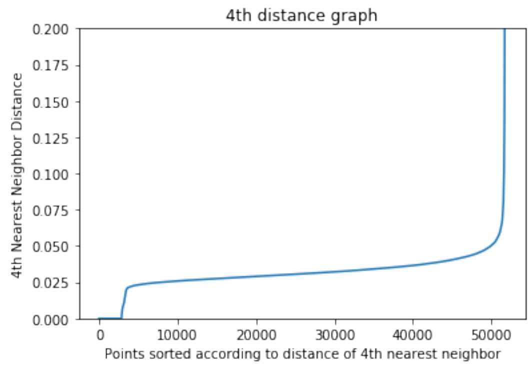


Fig. 6: 4-distance-graph

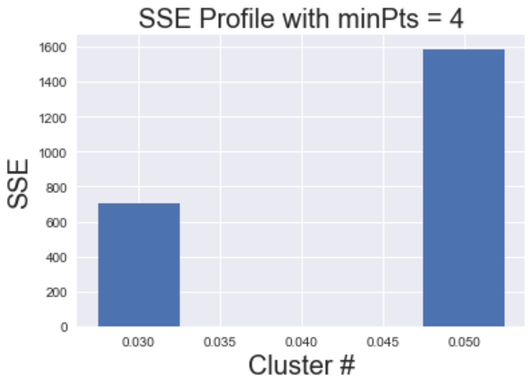
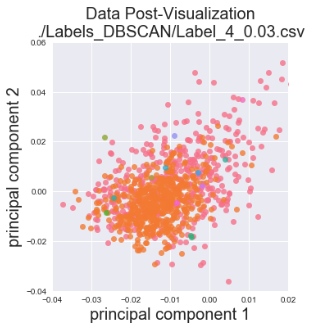
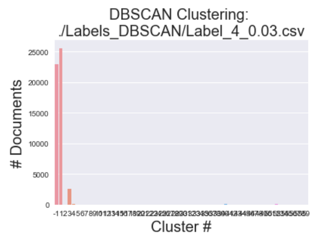


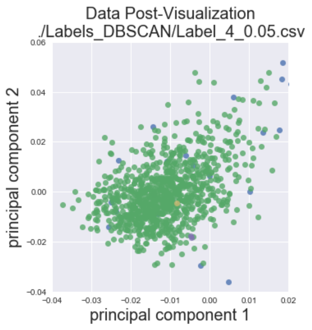
Fig. 8: 4-distance-graph



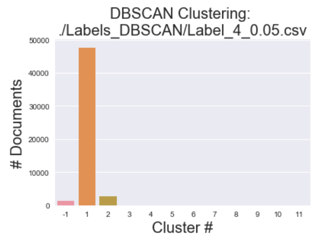
(a) PRM: (4, 0.03)



(c) PRM: (4, 0.03)



(b) PRM: (4, 0.05)



(d) PRM: (4, 0.05)

Fig. 7: Post-Visualization of DBSCAN

SSE for DBSCAN is shown in Fig. 8. Because of parameter sensitivity, either many documents are classified as outliers as in the case (4, 0.03) or almost all documents are clustered into a single cluster as in the case (4, 0.05).

VI. Analysis

Even though the SSE of DBSCAN as shown in Fig. 8 is comparable to SSE of K-Means as shown in Fig. 4, the distribution of documents in each cluster for K-Means seems much better than for DBSCAN as shown in Figs. 3 and 7.

VII. Conclusion

For the current dataset, K-Means seem to work better than DBSCAN based on the distribution of documents in different clusters. This is because DBSCAN does not perform so well in high-dimensional data. Because of time limitation, DBSCAN clustering under reduced dimensionality could not be tested.