HW3 Part 1:Text clustering

Hsin-Ping Hsu (G01167652)

*Hung-Mao Chen (G01145163)*

* Name registered on miner website: djmp
* Rank and score: 7/ 0.67
* Approach:

1. Read feature and test data then revert the sparse matrix: read the features data test data. Combine them into a 2D array; therefore, the array size is 8580 \* 126373.

Function: read\_data()

1. Preprocess the data via TF-IDF: utilize the TfidfTransformer from sklearn to preprocess the data. By transforming the matrix to TF-IDF, features will be weighted, and the values of some features would be 0.

Function: tf\_idf()

1. Perform SVD for dimension reduction: Since this is a sparse dataset, we reduce the dimension through SVD from sklearn, and set the n\_components = 3000 so the fraction of total variance is about 0.85.

Function: svd(test\_tfidf)

1. K-means clustering:
   1. Initial centroids: select 7 data points randomly by applying k-means++ algorithm to avoid empty cluster issue.
   2. We computed cosine similarity, correlation, and Chebyshev distance between each point and centroid, then assign the cluster with the centroid that is closest to the data point. We observed that measuring the correlation might be a better option for this dataset.
   3. After assign cluster to each data point, recompute and update the centroids then repeat a and b steps.
   4. Set the maximum iteration of updating centroids (running a and b) as 300.
   5. Calculate the silhouette score and SSE.
   6. Repeat a to e 10 times, and return the best clustering result with the smallest SSE and highest silhouette score.

Function: k\_means(test\_svd), k\_means\_plus\_plus(test\_svd, K), get\_correlation(v1,v2, get\_cosine\_sim(v1, v2), get\_chebyshev\_dist(v1,v2), sil\_score (data, predicted), compute\_sse (test\_svd, clusters, centroids)

For the step f, we record the results as the tables

If we use cosine similarity as my distance function:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| iter 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| Sil 0.038 | 0.037 | 0.035 | 0.034 | 0.039 | 0.038 | 0.039 | 0.034 | 0.037 | 0.031 |
| SSE 7001 | 7016 | 7039 | 7053 | 7002 | 6994 | 7007 | 7014 | 7020 | 7040 |

Second row are silhouette score, and the third row are SSE.

In iteration four, we can get the maximum silhouette score 0.0397728, and in iteration five, we can get the minimum SSE which is 6994.34.  
If I use correlation as my distance function:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| iter 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| Sil 0.04 | 0.039 | 0.035 | 0.038 | 0.04 | 0.04 | 0.04 | 0.033 | 0.044 | 0.035 |
| SSE 7005 | 7010 | 7016 | 7001 | 6992 | 7000 | 6984 | 7018 | 7004 | 7030 |

Second row are silhouette score, and the third row are SSE.

In iteration eight, we can get the maximum silhouette score 0.0442088, and in iteration six, we can get the minimum SSE which is 6984.90.

With the results above, we apparently find that using correlation as our distance measures can get higher silhouette score and lower SSE, this is the reason that we consider correlation a better option for this dataset.

* Evaluate K-means clusters with SSE:

Fig. 1 shows the SSE curve with the value of K increasing from 3 to 21. We observed that increasing the value of K can reduce the SSE; however, a good cluster can have a low SSE with small K.

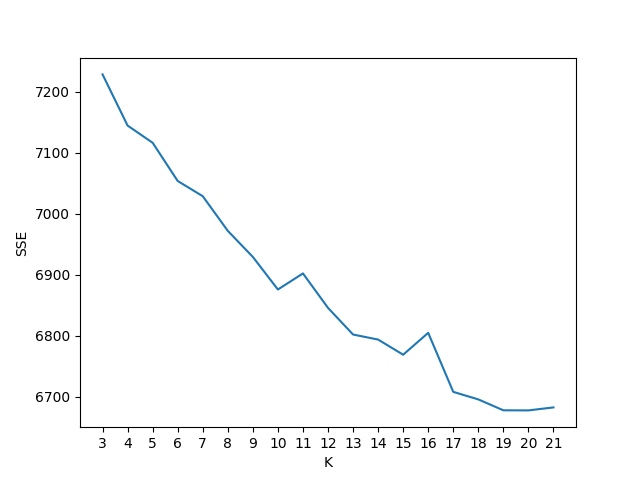


Fig. 1 SSE curve

**References:**

1. <https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfTransformer.html>
2. <https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.TruncatedSVD.html>
3. <https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html>
4. <https://stackoverflow.com/questions/5466323/how-exactly-does-k-means-work>
5. <https://scikit-learn.org/stable/modules/generated/sklearn.metrics.silhouette_score.html>
6. <https://arxiv.org/pdf/1405.7471.pdf>