COMP 41450
Non- negative Matrix Factorization

Saransh Agarwal
14203485
05 December 2014

1 Problem Description

A new implementation of the Euclidean distance formulation of Non- negative Matrix Factorization as proposed by Lee & Seung [1]. The implementation is in the form of a sparse term- document matrix.

2 Implementation

The implementation of above stated problem is performed on Java SE 8 using Eclipse Luna IDE on sample data files (`bbcnews.mtx and bbcnews.terms`). Jama [2], a package for matrix algebra is used as reference library. Following were the tasks performed on for this assignment:

1. Reading of sparse term- document matrix from bbcnews.mtx –
   To read matrix from file, a method `read( )` is invoked in `matrixRead.java` file. The method splits the matrix in orders of terms, document and frequency to store it in separate numeric variables.
   Reading terms from bbcnews.terms –
   To read terms from file, a simple I/O function is implemented in `readTerms.java` file.

2. TF- IDF Normalization –
   TF- IDF is the product of two statistics, term frequency and inverse document frequency [3]. Term frequency tf(t,d) : The number of times a term t occurs in document d. I've used an augmented frequency formula here:

$$\mathrm{tf}(t,d) = 0.5 + \frac{0.5 \times \mathrm{f}(t,d)}{\max\{\mathrm{f}(w,d) : w \in d\}}$$

   Inverse document frequency idf(t,D): It is a measure of whether the term is common or rare across all documents. The formula used for IDF is:

$$\mathrm{idf}(t,D) = \log \frac{N}{|\{d \in D : t \in d\}|}$$

   Then, tf-idf is calculated as:

$$\mathrm{tfidf}(t,d,D) = \mathrm{tf}(t,d) \times \mathrm{idf}(t,D)$$

These functions are implemented in tf_idf( ) method by calculating frequency of terms, document with specific terms and using inbuilt library functions like Math.log( ) to calculate the resultant normalized term- document matrix.

3. Initializing factors for NMF randomly –
Initialization of factors W (Basis Vector) and H (Coefficient Matrix) is done through an inbuilt Math function `public double nextdouble( )`. It returns the next pseudorandom, uniformly distributed double value between 0.0 and 1.0. The dimensions of matrix factors W and H are $n \times k$ and $k \times m$, respectively. The rank k of the factorization is chosen between 2 to 6 for the given problem.

4. Applying Euclidean NMF –
The Euclidean NMF is described by updating factors W and H respectively until maximum number of iterations.
The factor H (Coefficient Matrix) is updated as:

$$H_{cj} \leftarrow H_{cj} \frac{(W'A)_{cj}}{(W'WH)_{cj}}$$

The factor W (Basis Vector) is updated as:

$$W_{ic} \leftarrow W_{ic} \frac{(AH')_{ic}}{(WHH')_{ic}}$$

In Java, the matrices are updated in methods w_new( ) and h_new( ). Functions from Jama package were used like `public Matrix times()` and `public Matrix transpose()`.

5. Top terms in each cluster –
To achieve top terms in each cluster, the index of terms from factor W[n,k] is chosen and stored in an integer array. Further, they are sorted in ascending numerical order using `public static void sort (double[] a)`.

3 SUMMARIZATION OF OUTPUT

Below shows output for Euclidean NMF when applied to *bbcnews* dataset. The output shows top 10 terms of each cluster, for k=2 to k=6 clusters.

1. For k=2

Rank	Cluster1	Cluster2
1	company	secretary
2	2005	think
3	industry	told
4	companies	just
5	2004	before
6	million	got
7	analysts	right
8	firm	labour
9	growth	against
10	market	bbc

2. For k=3

Rank	Cluster1	Cluster2	Cluster3
1	oil	issue	mobile
2	market	side	research
3	bank	told	net
4	shares	prime	internet
5	prices	minister	online
6	analysts	election	service
7	rise	government	music
8	economic	against	video
9	economy	secretary	users
10	growth	labour	technology

3. For k=4

Rank	Cluster1	Cluster2	Cluster3	Cluster4
1	plans	web	oil	play
2	issue	research	market	final
3	spokesman	video	bank	chance
4	public	service	shares	second
5	prime	net	economic	nations
6	election	internet	prices	first
7	minister	online	analysts	ireland
8	secretary	music	rise	got
9	labour	users	economy	six
10	government	technology	growth	side

4. For k=5

Rank	Cluster1	Cluster2	Cluster3	Cluster4	Cluster5
1	company	first	general	service	rose
2	filed	second	public	research	market
3	action	match	chancellor	broadband	bank
4	bankruptcy	chance	plans	net	shares
5	protection	final	prime	phone	economic
6	rights	nations	government	mobile	analysts
7	case	ireland	secretary	music	prices
8	law	got	minister	video	rise
9	legal	six	election	users	economy
10	court	side	labour	technology	growth

5. For k=6

Rank	Cluster1	Cluster2	Cluster3	Cluster4	Cluster5	Cluster6
1	tax	secretary	chance	demand	filed	phone
2	government	legal	second	dollar	russia	web
3	general	legislation	france	rates	assets	service
4	gordon	court	match	rate	russian	internet
5	brown	rules	nations	rose	khodorkovsky	net
6	minister	lord	final	economic	firm	online
7	prime	case	ireland	prices	unit	video
8	chancellor	laws	got	rise	yukos	music
9	election	rights	six	economy	company	users
10	labour	law	side	growth	bankruptcy	technology

4 DISCUSSION

As the results show, the values in the cluster differ for different values of k. In each of the cluster, the algorithm has produced the top terms such that they are related to each other. For example-

1. For k=2, cluster 1 highlights 'economy', but it is not very clear. Cluster 2 is not producing any specific topic that is related to most of the terms in the cluster.
2. For k=3, in cluster 1, term 'economy' appeared that makes some sense with the corresponding terms in the cluster like 'shares', 'prices', 'growth', 'rise', 'bank'. Meanwhile, cluster 2 shows no correlation between terms. There is a sparse connection to 'election' while looking at the words 'issue', 'prime', 'minister', 'government'. Cluster 3 highlights 'technology' with other related words being 'mobile', 'online', music', 'video', 'internet'.

3. For k=4, there is a weak relation with other terms and `'election'` seems to be the common. Cluster 2 show casts `'internet'` in the same manner as in k=3. Cluster 3 emphasize on `'economy'` with related terms. Cluster 4 does not give a clear picture, but the topic seems to be `'play'`.
4. For k=5, there is a strong correlation between terms in clusters. Cluster 1 describes `'legal'`, with relation to `'court', 'case', 'law', 'rights'`. Cluster 2 focusses on `'match'` with sparse connections with `'final', 'side'`. For Cluster 3, the group is formed by `'election'` concerned to terms `'public', 'plans', 'government', 'minister', 'labour'`. Cluster 4 is similar to previous values of k in which `'technology'` is related to `'broadband', 'phone', 'mobile', 'net'`. Cluster 5 gives connection among words that relates to `'economy'`.
5. For k=6, there are similar results as for k=5, apart from extra cluster terms in 5th cluster, that gives no clue of the meanings.

5 Conclusion

The results produced by NMF were <u>comparable</u> easily. In text mining, like the one performed here on *bbcnews* data, the algorithm uses context for differentiating other terms. For example, the word `'technology'` was easily related to other words like `'broadband', 'phone', 'mobile', 'video', 'internet'`. This tells that NMF works great in finding words with same meaning in the corpus.

On the other hand, it <u>does not produced a unique and deterministic solution</u>. As seen, the terms for all the clusters were more than slightly different on different values of k. Moreover, at a higher value of k=6, the cluster 5 was unable to produce related terms.

6 References

[1] D.D. Lee & H.S. Seung. "Learning the parts of objects by nonnegative matrix factorization". *Nature*, 401:788–91, 1999.
[2] Jama Documentation, http://math.nist.gov/javanumerics/jama/doc/
[3] TF-IDF, http://en.wikipedia.org/wiki/Tf%E2%80%93idf
[4] Oracle Java Documentation, http://docs.oracle.com/javase/tutorial/