**Text Mining –Clustering**

**Wenxin Zhao**

**Topic: Text Mining--Clustering**

**Description of the particular problem within the selected data mining topic to be addressed in this project:**

Suppose we have a set of English text documents and wish to determine which book is most relevant to the query "the beautiful eye". A simple way to start out is by eliminating documents that do not contain all three words "the", "beautiful", and "eye", but this still leaves many documents. To further distinguish them, we might count the number of times each term occurs in each document and sum them all together; the number of times a term occurs in a document is called its term frequency. However, because the term "the" is so common, this will tend to incorrectly emphasize documents which happen to use the word "the" more frequently, without giving enough weight to the more meaningful terms "beautiful" and "eye". The term "the" is not a good keyword to distinguish relevant and non-relevant documents and terms, unlike the less common words "brown" and "cow".

1. **Description of the approach used in this project to tackle the above problem:**

Calculate tf-idf value for each word.

The tf-idf value increases [proportionally](http://en.wikipedia.org/wiki/Proportionality_(mathematics)) to the number of times a word appears in the document, but is offset by the frequency of the word in the corpus, which helps to control for the fact that some words are generally more common than others.

tf–idf is the product of two statistics, term frequency and inverse document frequency.

**term frequency** tf(t,d)---the number of times that term t occurs in document d.

**Inverse document frequency** is a measure of whether the term is common or rare across all documents. It is obtained by dividing the total number of [documents](http://en.wikipedia.org/wiki/Documents) by the number of documents containing the term, and then taking the [logarithm](http://en.wikipedia.org/wiki/Logarithm) of that [quotient](http://en.wikipedia.org/wiki/Quotient).



Then tf–idf is calculated as



1. **Dataset Name:**

OliverTwist, DonQuixote and Pride&Prejudice.

1. **Where found:**

Free ebooks - Project Gutenberg (http://www.gutenberg.org/)

1. **Dataset Description:**

The data collection used here has been extracted from three different books, Oliver Twist,Don Quixote,and Pride and Prejudice.they all novels. The documents in the data collection correspond to paragraphs of the original books.(only count paragraph which between 60 and 300 words in length.)The dataset has been already formatted into a structure array and saved into three separated files (one for each book-derived subset).

Each of the elements in the data structures contains the following fields: ‘**book**’, a string containing the name of the corresponding book; ‘**chap’**, an integer identifying the number of the corresponding chapter; ‘**text**’, a string containing the original raw text of the document (paragraph) represented by such element; ‘**token**’, a cell array of strings containing the individual tokens within the document; ‘**vocab’**, a cell array of strings containing the unique tokens within the document, i.e. the document’s vocabulary; and ‘**count’**, an integer array containing the term-frequency counts for the corresponding vocabulary terms.

There are 840 documents (paragraphs) from Oliver Twist, 843 documents (paragraphs) from Don Quixote and 666 documents (paragraphs) from Pride and Prejudice. We have total 2349 documents.

1. **Initial data preprocessing, if any:**

1. Extract the vocabulary of overall data collection from each sub collection and union them. Now we have 13664 vocabulary. 2.generates random indexes for each sub collection.3.We will consider 100 documents (random) from each of the three sub collections for constructing the data set, so the total size of the resulting test set will be 300(random) documents. 4. Compute ‘tf-idf’ (term frequency–inverse document frequency) and normalization.

**7**. Three Guiding Questions about the dataset domain:

1. What features does this three dataset (1,Oliver Twist.2,Don Quixote.3,Pride and Prejudice) have?

Answer: there are three plots for three sub-dataset--- frequency of document size (in words)-- shown below. Dataset1: Oliver Twist. Dataset2: Don Quixote. Dataset 3: Pride and Prejudice.



Seen from the three plots, we can tell dataset 1 (Oliver Twist) and dataset 3(Pride and Prejudice) have relatively large document size compared with other two.

2. What can we do to weight one word’s importance,ie, how important a word is to a [document](http://en.wikipedia.org/wiki/Document) in a collection ?

Answer: calculate tf.idf. Details in ‘summary what I learned’ part.

3. What can we do to narrow raw dataset size before experiment?

Answer: We may set random index for each documents, and then select part of documents according to index.In this project, I select around 1/3 of total documents. Only consider 100 documents (random) from each of the three sub collections for constructing the data set, so the total size of the resulting test set will be 300(random) documents.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | **Tool** | **Number of cluster** | **Mining**  **Technique** | **Distance** | **initial cluster centroid positions (changed as ‘start’ in matlab)** | **Evaluation(accuracy)** | **Comments** | **Clusters** |
| E1 |  | Matlab | 3 | K-means | sqEuclidean | sample | 65.554% | ‘sample’: Select k observations from X at random. | C0 81 (27%)  C1 102  (34%)  C2 117  (39%) |
| E2 |  | Matlab | 3 | K-means | cosine | sample | 66.6667% | Changed with distance and ‘initial cluster centroid positions’ the clusters and accuracy changed slightly | C0 80 (26.6%)  C1 103  (34.3%)  C2 117  (39%) |
| E3 |  | Matlab | 3 | K-means | cosine | uniform | 67.831% | ‘Uniform’ :Select k points uniformly at random from the range of X. Not valid with Hamming distance. | C0 71 (23.6%)  C1 103  (34.3%)  C2 126  (42%) |
| E4 |  | Matlab | 4 | K-means | sqEuclidean | sample | / | It seems like k-means divide one large cluster(42%) into two small clusters (22%) and (21%). | C0 63 (21%)  C1 66 (22%)  C2 100(33%)  C3 71 (23.67%) |
| E5 |  | Matlab | 5 | K-means | sqEuclidean | sample | / | k-means seems like it divide one relatively large cluster into two small clusters. | C0 46(15.3%)  C1 64(21.3%)  C2 33(11%)  C3 61(20.3%)  C4 96(32%) |

Additional post-possessing

* we need to compute a numeric array containing an index that indicates which book each document was extracted from: ‘1’ for Oliver Twist, ‘2’ for Don Quixote and ‘3’ for Pride and Prejudice. This category will only used for evaluation purpose.
* After k-means, 3 clusters case. we have 3 clusters which may not be corresponding to three books(1’ for Oliver Twist, ‘2’ for Don Quixote and ‘3’ for Pride and Prejudice). For calculate an accurate accuracy, we can find a ‘best’ sequence with lowest distance value(in this case, I use ‘cosine’ distance to generate the ‘best’ sequence of clusters.) Figure showed below indicates relation between ‘Actual Category Index’ vs ‘Prediction Cluster Index’. (add some noise to show the frequency of each index-pair location.)

So the ‘best’ sequence of clusters is [2,3,1].



**Analysis of Results:**

1. Analyze the effect of varying parameters/experimental settings on the results. 2. Analyze the results from the point of view of the Domain, and discuss the answers that the experiments provided to your guiding questions. 3. Include and explain (some of) the best / most interesting results you obtained in your experiments. 4. Include visualizations.

1. Changed with distance and ‘initial cluster centroid positions’ the clusters and accuracy changed slightly, around 65%-67%. It is relatively good accuracy, because if we assign one document into three clusters randomly, we got an accuracy 33%.
2. Changed with number of clusters, from 3 clusters to 4 clusters, k-means algorithm using distance as ‘sqEuclidean’ and ‘start’ as ‘sample’. In 3 clusters case, we get C0 71 (23.6%), C1 103 (34.3%), C2 126 (42%). In 4 clusters case, we get C0 63 (21%), C1 66 (22%), C2 100(33%), C3 71 (23.67%). It seems like k-means divide one large cluster(42%) into two small clusters (22%) and (21%).
3. Best model: Select k points uniformly at random from the range of X.--’uniform’ and distance type as ’cosine’, we got accuracy equals to ‘70%’, with confusion matrix as follow.

Cat 1 Cat 2 Cat 3

Cluster 1: 72 26 0

Cluster 2: 18 70 32

Cluster 3: 10 4 68

1. For Cluster number =3, distance=’cosine’, ‘start’=’simple’, we have this confusion matrix as follow.



Cat 1 Cat 2 Cat 3

Cluster 1: 72 26 0

Cluster 2: 18 70 32

Cluster 3: 10 4 68

We can see from this confusion matrix, cluster 3 mainly composed of documents belongings to category3 (Pride and Prejudice), around 63% of cluster 2 composed of documents belongings to correct category2 (Don Quixote), 69% of cluster 1 composed of documents belongings to correct category1 (Oliver Twist).

**Summary of what you learned in this project:**

* What is Tf-idf weighting ?

assigns to term 't' a weight in document 'd' that is

1.high value when a 'word' occurs many times within a small number of documents (thus lending high discriminating power to those documents);

2.lower value when the term occurs fewer times in a document, or occurs in many documents (thus offering a less pronounced relevance signal);

3.low value when the term occurs in virtually all documents.

* How to pre-possessing structure data as I mentioned above using Matlab.
* How to post-possessing data for evaluation as I mentioned above using Matlab.