4

Feature Selection and Document Clustering

Inderjit Dhillon
Jacob Kogan
Charles Nicholas

Overview

Feature selection is a basic step in the construction of a vector space or *bag of words* model [BB99]. In particular, when the processing task is to partition a given document collection into clusters of similar documents a choice of good features along with good clustering algorithms is of paramount importance. This chapter suggests two techniques for feature or term selection along with a number of clustering strategies. The selection techniques significantly reduce the dimension of the vector space model. Examples that illustrate the effectiveness of the proposed algorithms are provided.

4.1 Introduction

A common form of text processing in many information retrieval systems is based on the analysis of word occurrences across a document collection. The number of words/terms used by the system defines the dimension of a vector space in which the analysis is carried out. Reduction of the dimension may lead to significant savings of computer resources and processing time. However poor feature selection may dramatically degrade the information retrieval system's performance.

Dhillon and Modha [DM01] have recently used the spherical k–means algorithm for clustering text data. In one of the experiments of [DM01] the algorithm was applied to a data set containing 3893 documents. The data set contains the following three document collections (available from `ftp://ftp.cs.cornell.edu/pub/smart`):

- Medlars Collection (1033 medical abstracts),

- CISI Collection (1460 information science abstracts),
- Cranfield Collection (1400 aerodynamics abstracts).

Partitioning the entire collection into 3 clusters generates the following "confusion" matrix reported in [DM01]:

	Medlars	CISI	Cranfield
cluster 0	1004	5	4
cluster 1	18	1440	16
cluster 2	11	15	1380

(here the entry ij is the number of documents that belong to cluster i and document collection j). The confusion matrix shows that only 69 documents (i.e., less that 2% of the entire collection) have been "misclassified" by the algorithm. After removing stopwords Dhillon and Modha [DM01] reported 24,574 unique words, and after eliminating low–frequency and high–frequency words they selected 4,099 words to construct the vector space model.

The main goal of this contribution is to provide algorithms for (a) selection of a small set of terms and (b) clustering of document vectors. In particular, for data similar to described above, we are able to generate better or similar quality confusion matrices while reducing the dimension of the vector space model by more than 70%.

The outline of the chapter is the following. A brief review of existing algorithms we employ for clustering documents is provided in Section 4.2. The data is described in Section 4.3. The term selection techniques along with the clustering results are presented in Sections 4.4 and 4.5, while Section 4.6 contains a new clustering algorithm along with the corresponding clustering results. Future research directions are briefly outlined in Section 4.7.

4.2 Clustering Algorithms

In this section, we review two known clustering algorithms we apply to partition documents into clusters. The means algorithm, introduced in [Kog01b], is a combination of the batch $k-$means and the incremental $k-$means algorithms (see [DHS01b]). The Principal Direction Divisive Partitioning method was introduced recently by D. Boley [Bol98].

4.2.1 Means clustering algorithm

For a set of vectors $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_d\}$ in Euclidean space $\mathbf{R}^w$ denote the centroid of the set $\frac{1}{d}\sum_{i=1}^{d}\mathbf{x}_i$ by $\mathbf{m}(\mathbf{X})$.

Let $\{\pi_l\}_{l=1}^{k}$ be a partition of $\mathbf{X}$ with the corresponding centroids $\mathbf{m}_1 = \mathbf{m}(\pi_1), \ldots, \mathbf{m}_k = \mathbf{m}(\pi_k)$. Define the quality $\mathcal{Q}_2$ of the partition $\{\pi_l\}_{l=1}^{k}$ by

$$\mathcal{Q}_2\left(\{\pi_l\}_{l=1}^{k}\right) = \sum_{l=1}^{k} \sum_{\mathbf{x}\in\pi_l} \|\mathbf{x} - \mathbf{m}(\pi_l)\|^2 = \sum_{l=1}^{k} \sum_{\mathbf{x}\in\pi_l} \|\mathbf{x} - \mathbf{m}_l\|^2 . \tag{4.1}$$

For $\mathbf{x} \in \pi_i \subseteq \mathbf{X}$ denote the index of the centroid nearest $\mathbf{x}$ by $\mathtt{min}(\mathbf{x})$ (i.e., $\left\|\mathbf{x} - \mathbf{m}_{\mathtt{min}\,(\mathbf{x})}\right\| \le \|\mathbf{x} - \mathbf{m}_l\|$, $l = 1, \ldots, k$). Define now the partition $\mathtt{nextKM}\left(\{\pi_l\}_{l=1}^{k}\right) = \{\pi_l'\}_{l=1}^{k}$ as follows:

$$\pi_i' = \{\mathbf{x} \,:\, \mathtt{min}\,(\mathbf{x}) = i\} .$$

It is easy to see [DHS01b] that

$$\mathcal{Q}_2\left(\{\pi_l\}_{l=1}^{k}\right) \ge \mathcal{Q}_2\left(\mathtt{nextKM}\left(\{\pi_l\}_{l=1}^{k}\right)\right) . \tag{4.2}$$

Next we present the classical batch $k-$means algorithm and discuss some of its deficiencies. The algorithm suffers from the two major drawbacks:

Batch $k-$means clustering algorithm (Forgy [For65]).
For a user supplied tolerance $\mathtt{tol} < 0$ do the following:

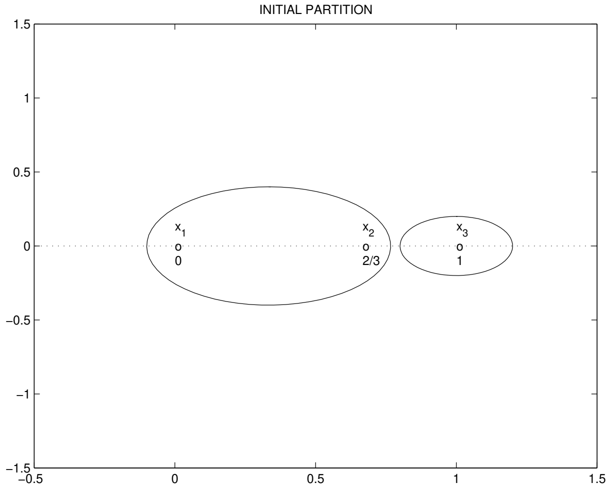
1. Start with an arbitrary partitioning $\left\{\pi_l^{(0)}\right\}_{l=1}^{k}$. Set the index of iteration $t = 0$.
2. Generate the partition $\mathtt{nextKM}\left(\left\{\pi_l^{(t)}\right\}_{l=1}^{k}\right)$.
   if $\left[\mathcal{Q}_2\left(\mathtt{nextKM}\left(\left\{\pi_l^{(t)}\right\}_{l=1}^{k}\right)\right) - \mathcal{Q}_2\left(\left\{\pi_l^{(t)}\right\}_{l=1}^{k}\right) \le \mathtt{tol}\right]$
   set $\left\{\pi_l^{(t+1)}\right\}_{l=1}^{k} = \mathtt{nextKM}\left(\left\{\pi_l^{(t)}\right\}_{l=1}^{k}\right)$
   increment t by 1.
   go to 2
3. Stop.

---

1. The quality of the final partition depends on a good choice of the initial partition.
2. The algorithm may get trapped at a local minimum even for a very simple one dimensional set $\mathbf{X}$.

We address the first point in Sections 4.2.2 and 4.6. The second point is illustrated by the following example.

EXAMPLE 4.2.1 *Let* $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3\}$ *with* $\mathbf{x}_1 = 0$, $\mathbf{x}_2 = 2/3$, *and* $\mathbf{x}_3 = 1$. *Consider the initial partition* $\pi_1^{(0)} = \{\mathbf{x}_1, \mathbf{x}_2\}$, $\pi_2^{(0)} = \{\mathbf{x}_3\}$ *with* $\mathcal{Q}_2\left\{\pi_1^{(0)}, \pi_2^{(0)}\right\} = 2/9$.



Note that an application of the batch k–means algorithm does not change the initial partition $\left\{\pi_1^{(0)}, \pi_2^{(0)}\right\}$. *At the same time it is clear that the partition* $\{\pi_1', \pi_2'\}$ ($\pi_1' = \{\mathbf{x}_1\}$, $\pi_2' = \{\mathbf{x}_2, \mathbf{x}_3\}$) *with* $\mathcal{Q}_2\{\pi_1', \pi_2'\} = 2/36$ *is superior to the initial partition.*

A different version of the $k-$means algorithm, incremental $k-$means clustering, is discussed next. This version remedies the problem illustrated in Example 4.2.1.

The decision of whether a vector $\mathbf{x} \in \pi_i$ should be moved from cluster π_i to cluster π_j is made by the batch k–means algorithm based on the sign of

$$\Delta = -\|\mathbf{x} - \mathbf{m}(\pi_i)\|^2 + \|\mathbf{x} - \mathbf{m}(\pi_j)\|^2 . \tag{4.3}$$

If Δ is negative, then the vector $\mathbf{x}$ is moved by the batch k–means algorithm. The exact change in the value of the objective function (i.e., the difference between the "new" and the "old" values of the objective function) caused by the move is

$$\Delta_{\text{exact}} = -\frac{n_i}{n_i - 1}\|\mathbf{x} - \mathbf{m}(\pi_i)\|^2 + \frac{n_j}{n_j + 1}\|\mathbf{x} - \mathbf{m}(\pi_j)\|^2 , \tag{4.4}$$

where $n_j = |\pi_j|$, $n_i = |\pi_i|$ are the number of vectors in clusters π_j and π_i respectively (see e.g. [Kog01a]). The more negative Δ_{exact} is the larger the drop in the value of the objective function. The difference between the expressions

$$\Delta - \Delta_{\text{exact}} = \frac{1}{n_i - 1}\|\mathbf{x} - \mathbf{m}(\pi_i)\|^2 + \frac{1}{n_j + 1}\|\mathbf{x} - \mathbf{m}(\pi_j)\|^2 \geq 0$$

is negligible when the clusters π_i and π_j are large. However $\Delta - \Delta_{\text{exact}}$ may become significant for small clusters. For example, for $\mathbf{x} = \mathbf{x}_2$ in Example 4.2.1 one has $\Delta = 0$, and $\Delta_{\text{exact}} < 0$. This is why batch $k-$means misses the "better" partition $\{\pi_1', \pi_2'\}$. The incremental $k-$means clustering algorithm eliminates this problem. Before presenting the algorithm, we need a few additional definitions.

DEFINITION 4.2.1 *A first variation of a partition $\{\pi_l\}_{l=1}^k$ is a partition $\{\pi_l'\}_{l=1}^k$ obtained from $\{\pi_l\}_{l=1}^k$ by removing a single vector $\mathbf{x}$ from a cluster π_i of $\{\pi_l\}_{l=1}^k$ and assigning this vector to an existing cluster π_j of $\{\pi_l\}_{l=1}^k$.*

Note that the partition $\{\pi_l\}_{l=1}^k$ is a first variation of itself. Next we look for the "steepest descent" first variation, i.e., a first variation that leads to the maximal decrease of the objective function. The formal definition follows:

DEFINITION 4.2.2 *The partition $\texttt{nextFV}\left(\{\pi_l\}_{l=1}^k\right)$ is a first variation of $\{\pi_l\}_{l=1}^k$ so that for each first variation $\{\pi_l'\}_{l=1}^k$ one has*

$$\mathcal{Q}_2\left(\texttt{nextFV}\left(\{\pi_l\}_{l=1}^k\right)\right) \leq \mathcal{Q}_2\left(\{\pi_l'\}_{l=1}^k\right). \tag{4.5}$$

Incremental $k-$means clustering algorithm (also see [DHS01b], Section 10.8). For a user supplied tolerance $\texttt{tol} < 0$ do the following:

1. Start with an arbitrary partitioning $\left\{\pi_l^{(0)}\right\}_{l=1}^k$. Set the index of iteration $t = 0$.
2. Generate the partition $\texttt{nextFV}\left(\left\{\pi_l^{(t)}\right\}_{l=1}^k\right)$.

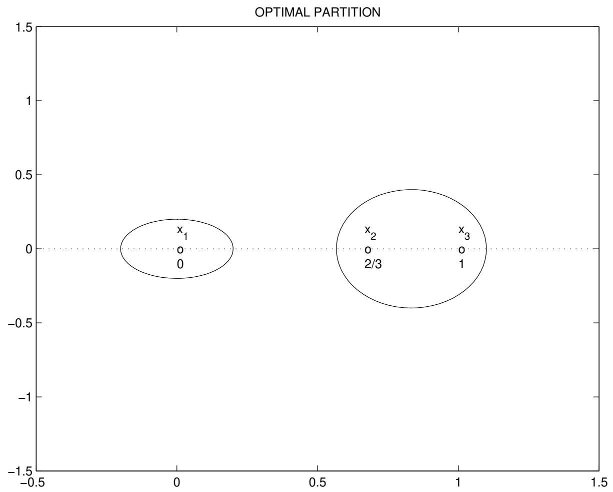
   if $\left[\mathcal{Q}_2\left(\texttt{nextFV}\left(\left\{\pi_l^{(t)}\right\}_{l=1}^k\right)\right) - \mathcal{Q}_2\left(\left\{\pi_l^{(t)}\right\}_{l=1}^k\right) \leq \texttt{tol}\right]$

   set $\left\{\pi_l^{(t+1)}\right\}_{l=1}^k = \texttt{nextKM}\left(\left\{\pi_l^{(t)}\right\}_{l=1}^k\right)$

   increment t by 1.

   go to 2
3. Stop.

EXAMPLE 4.2.2 *Let the vector set and the initial partition be given by Example 4.2.1. A single iteration of incremental k–means generates the optimal partition $\pi_1^{(1)} = \{\mathbf{x}_1\}$, $\pi_2^{(1)} = \{\mathbf{x}_2, \mathbf{x}_3\}$ as shown in the following figure.*



While computationally more accurate, incremental $k-$means is slower than batch $k-$means. Each iteration of incremental $k-$means changes cluster affiliation of a single vector only. The examples suggest the following "merger" of the two algorithms: Unlike the means algorithm of [Kog01b] the algorithm described above keeps the number of clusters k fixed throughout the iterations. Otherwise the above algorithm enjoys advantages of the means algorithm:

1. The means algorithm always outperforms batch $k-$means in cluster quality (see [Kog01b]).
2. All numerical computations associated with Step 3 of the means algorithm have been already performed at Step 2 (see (4.3) and (4.4)). The improvement over batch $k-$means comes, therefore, at virtually no additional computational expense.

For simplicity we shall henceforth refer to Algorithm 4.2.1 as the means algorithm.

The $k-$means algorithm is known to be sensitive to the choice of an initial partition. A clustering algorithm that may be used for generating good initial partitions is presented next.

4.2.2 Principal Direction Divisive Partitioning

A memory efficient and fast clustering algorithm was introduced recently by D. Boley [Bol98]. The method is not based on any distance or similarity measure, and takes advantage of sparsity of the "word by document" matrix.

The algorithm proceeds by dividing the entire collection into two clusters by using principal directions. Each of these two clusters will be divided into two sub-

Simplified version of the means clustering algorithm (see [Kog01b]).
For user supplied tolerances $\mathtt{tol}_1 < 0$ and $\mathtt{tol}_2 < 0$ do the following:

1. Start with an arbitrary partitioning $\left\{\pi_l^{(0)}\right\}_{l=1}^{k}$. Set the index of iteration $t = 0$.
2. Generate the partition $\mathtt{nextKM}\left(\left\{\pi_l^{(t)}\right\}_{l=1}^{k}\right)$.
   if $\left[\mathcal{Q}_2\left(\mathtt{nextKM}\left(\left\{\pi_l^{(t)}\right\}_{l=1}^{k}\right)\right) - \mathcal{Q}_2\left(\left\{\pi_l^{(t)}\right\}_{l=1}^{k}\right) \leq \mathtt{tol}_1\right]$
   set $\left\{\pi_l^{(t+1)}\right\}_{l=1}^{k} = \mathtt{nextKM}\left(\left\{\pi_l^{(t)}\right\}_{l=1}^{k}\right)$
   increment t by 1.
   go to 2
3. Generate the partition $\mathtt{nextFV}\left(\left\{\pi_l^{(t)}\right\}_{l=1}^{k}\right)$.
   if $\left[\mathcal{Q}_2\left(\mathtt{nextFV}\left(\left\{\pi_l^{(t)}\right\}_{l=1}^{k}\right)\right) - \mathcal{Q}_2\left(\left\{\pi_l^{(t)}\right\}_{l=1}^{k}\right) \leq \mathtt{tol}_2\right]$
   set $\left\{\pi_l^{(t+1)}\right\}_{l=1}^{k} = \mathtt{nextFV}\left(\left\{\pi_l^{(t)}\right\}_{l=1}^{k}\right)$.
   increment t by 1.
   go to 2
4. Stop.

clusters using the same process recursively. The subdivision of a cluster is stopped when the cluster satisfies a certain "quality" criterion (for example, the cluster's variance does not exceed a predefined threshold).

Clustering of a set of vectors in $\mathbf{R}^n$ is, in general, a difficult task. There is, however, an exception. When $n = 1$, and all the vectors belong to a one dimensional line, clustering becomes relatively easy. In many cases a good partition of a one–dimensional set Y into two subsets Y_1 and Y_2 amounts to a selection of a number, say μ, so that

$$Y_1 = \{y \ : \ y \in Y, \ y \leq \mu\}, \text{ and } Y_2 = \{y \ : \ y \in Y, \ y > \mu\} \tag{4.6}$$

(in [Bol98], for example, μ is the mean).

The basic idea of Boley's Principal Direction Divisive Partitioning algorithm (PDDP) is the following:

1. Given a set of vectors $\mathbf{X}$ in $\mathbf{R}^n$ determine the line Ł that approximates $\mathbf{X}$ in the "best possible way".
2. Project $\mathbf{X}$ onto Ł, and denote the projection of the set $\mathbf{X}$ by Y (note that Y is just a set of scalars). Denote the projection of a vector $\mathbf{x}$ by y.

3. Partition Y into two subsets Y_1 and Y_2 as described by (4.6).
4. Generate the induced partition $\{\mathbf{X}_1, \mathbf{X}_2\}$ of $\mathbf{X}$ as follows:

$$\mathbf{X}_1 = \{\mathbf{x} \; : \; y \in Y_1\}, \text{ and } \mathbf{X}_2 = \{\mathbf{x} \; : \; y \in Y_2\}. \qquad (4.7)$$

D. Boley has suggested the line that maximizes variance of the projections as the best one dimensional approximation of an n dimensional set. The direction of the line is defined by the eigenvector of the covariance matrix C corresponding to the largest eigenvalue. Since C is symmetric and positive semidefinite all the eigenvalues λ_i, $i = 1, 2, \ldots, n$ of the matrix are real and non-negative, i.e., $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n \geq 0$. Furthermore, while the "scatter" value of the document set is $\lambda_1 + \lambda_2 + \cdots + \lambda_n$, the scatter value of the one dimensional projection is only λ_1 (see [Bol98]). The quantity

$$\frac{\lambda_1}{\lambda_1 + \lambda_2 + \cdots + \lambda_n} \qquad (4.8)$$

may, therefore, be considered as the fraction of information preserved under the projection (in contrast with the "lost" information $\frac{\lambda_2 + \cdots + \lambda_n}{\lambda_1 + \lambda_2 + \cdots + \lambda_n}$). Inspite of the fact that the numerator of (4.8) contains only one eigenvalue of a large matrix the algorithm generates remarkable results (see e.g. [Bol98], [BGG+99a], [BGG+99b]). For instance, examples provided in [Kog01b] show that an application of the $k-$means clustering algorithm to a partition generated by PDDP leads to only about 5% improvement in the objective function value.

In the next section, we describe the data set and corresponding feature selection problem considered in this study.

4.3 Data and term quality

Our data set is a merger of the three document collections (available from `http://www.cs.utk.edu/ lsi/`):

- DC0 (Medlars Collection 1033 medical abstracts)
- DC1 (CISI Collection 1460 information science abstracts)
- DC2 (Cranfield Collection (1398 aerodynamics abstracts)

The Cranfield collection tackled by Dhillon and Modha contained two empty documents. These two documents have been removed from DC2. The other document collections are identical.

We denote the overall collection of 3891 documents by DC. After stopword removal (see `ftp://ftp.cs.cornell.edu/pub/smart/english.stop`), and stemming (see [Por80]) the data set contains 15,864 unique terms (no stemming was applied to the 24,574 unique words reported in [DM01]).

Our first goal is to select "good" index terms. We argue that for recovering the three document collections the term "blood" is much more useful than the term "case". Indeed, while the term "case" occurs in 253 Medlars documents, 72 CISI documents, and 365 Cranfield documents, the term "blood" occurs in 142 Medlars documents, 0 CISI documents, and 0 Cranfield documents. With each term $\mathbf{t}$ we associate a three dimensional "direction" vector $\mathbf{d}(\mathbf{t}) = (d_0(\mathbf{t}), d_1(\mathbf{t}), d_2(\mathbf{t}))$, so that $d_i(\mathbf{t})$ is the number of documents in a collection DCi containing the term $\mathbf{t}$. So, for example, $\mathbf{d}$("case") $= (253, 72, 365)$, and $\mathbf{d}$("blood") $= (142, 0, 0)$. In addition to "blood", terms like "layer" ($\mathbf{d}$("layer") $= (6, 0, 358)$), or "retriev" ($\mathbf{d}$("retriev") $= (0, 262, 0)$) seem to be much more useful than the terms "case", "studi" and "found" with $\mathbf{d}$("studi") $= (356, 341, 238)$, and $\mathbf{d}$("found") $= (211, 93, 322)$, respectively.

When only the "combined" collection DC of 3891 documents is available the above described construction of direction vectors is not possible. In Sections 4.4 and 4.5, we present algorithms that attempt to select "useful" terms when the direction vector $\mathbf{d}(\mathbf{t})$ is not available.

For each selection algorithm described in this chapter we introduce a quality functional q, so that the quality of a term $\mathbf{t}$ is given by $q(\mathbf{t})$. Higher values of $q(\mathbf{t})$ correspond to "better" terms $\mathbf{t}$. To exploit statistics of term occurrence throughout the corpus we remove terms that occur in less than r sentences across the collection, and denote the remaining terms by slice(r) (r should be collection dependent, the experiments in this chapter are performed with $r = 20$). The first l best quality terms that belong to slice(r) define the dimension of the vector space model.

In the next two sections, we present two different term selection techniques along with corresponding document clustering results.

4.4 Term variance quality

We denote the frequency of a term $\mathbf{t}$ in the document $\mathbf{d}_j$ by f_j. Following the ideas of Salton and McGill [SM83] we measure the quality of the term $\mathbf{t}$ by

$$q_0(\mathbf{t}) = \sum_{j=1}^{n_0} f_j^2 - \frac{1}{n_0}\left[\sum_{j=1}^{n_0} f_j\right]^2, \tag{4.9}$$

where n_0 is the total number of documents in the collection (note that $q_0(\mathbf{t})$ is proportional to the term frequency variance). Tables 4.1 and 4.2 present 15 "best", and 15 "worst" terms for slice(20) in our collection of 3891 documents.

To evaluate the impact of feature selection by q_0 on clustering we conduct the following experiment. The best quality 600 terms are selected, and unit norm

term	$q_0(\mathbf{t})$	$d_0(\mathbf{t})$	$d_1(\mathbf{t})$	$d_2(\mathbf{t})$
flow	7687.795	35	34	714
librari	7107.937	0	523	0
pressur	5554.151	57	12	533
number	5476.418	92	204	568
cell	5023.158	210	2	2
inform	4358.370	28	614	44
bodi	3817.281	84	23	276
system	3741.070	82	494	84
wing	3409.713	1	0	216
effect	3280.777	244	159	539
method	3239.389	121	252	454
layer	3211.331	6	0	358
jet	3142.879	1	0	92
patient	3116.628	301	3	0
shock	3085.249	4	1	224

Table 4.1. 15 “best” terms in slice(20) according to q_0

term	$q_0(\mathbf{t})$	$d_0(\mathbf{t})$	$d_1(\mathbf{t})$	$d_2(\mathbf{t})$
suppos	21.875	6	7	9
nevertheless	21.875	6	11	5
retain	21.875	9	4	9
art	21.875	0	20	2
compos	21.875	5	5	12
ago	21.875	2	18	2
elabor	21.875	3	16	3
obviou	21.897	4	9	6
speak	20.886	6	12	3
add	20.886	3	14	4
understood	20.886	2	14	5
pronounc	20.886	18	0	3
pertain	19.897	3	8	9
merit	19.897	1	9	10
provis	19.897	1	18	1

Table 4.2. 15 “worst” terms in slice(20) according to q_0

vectors for the 3891 documents are built (we use the `tfn` scheme to construct document vectors, for details see [DM01]). A two step procedure is employed to partition the 3891 vectors into 3 clusters:

1. the PDDP algorithm is applied to generate 3 clusters (the obtained clusters are used as an initial partition in the next step),
2. the means algorithm is applied to the partition obtained in the previous step.

Note that there is no *a priori* connection between document collection i and cluster i. Hence, one can not expect the confusion matrix to have diagonal structure unless rows (or columns) of the matrix are suitably permuted. A good clustering procedure should be able to produce a confusion matrix with a single "dominant" entry in each row. The confusion matrices for the three clusters provided in Tables 4.3 and 4.4 illustrate this remark.

	DC0	DC1	DC2
cluster 0	272	9	1379
cluster 1	4	1285	11
cluster 2	757	166	8
empty documents			
cluster 3	0	0	0

Table 4.3. PDDP generated initial confusion matrix with 470 misclassifieddocuments using 600 best q_0 terms

When the number of terms is relatively small some documents may contain no selected terms, and their corresponding vectors are zeros. We always remove these vectors ahead of clustering and assign the "empty" documents into a special cluster. This cluster concludes the confusion matrix (and is empty in this experiment).

	DC0	DC1	DC2
cluster 0	1	3	1365
cluster 1	8	1433	18
cluster 2	1024	24	15
empty documents			
cluster 3	0	0	0

Table 4.4. Means generated final confusion matrix with 69 misclassified documents using 600 best q_0 terms

While the quality of the confusion matrix presented above is similar to that reported in [DM01] (see Section 9.1), the dimension of our vector space model, 600, is about only 15% of the vector space dimension reported in [DM01].

The abstracts comprising the document collection DC are relatively short documents (from a half page to a page and a half long). It is not unusual to find terms that occur in many documents only once. Such terms score high by (4.9). At the same time these terms may lack any specificity. Indeed, the term "studi" with $\mathbf{d}(\text{"studi"}) = (356, 341, 238)$ is ranked $28th$ by q_0, and the term "present" with $\mathbf{d}(\text{"present"}) = (236, 314, 506)$ is ranked $35th$. In order to penalize such terms, we modify (4.9) and introduce the quality of term $q_1(\mathbf{t})$ as the variance of $\mathbf{t}$ over documents that contain the term *at least once*. That is

$$q_1(\mathbf{t}) = \sum_{j=1}^{n_1} f_j^2 - \frac{1}{n_1}\left[\sum_{j=1}^{n_1} f_j\right]^2, \tag{4.10}$$

where n_1 is the number of documents in which $\mathbf{t}$ occurs at least once, and $f_j \geq 1$, $j = 1, \ldots, n_1$. Tables 4.5 and 4.6 present the 15 "best", and the 15 "worst" q_1 terms for slice(20) respectively.

term	$q_1(\mathbf{t})$	$d_0(\mathbf{t})$	$d_1(\mathbf{t})$	$d_2(\mathbf{t})$
librari	3147.074	0	523	0
flow	3146.048	35	34	714
number	2734.665	92	204	568
pressur	2528.225	57	12	533
cell	2225.177	210	2	2
inform	1851.231	28	614	44
bodi	1768.182	84	23	276
system	1518.877	82	494	84
shock	1490.113	4	1	224
jet	1463.569	1	0	92
theori	1341.363	23	117	452
method	1303.141	121	252	454
layer	1296.008	6	0	358
patient	1247.944	301	3	0
effect	1210.772	244	159	539

Table 4.5. 15 "best" terms in slice(20) according to q_1

We select the best 600 terms and apply first the PDDP algorithm, and then the means algorithm to the corresponding 3891 vectors. The resulting confusion matrices are given in Tables 4.7 and 4.8. An increase in the number of selected terms does lead to a modest improvement in the quality of confusion matrices. In what follows, we summarize the improvement for term selections based on q_0 and q_1. Table 4.9 presents results for terms selected by q_0. The first row of Table 4.9 lists clustering algorithms, and the first column shows the number of selected terms. The other columns indicate the number of misclassified documents. The displayed results indicate that the algorithm "collapses" when the number of selected terms drops below 600. Table 4.10 contains information relevant to q_1.

term	$q_1(\mathbf{t})$	$d_0(\mathbf{t})$	$d_1(\mathbf{t})$	$d_2(\mathbf{t})$
add	0.000	3	14	4
retain	0.000	9	4	9
reproduc	0.000	7	12	5
provis	0.000	1	18	1
pronounc	0.000	18	0	3
diminish	0.000	16	5	14
suppos	0.000	6	7	9
doubt	0.000	4	12	10
speak	0.000	6	12	3
context	0.000	7	45	1
understood	0.000	2	14	5
pertain	0.000	3	8	9
bring	0.000	8	30	8
ago	0.000	2	18	2
occasion	0.000	18	11	1

Table 4.6. 15 “worst” terms in slice(20) according to q_1

	DC0	DC1	DC2
cluster 0	461	10	1380
cluster 1	3	803	0
cluster 2	569	647	18
“empty” documents			
cluster 3	0	0	0

Table 4.7. PDDP generated initial confusion matrix with 1061 misclassified documents using 600 best q_1 terms

	DC0	DC1	DC2
cluster 0	0	3	1360
cluster 1	6	1416	13
cluster 2	1027	41	25
empty documents			
cluster 3	0	0	0

Table 4.8. Means generated final confusion matrix with 88 misclassified documents using 600 best q_1 terms

The tables indicate that with 1,300 selected terms (i.e., only about 30% of the 4,099 terms reported in [DM01]) the number of “misclassified” documents is slightly lower than the number reported in [DM01].

	documents misclassified by	
# of terms	pddp	means
500	1062	989
600	470	69
700	388	63
1000	236	55
1300	181	53

Table 4.9. Number of misclassified documents for term selection based on q_0

	documents misclassified by	
# of terms	pddp	means
500	1055	94
600	1061	88
700	617	74
1000	410	64
1300	232	55

Table 4.10. Number of "misclassified" documents for term selection based on q_1

In the next section, we introduce a measure of distance between terms. The distance is based on term co–occurrence in sentences across the document collection. The quality of a term $\mathbf{t}$ presented next is based on distribution of terms "similar" to $\mathbf{t}$ and co–occurring with $\mathbf{t}$ in sentences across the document collection.

4.5 Same context terms

The second approach to the term selection problem is based on co–occurrence of "similar" terms in "the same context". Our departure point is the definition (attributed to Leibniz): two expressions are synonymous if the substitution of one for the other never changes the truth value of a sentence in which the substitution is made.

We follow ideas of Grefenstette [G.94]: "you can begin to know the meaning of a word (or term) by the company it keeps" and "words or terms that occur in 'the same context' are 'equivalent'", and Schütze and Pedersen [SP95]: "the assumption is that words with similar meanings will occur with similar neighbors if enough text material is available." Profiles introduced below formalize these notions.

4.5.1 Term profiles

Our construction is the following:

1. Let $\mathbf{T} = \{\mathbf{t}_1, \ldots, \mathbf{t}_m\}$ be an alphabetically sorted list of unique terms that occur in the document collection DC.
2. For each term $\mathbf{t}$ in $\mathbf{T}$ denote the set of sentences in DC containing $\mathbf{t}$ by $\mathbf{s}(\mathbf{t})$. The size of the set is denoted by $|\mathbf{s}(\mathbf{t})|$, and $s_{\max} = \max_{\mathbf{t}\in\mathbf{T}} |\mathbf{s}(\mathbf{t})|$.
3. For each term $\mathbf{t} \in \mathbf{T}$ the profile $\mathcal{P}(\mathbf{t})$ is defined next:

   DEFINITION 4.5.1 *The profile $\mathcal{P}(\mathbf{t})$ of the term $\mathbf{t}$ is a set of terms from the list $\mathbf{T}$ that co–occur in sentences together with the term $\mathbf{t}$, i.e.,*

   $$\mathcal{P}(\mathbf{t}) = \{\mathbf{t}' \;:\; \mathbf{t}' \in \mathbf{s}(\mathbf{t})\}.$$

   Profile $\mathcal{P}(\mathbf{t})$ contains corpus dependent information concerning the term $\mathbf{t}$ and "the company it keeps". There are number of ways to compute term similarity based on the respective profiles [Kog02]. A way to express the similarity is described below.
4. Let $\{\mathbf{s}_1, \ldots, \mathbf{s}_n\}$ be the set of all sentences contained in the document collection DC. The "term by sentence" matrix $\mathbf{S}$ is an $m \times n$ matrix whose entry $\mathbf{S}_{ij}$ is the number of times the term $\mathbf{t}_i$ occurs in the sentence $\mathbf{s}_j$. The term $\mathbf{t}_i$ profile vector $\mathbf{P}(\mathbf{t}_i) = (P_1, \ldots, P_m)^T$ is the i–th column of the symmetric matrix $\mathbf{SS}^T$. The j–th coordinate of the vector, $P_j = \left(\mathbf{SS}^T\right)_{ij}$, is the number of times the terms $\mathbf{t}_i$ and $\mathbf{t}_j$ co-occur in sentences across the document collection DC. Since $P_i \neq 0$, the vector $\mathbf{P}(\mathbf{t}_i)$ can be normalized.
5. DEFINITION 4.5.2 *Unit profile vector $\mathbf{P}(\mathbf{t})$ of term $\mathbf{t}$ is defined to be* $\dfrac{\mathbf{P}(\mathbf{t})}{\|\mathbf{P}(\mathbf{t})\|}$.

Words/terms with "similar meanings" (as per a given document collection) generate similar unit profile vectors (for details see [Kog02]). We next provide a formula for term quality based on term profile.

4.5.2 Term profile quality

The term profile quality $q_{\mathbf{P}}(\mathbf{t})$ introduced in this section is based on the distribution of terms similar to $\mathbf{t}$ in the profile $\mathcal{P}(\mathbf{t})$.

For each $\mathbf{t}' \in \mathcal{P}(\mathbf{t})$ compute the dot product $c' = \mathbf{P}(\mathbf{t})^T\mathbf{P}(\mathbf{t}')$. We now sort the profile $\mathcal{P}(\mathbf{t})$ with respect to the dot products c', so that if $\mathcal{P}(\mathbf{t}) = \{\mathbf{t}_0, \mathbf{t}_1, \ldots, \mathbf{t}_n\}$, ($\mathbf{t}_0 = \mathbf{t}$), then $1 = c_0 \geq c_1 \geq \cdots \geq c_n \geq 0$. We denote the frequency of the term $\mathbf{t}_i$ in the profile $\mathcal{P}(\mathbf{t})$ by f_i and define the term profile quality $q_{\mathbf{P}}(\mathbf{t}_0)$ by a somewhat contrived formula (justification is given below):

$$q_{\mathbf{p}}(\mathbf{t}_0) = \left[\frac{|\mathbf{s}(\mathbf{t}_0)|}{s_{\max}}\right]^{0.2} \frac{1}{f_0 + ke(f)} \sum_{i=1}^{k} \left[f_i c_i \times \left(1 - \sqrt{\frac{|e(f) - f_i|}{ke(f)}}\right)\right], \quad (4.11)$$

where $e(f) = (1/k)\sum_{i=1}^{k} f_i$, and in this experiment $k = 2$. We note the following concerning the expression for the profile quality $q_{\mathbf{p}}$:

1. due to the small power, 0.2, the term $\left[\frac{|\mathbf{s}(\mathbf{t}_0)|}{s_{\max}}\right]^{0.2}$ penalizes very frequent collection terms,
2. the normalizing term $\frac{1}{f_0 + ke(f)}$ attempts to suppress the importance of very frequent profile terms similar to $\mathbf{t}_0$,
3. the term $f_i c_i$ reflects the measure of similarity between $\mathbf{t}_0$ and $\mathbf{t}_i$,
4. the term $1 - \sqrt{\frac{|e(f) - f_i|}{ke(f)}}$ imposes a penalty on a term's deviation from the expected frequency.

Table 4.11 and Table 4.12 present 15 "best", and 15 "worst" terms for slice(20).
For clustering purposes we selected 1,000 best quality index terms. Although

term	$q_{\mathbf{p}}(\mathbf{t})$	$d_0(\mathbf{t})$	$d_1(\mathbf{t})$	$d_2(\mathbf{t})$
laminar	0.264	0	0	231
layer	0.205	6	0	358
number	0.204	92	204	568
septal	0.202	25	0	0
free-stream	0.195	0	0	97
boundari	0.195	0	7	413
nephrectomi	0.168	23	0	0
unilater	0.161	27	0	0
defect	0.157	64	5	4
reynold	0.152	0	1	197
mach	0.141	0	0	384
nomin	0.136	2	2	17
moment	0.135	4	4	89
autom	0.128	0	46	0
biliari	0.126	17	0	0

Table 4.11. 15 "best" terms in slice(20) according to $q_{\mathbf{p}}$

each selected term is contained in at least 20 sentences, the selected 1,000 unit

term	$q_{\mathbf{P}}(\mathbf{t})$	$d_0(\mathbf{t})$	$d_1(\mathbf{t})$	$d_2(\mathbf{t})$
determin	0.001	108	116	299
larg	0.001	80	175	201
approxim	0.001	31	46	377
found	0.001	211	93	322
analysi	0.001	42	184	276
includ	0.001	75	169	225
rate	0.001	111	49	145
paper	0.001	31	265	200
experi	0.001	105	133	152
result	0.000	278	288	692
effect	0.000	244	159	539
studi	0.000	356	341	238
gener	0.000	76	311	329
develop	0.000	176	366	264
case	0.000	253	72	365

Table 4.12. 15 "worst" terms in slice(20) according to $q_{\mathbf{P}}$

profile vectors $\mathbf{P}(\mathbf{t})$ of dimension 15,864 (which is the total number of the unique terms, see Section 4.3) are sparse. The average number of non-zero entries in a unit profile vector is 617, i.e., less than 4% of 15,864-the dimension of the vector space.

Next we apply the means algorithm to partition 1,000 term vectors into 3 term clusters $\mathbf{T}_0$, $\mathbf{T}_1$ and $\mathbf{T}_2$. The partition of the document collection DC is based on the term clusters. For each document $\mathbf{d}$ we construct a three dimensional vector $(t_0(\mathbf{d}), t_1(\mathbf{d}), t_2(\mathbf{d}))$, where $t_i(\mathbf{d})$ is the number of terms from term cluster $\mathbf{T}_i$ contained in the document. The document $\mathbf{d}$ belongs to document cluster i if

$$t_i(\mathbf{d}) \geq t_j(\mathbf{d}),\ j = 0, 1, 2.$$

The confusion matrix for this partition of 3,891 documents with 105 misclassified documents is given in Table 4.13. A 30% increase in the number of index

	DC0	DC1	DC2
cluster 0	23	32	1386
cluster 1	975	3	1
cluster 2	35	1424	11
empty documents			
cluster 3	0	1	0

Table 4.13. Means generated final confusion matrix with 105 misclassified documents

terms leads to the decrease in the number of misclassified documents to 94, and eliminates empty documents.

The clustering algorithms discussed so far deal with general vector sets in $\mathbf{R}^n$. In the next section we present a clustering algorithm specifically designed to handle unit norm document vectors.

4.6 Spherical Principal Directions Divisive Partitioning

In this section we mimic the simple and elegant idea due to Boley and approximate a set of unit vectors $\mathbf{X} \subset \mathbf{R}^n$ by a one dimensional great circle of $\mathbf{S}^{n-1}$. A great circle is represented by an intersection of $\mathbf{S}^{n-1}$ and a two dimensional subspace $\mathbf{P}$ of $\mathbf{R}^n$. The proposed algorithm is the following: If, following ideas of [Bol98],

Spherical Principal Directions Divisive Partitioning (sPDDP) clustering algorithm.

1. Given a set of unit vectors $\mathbf{X}$ in $\mathbf{R}^n$ determine the two dimensional plane $\mathbf{P}$ that approximates $\mathbf{X}$ in the "best possible way".
2. Project $\mathbf{X}$ onto $\mathbf{P}$. Denote the projection of the set $\mathbf{X}$ by $\mathbf{Y}$, and the projection of a vector $\mathbf{x}$ by $\mathbf{y}$ (note that $\mathbf{y}$ is two dimensional).
3. If $\mathbf{y} \neq 0$ "push" $\mathbf{y} \in \mathbf{Y}$ to the great circle, and denote the corresponding vector by $\mathbf{z} = \frac{\mathbf{y}}{\|\mathbf{y}\|}$. Denote the constructed set by $\mathbf{Z}$.
4. Partition $\mathbf{Z}$ into two clusters $\mathbf{Z}_1$ and $\mathbf{Z}_2$. Assign projections $\mathbf{y}$ with $\|\mathbf{y}\| = 0$ to $\mathbf{Z}_1$.
5. Generate the induced partition $\{\mathbf{X}_1, \mathbf{X}_2\}$ of $\mathbf{X}$ as follows:
$$\mathbf{X}_1 = \{\mathbf{x} \ : \ \mathbf{z} \in \mathbf{Z}_1\}, \text{ and } \mathbf{X}_2 = \{\mathbf{x} \ : \ \mathbf{z} \in \mathbf{Z}_2\}. \tag{4.12}$$

the best two dimensional approximation of the document set is the plane $\mathbf{P}$ that maximizes variance of the projections, then $\mathbf{P}$ is defined by two eigenvectors of the covariance matrix C corresponding to the largest eigenvalues λ_1 and λ_2. The "preserved" information under this projection is

$$\frac{\lambda_1 + \lambda_2}{\lambda_1 + \lambda_2 + \cdots + \lambda_n}. \tag{4.13}$$

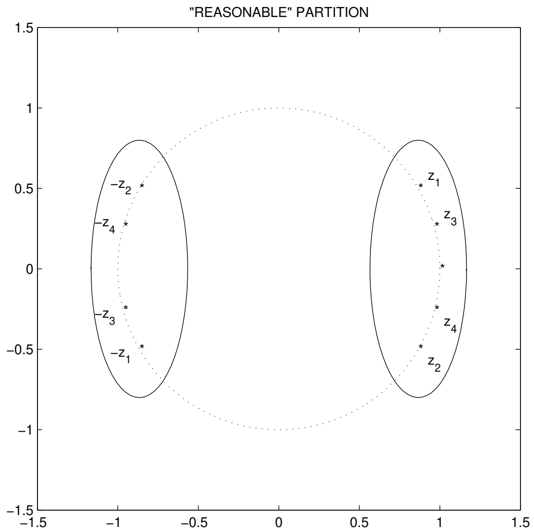
Note that the quantity given by (4.13) may be almost twice as much as the preserved information under the projection on the one dimensional line given by (4.8). As we show later in this section this may lead to a significant improvement over results provided by PDDP.

4.6.1 Two cluster partition of vectors on the unit circle

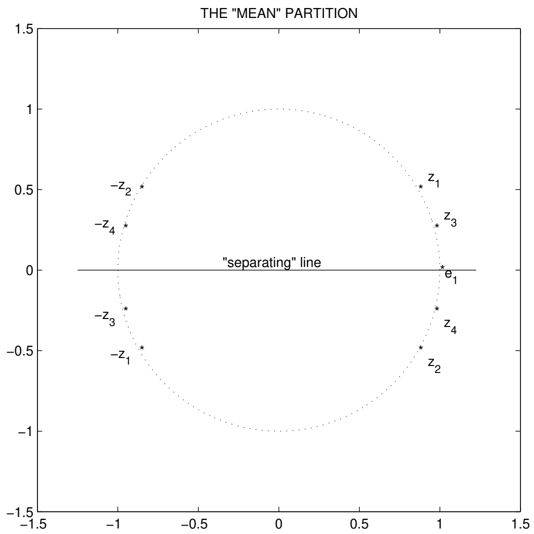
We now describe in detail Step 4 of Algorithm 4.6. Specifically we are concerned with the following problem: Given a set of unit vectors $\mathbf{Z} = \{\mathbf{z}_1, \ldots, \mathbf{z}_m\} \subset \mathbf{R}^2$ partition $\mathbf{Z}$ into two "optimal" clusters π_1^o and π_2^o.

A straightforward imitation of Boley's construction leads to the following solution: If $\mathbf{z} = \mathbf{z}_1 + \cdots + \mathbf{z}_m \neq 0$, then the line defined by $\mathbf{z}$ cuts the plane into two half–planes. The subset of $\mathbf{Z}$ that belongs to the "left" half–plane is denoted by $\mathbf{Z}_-$, and the subset of $\mathbf{Z}$ that belongs to the "right" half–plane is denoted by $\mathbf{Z}_+$. If $\mathbf{z}$ is zero, then, in order to generate the partition, we choose an arbitrary line passing through the origin.

Lack of robustness is, probably, the most prominent drawback of the suggested partitioning. Indeed, let $\{\mathbf{z}_1, \ldots, \mathbf{z}_m\}$ be a set of unit vectors concentrated around, say, the vector $\mathbf{e}_1 = (1, 0)^T$. If the set $\mathbf{Z}$ contains vectors $\{\mathbf{z}_1, \ldots, \mathbf{z}_m\}$ and their negatives $\{-\mathbf{z}_1, \ldots, -\mathbf{z}_m\}$, then $\mathbf{z} = 0$. This $\mathbf{z}$ does not do much to recover "good" clusters (although $\pi_1 = \{\mathbf{z}_1, \ldots, \mathbf{z}_m\}$, and $\pi_2 = \{-\mathbf{z}_1, \ldots, -\mathbf{z}_m\}$ looks like a reasonable partition, see figure below).



Things get worse when $\mathbf{e}_1$ is assigned to the vector set $\mathbf{Z}$, i.e., $\mathbf{Z} = \{\mathbf{z}_1, \ldots, \mathbf{z}_m, -\mathbf{z}_1, \ldots, -\mathbf{z}_m, \mathbf{e}_1\}$. Now $\mathbf{z} = \mathbf{e}_1$, and regardless of how "densely" the vectors $\{\mathbf{z}_1, \ldots, \mathbf{z}_m\}$ are concentrated around $\mathbf{e}_1$ the clusters $\mathbf{Z}_+$ and $\mathbf{Z}_-$ most probably contain vectors from both sets $\{\mathbf{z}_1, \ldots, \mathbf{z}_m\}$ and $\{-\mathbf{z}_1, \ldots, -\mathbf{z}_m\}$. This poor partition is illustrated by the figure below.



To define an optimal partition we measure the quality of a partition $\{\pi_1, \pi_2\}$ by the "spherical" objective function

$$\mathcal{Q}_s(\{\pi_1, \pi_2\}) = \left\| \sum_{\mathbf{z} \in \pi_1} \mathbf{z} \right\| + \left\| \sum_{\mathbf{z} \in \pi_2} \mathbf{z} \right\| \tag{4.14}$$

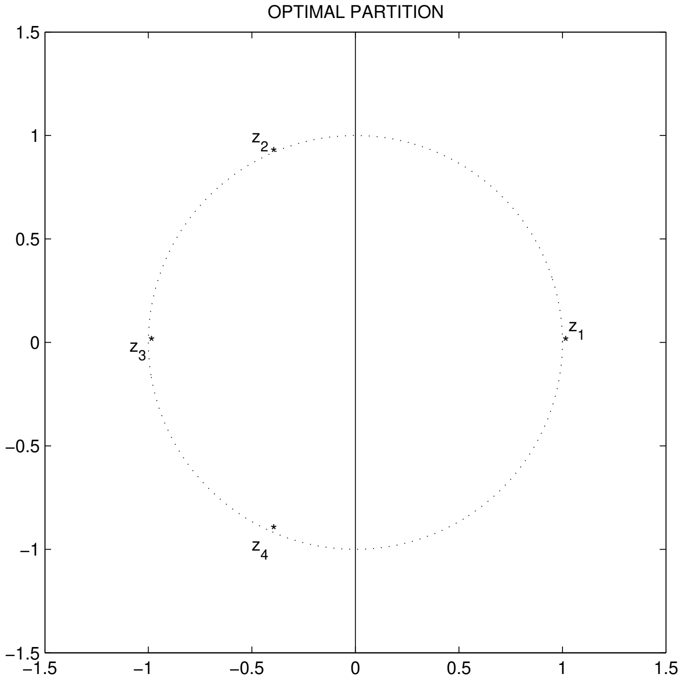
introduced by Dhillon and Modha [DM01]. Denote an optimal partition, that is one that maximizes (4.14), by $\{\pi_1^o, \pi_2^o\}$. It can be seen that for each optimal partition $\{\pi_1^o, \pi_2^o\}$ there is a nonzero vector $\mathbf{x}^o$ so that the clusters π_1^o and π_2^o are separated by the line passing through the origin and defined by $\mathbf{x}^o$ (see [DM01]).

Since each unit vector $\mathbf{z} \in \mathbf{R}^2$ can be uniquely represented by $e^{i\theta}$ with $0 \leq \theta < 2\pi$ the associated clustering problem is essentially one dimensional. We denote $\mathbf{z}_j$ by $e^{i\theta_j}$, and assume (without any loss of generality), that

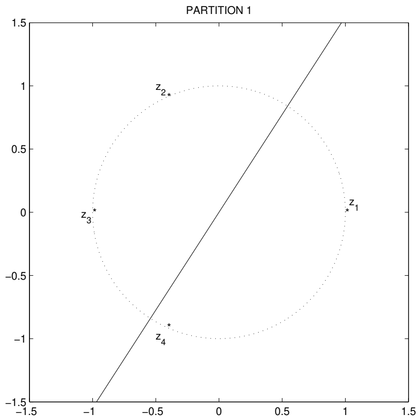
$$0 \leq \theta_1 \leq \theta_2 \leq \cdots \leq \theta_m < 2\pi.$$

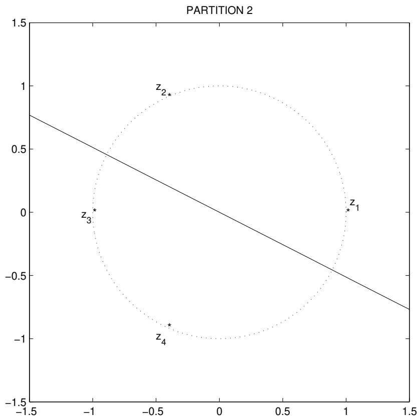
As in the case of clustering points on a line, it is tempting to assume that for some j a line passing through the origin and midway between $\mathbf{z}_j$ and $\mathbf{z}_{j+1}$ recovers the optimal partition. We show by the following example that this is not necessarily the case.

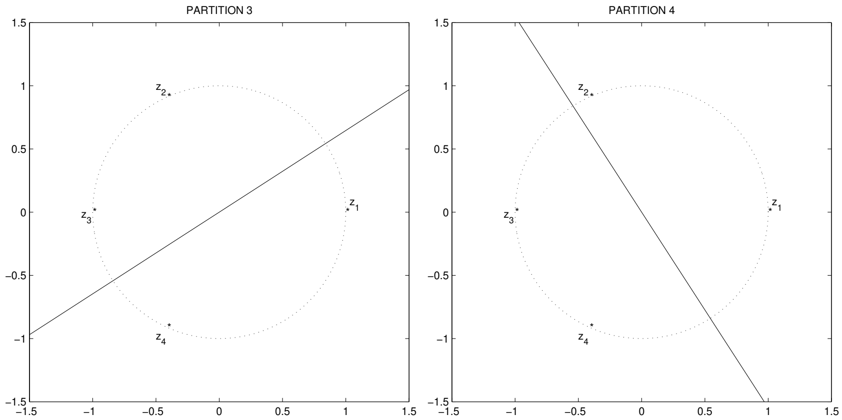
EXAMPLE 4.6.1 *Let* $\mathbf{z}_1 = (1, 0)^T$, $\mathbf{z}_2 = (\cos(2\pi/3 - \epsilon), \sin(2\pi/3 - \epsilon))^T$, $\mathbf{z}_3 = -\mathbf{z}_1$, *and* $\mathbf{z}_4 = (\cos(-2\pi/3 + \epsilon), \sin(-2\pi/3 + \epsilon))^T$. *It is easy to see that when* $\epsilon = 0$ *the optimal partition is* $\{\pi_1^o, \pi_k^o\} = \{\{\mathbf{z}_1\}, \{\mathbf{z}_2, \mathbf{z}_3, \mathbf{z}_4\}\}$ *with* $\mathcal{Q}_s(\{\pi_1^o, \pi_2^o\}) = 3$.



While a small positive ϵ (for example $\epsilon = \pi/36$) does not change the optimal partition, the four "midpoint" lines generate clusters containing two vectors each (a partition i is generated by a line passing through the origin and the midpoint between $\mathbf{z}_i$ and $\mathbf{z}_{i+1}$). These partitions do not contain the optimal partition $\{\pi_1^o, \pi_2^o\}$. We next present the four midpoint line partitions with $\epsilon = \pi/36$.







To analyze the failure of Example 4.6.1, and to propose a remedy we introduce the formal definition of the "left" and "right" half–planes generated by a vector $\mathbf{x}$, and describe a procedure that computes the optimal "separator" $\mathbf{x}^o$.

- For a nonzero vector $\mathbf{x} \in \mathbf{R}^2$ we denote by $\mathbf{x}^\perp$ the vector obtained from $\mathbf{x}$ by rotating it clockwise by an angle of 90^0, i.e.,

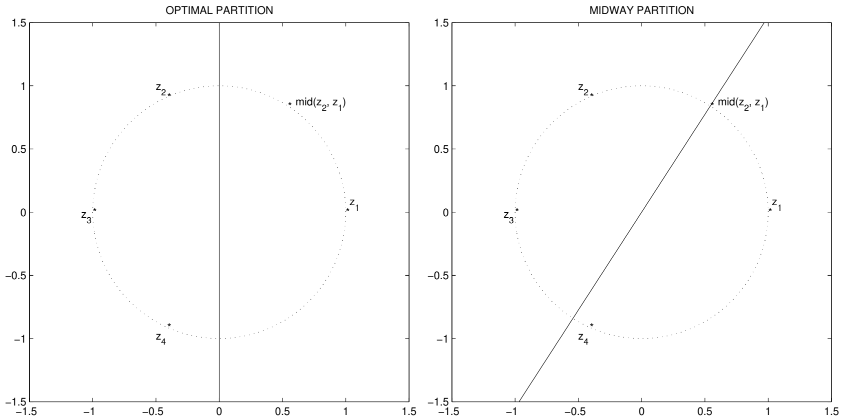
$$\mathbf{x}^\perp = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \mathbf{x}.$$

- For a nonzero vector $\mathbf{x} \in \mathbf{R}^2$, and a set of vectors $\mathbf{Z} = \{\mathbf{z}_1 \ldots, \mathbf{z}_m\} \subset \mathbf{R}^2$ define two subsets of $\mathbf{Z}$ — the "positive" $\mathbf{Z}_+(\mathbf{x}) = \mathbf{Z}_+$, and the "negative" $\mathbf{Z}_-(\mathbf{x}) = \mathbf{Z}_-$ as follows:

$$\mathbf{Z}_+ = \{\mathbf{z} : \mathbf{z} \in \mathbf{Z},\ \mathbf{z}^T\mathbf{x}^\perp \geq 0\},\ \text{and}\ \mathbf{Z}_- = \{\mathbf{z} : \mathbf{z} \in \mathbf{Z},\ \mathbf{z}^T\mathbf{x}^\perp < 0\}. \qquad (4.15)$$

- For two unit vectors $\mathbf{z}' = e^{i\theta'}$ and $\mathbf{z}'' = e^{i\theta''}$ we denote the "midway" vector $e^{i\frac{\theta'+\theta''}{2}}$ by $\mathtt{mid}\,(\mathbf{z}', \mathbf{z}'')$.

As the "optimal" separating line in Example 4.6.1 is rotated clockwise to $\mathtt{mid}\,(\mathbf{z}_2, \mathbf{z}_1)$ it crosses $\mathbf{z}_4$ changing cluster affiliation of this vector (see figures below).



This could have been prevented if instead of rotating the "optimal" separating line all the way to $\mathtt{mid}\,(\mathbf{z}_2, \mathbf{z}_1)$ one would rotate it to $\mathtt{mid}\,(\mathbf{z}_2, -\mathbf{z}_4)$. The "optimal" separating line and the line passing through $\mathtt{mid}\,(\mathbf{z}_2, -\mathbf{z}_4)$ and the origin generate identical partitions (see Table 4.14). In general, if the set $\mathbf{Z} = \{\mathbf{z}_1 =$

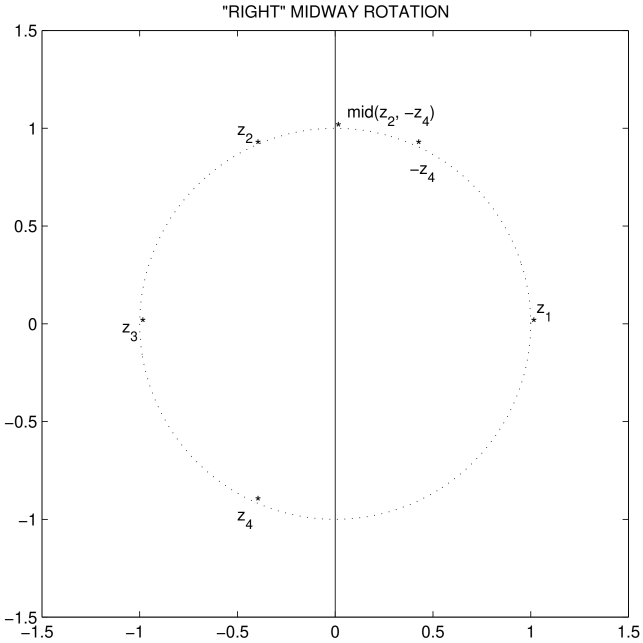


Table 4.14. Optimal partition

$e^{i\theta_1}, \ldots, \mathbf{z}_m = e^{i\theta_m}\}$ is symmetric with respect to the origin, (i.e., for each

$\mathbf{z}_i \in \mathbf{Z}$ there exists $\mathbf{z}_l \in \mathbf{Z}$ such that $\mathbf{z}_i = -\mathbf{z}_l$), then for

$$\mathbf{x}' = e^{i\theta'},\ \mathbf{x}'' = e^{i\theta''},\ \text{ with } \theta_j < \theta' \leq \theta'' < \theta_{j+1}.$$

the partitions

$$\{\mathbf{Z}_+(\mathbf{x}'), \mathbf{Z}_-(\mathbf{x}')\},\ \text{ and }\ \{\mathbf{Z}_+(\mathbf{x}''), \mathbf{Z}_-(\mathbf{x}'')\}$$

are identical. This observation suggests the following simple procedure for recovering the optimal partition $\{\pi_1^o, \pi_2^o\}$:

1. Let $\mathbf{W} = \{\mathbf{w}_1, \ldots, \mathbf{w}_m, \mathbf{w}_{m+1}, \ldots, \mathbf{w}_{2m}\}$ be a set of two dimensional vectors defined as follows:

$$\mathbf{w}_i = \mathbf{z}_i \text{ for } i = 1, \ldots, m, \text{ and } \mathbf{w}_i = -\mathbf{z}_i \text{ for } i = m+1, \ldots, 2m.$$

2. If needed reassign indices so that

$$\mathbf{w}_j = e^{i\theta_j},\ \text{and } 0 \leq \theta_1 \leq \theta_2 \leq \cdots \leq \theta_{2m} < 2\pi.$$

3. With each subscript j associate a partition $\{\pi_1^j, \pi_2^j\}$ of $\mathbf{Z}$ as follows:
   (a) set $\mathbf{x} = \dfrac{\mathbf{w}_j + \mathbf{w}_{j+1}}{2}$
   (b) set $\pi_1^j = \mathbf{Z}_+(\mathbf{x})$, and $\pi_2^j = \mathbf{Z}_-(\mathbf{x})$.

   Note that:
   (a) The indices j and $j + m$ generate identical partitions. We, therefore, have to consider at most m distinct partitions generated by $j = 1, \ldots, m$.
   (b) The optimal partition that maximizes (4.14) is among the generated ones.
4. With each partition $\{\pi_1^j, \pi_2^j\}$ associate the value of the objective function $Q_s^j = \mathcal{Q}_s\left(\{\pi_1^j, \pi_2^j\}\right)$. Let $Q_s^k = \max\limits_{j=1,\ldots,m} Q_s^j$, then the desired partition of $\mathbf{Z}$ is $\{\pi_1^o, \pi_2^o\} = \{\pi_1^k, \pi_2^k\}$.

4.6.2 Clustering with sPDDP

In what follows, we display clustering results for the document collection DC described in Section 4.3. To compare the results with those presented in Section 4.4, we select the 600 best q_0 quality terms (see Equation (4.9)) to build document vectors. The confusion matrix for the three cluster partition generated by sPDDP is given in Table 4.15 below. We remark that the confusion matrix is a significant improvement over the result presented in Table 4.3. A subsequent application of the means algorithm to the partition generated by sPDDP leads to a minor improvement of the result both in terms of confusion matrices, as well as in terms of the objective function $\mathcal{Q}_2$ (see Table 4.16).

	DC0	DC1	DC2
cluster 0	1000	3	1
cluster 1	8	10	1376
cluster 2	25	1447	21
empty documents			
cluster 3	0	0	0

Table 4.15. sPDDP generated initial confusion matrix with 68 misclassified documents, and the partition quality $\mathcal{Q}_2 = 3630.97$

	DC0	DC1	DC2
cluster 0	1023	21	10
cluster 1	1	3	1370
cluster 2	9	1436	18
empty documents			
cluster 3	0	0	0

Table 4.16. Means generated final confusion matrix with 62 misclassified documents, the partition quality $\mathcal{Q}_2 = 3630.38$.

Table 4.17 summarizes clustering results for the sPDDP algorithm combined with the means clustering algorithm for different choices of index terms (all term selections are based on the q_0 criterion). Note that while the combination of the

	documents misclassified by	
# of terms	pddp	means
300	228	100
400	88	80
500	76	62
600	68	62

Table 4.17. Number of misclassified documents for term selection based on q_0

PDDP and the means algorithms "collapses" when the number of selected terms drops below 600 (see Table 4.9), the combination of the sPDDP and the means algorithms performs reasonably well even when the number of selected terms is only 300.

Clustering results for different choices of index terms based on the q_1 criterion are similar to those presented above. The results are summarized in Table 4.18.

	documents misclassified by	
# of terms	pddp	means
300	224	101
400	91	86
500	74	71
600	71	68

Table 4.18. Number of misclassified documents for term selection based on q_1

4.7 Future Research

This chapter presents preliminary results concerning two information retrieval related problems:

1. feature selection, and
2. document clustering.

We plan to further investigate profile based term selection techniques as well as techniques based on term distribution across documents [GK02], and to run term selection experiments on larger document collections.

Clustering experiments with seven different objective functions reported by Zhao and Karypis [ZK02] indicate that the objective function based on cosine similarity (and used in [DM01]) “leads to the best solutions irrespective of the number of clusters for most of the data sets.” We intend to combine the Spherical Principal Direction Divisive Partitioning algorithm with the modification of the spherical k–means algorithm recently reported by [DGK02].

The Spherical Principal Directions Divisive Partitioning algorithm introduced in the chapter utilizes the unit norm constraint imposed on document vectors. In many data mining applications, vectors representing data are normalized. For example:

1. In bioinformatics applications, fingerprint data is transformed to have mean zero and variance one, a fixed l_2 norm, or a fixed l_∞ norm [SS02].
2. In contemporary $k-$means type frameworks for word clustering, a word is represented by a discrete probability distribution, i.e., by a vector of l_1 unit norm [DMK02], [BB02], [ST01].
3. The $n-$gram technique leads to a vector space model where document vectors have l_1 unit norm [Dam95]. The technique is proved to be useful in information retrieval applications [PN96], as well as in bioinformatics [GKSR$^+$02].

We plan to derive and investigate clustering algorithms utilizing special constraints (among them l_p constraints mentioned above) imposed upon vector data sets.

While this chapter deals with a vector space model based on word occurrence across documents, additional research directions include clustering of vectors whose components are the frequencies of their distinct constituent $n-$grams [Dam95]. The $n-$gram representation of a document is sparse, simple, and language independent. The sparsity of the vectors lends itself to processing with numerical linear algebra tools, although the matrices themselves may be much larger. We believe that best clustering results may be achieved by combining a number of different techniques.

Acknowledgments: The authors thank Robert Libier for valuable suggestions that improved exposition of the results. The work of Dhillon was supported by NSF grant No. ACI-0093404, and grant no. 003658-0431-2001 from the Texas Higher Education Coordinating Board. The work of Kogan and Nicholas was supported in part by the US Department of Defense.

References

[BB99] M.W. Berry and M. Browne.*Understanding Search Engines: Mathematical Modeling and Text Retrieval*.SIAM, Philadelphia, PA, 1999.

[BB02] P. Berkhin and J.D. Becher.Learning simple relations: Theory and applications.In *Proc. Second SIAM International Conference on Data Mining*, pages 410–436, Arlington, April 2002.

[BGG+99a] D. Boley, M. Gini, R. Gross, E.-H. Han, K. Hastings, G. Karypis, V. Kumar, B. Mobasher, and J. Moore.Document categorization and query generation on the World Wide Web using WebACE.*AI Review*, 13(5,6):365–391, 1999.

[BGG+99b] D. Boley, M. Gini, R. Gross, E.-H. Han, K. Hastings, G. Karypis, V. Kumar, B. Mobasher, and J. Moore.Partitioning-based clustering for web document categorization.*Decision Support Systems*, 27(3):329–341, 1999.

[Bol98] D.L. Boley.Principal direction divisive partitioning.*Data Mining and Knowledge Discovery*, 2(4):325–344, 1998.

[Dam95] M. Damashek.Gauging similarity with n-grams: Language-independent categorization of text.*Science*, 267:843–848, 1995.

[DGK02] I. S. Dhillon, Y. Guan, and J. Kogan.Refining clusters in high-dimensional text data.In I. S. Dhillon and J. Kogan, editors, *Proceedings of the Workshop on Clustering High Dimensional Data and its Applications at the Second SIAM International Conference on Data Mining*, pages 71–82. SIAM, 2002.

[DHS01] R.O. Duda, P.E. Hart, and D.G. Stork.*Unsupervised Learning and Clustering*.Wiley InterScience, 2001.

[DM01] I.S. Dhillon and D.S. Modha.Concept decompositions for large sparse text data using clustering.*Machine Learning*, 42(1):143–175, January 2001.Also appears as IBM Research Report RJ 10147, July 1999.

[DMK02] I.S. Dhillon, S. Malella, and R. Kumar.Enhanced word clustering for hierarchical text classification.In *KDD-2002*, 2002.

[For65] E. Forgy.Cluster analysis of multivariate data: Efficiency vs. interpretability of classifications.*Biometrics*, 21(3):768, 1965.

[G.94] Grefenstette G.*Explorations in Automatic Thesaurus Discovery*.Kluwer, 1994.

[GK02] E. Gendler and J. Kogan.Index terms selection for clustering large text data.In M.W.Berry, editor, *Proceedings of the Workshop on Text Mining at the Second SIAM International Conference on Data Mining*, pages 87–94, 2002.

[GKSR+02] M. Ganapathiraju, J. Klein-Seetharaman, R. Rosenfeld, J. Carbonell, and R. Reddy.Rare and frequent n-grams in whole-genome protein sequences.In *RECOMB'02: The Sixth Annual International Conference on Research in Computational Molecular Biology*, 2002.

[Kog01a] J. Kogan.Clustering large unstructured document sets.In M.W.Berry, editor, *Computational Information Retrieval*, pages 107–117. SIAM, 2001.

[Kog01b] J. Kogan.Means clustering for text data.In M.W.Berry, editor, *Proceedings of the Workshop on Text Mining at the First SIAM International Conference on Data Mining*, pages 57–54, 2001.

[Kog02] J. Kogan.Computational information retrieval.In H. R. Lerche, editor, *Springer-Verlag Lecture Notes in Contributions to Statistics*, 2002. To appear.

[PN96] C. Pearce and C. Nicholas.TELLTALE: Experiments in a dynamic hypertext environment for degraded and multilingual data.*Journal of the American Society for Information Science*, 47:263–275, 1996.

[Por80] M.F. Porter.An algorithm for suffix stripping.*Program*, 14:130–137, 1980.

[SM83] G. Salton and M.J. McGill.*Introduction to Modern Information Retrieval*.McGraw-Hill, 1983.

[SP95] H. Schütze and J. Pedersen.Information retrieval based on word senses.In *Proceedings of the 4th Annual Symposium on Document Analysis and Information Retrieval*, pages 161–175, Las Vegas, NV, 1995.

[SS02] R. Shamir and R. Sharan.Algorithmic approaches to clustering gene expression data.In T. Jiang, T. Smith, Y. Xu, and M. Q. Zhang, editors, *Current Topics in Computational Molecular Biology*, pages 269–300. MIT Press, 2002.

[ST01] N. Slonim and N. Tishby.The power of word clusters for text classification.In *23rd European Colloquium on Information Retrieval Research (ECIR)*, Darmstadt, 2001.

[ZK02] Y. Zhao and G. Karypis.Comparison of agglomerative and partitional document clustering algorithms.In I. S. Dhillon and J. Kogan, editors, *Proceedings of the Workshop on Clustering High Dimensional Data and its Applications at the Second SIAM International Conference on Data Mining*, pages 83–93. SIAM, 2002.