Chapter 10

Weighting Method for Feature Selection in K-Means

Joshua Zhexue Huang

The University of Hong Kong

Jun Xu

The University of Hong Kong

Michael Ng

Hong Kong Baptist University

Yunming Ye

Harbin Institute of Technology, China

10.1 Introduction 193
10.2 Feature Weighting in k-Means 194
10.3 W-k-Means Clustering Algorithm 197
10.4 Feature Selection 198
10.5 Subspace Clustering with k-Means 200
10.6 Text Clustering 201
10.7 Related Work 204
10.8 Discussions 207
Acknowledgment 207
References 208

10.1 Introduction

The k-means type of clustering algorithms [13, 16] are widely used in real-world applications such as marketing research [12] and data mining due to their efficiency in processing large datasets. One unavoidable task of using k-means in real applications is to determine a set of features (or attributes). A common practice is to select features based on business domain knowledge and data exploration. This manual approach is difficult to use, time consuming, and frequently cannot make a right selection. An automated method is needed to solve the feature selection problem in k-means.

In this chapter, we introduce a recent development of the k-means algorithm that can automatically determine the important features in the k-means clus-

tering process [14]. This new algorithm is called W-k-means. In this algorithm a new step is added to the standard k-means clustering process to calculate the feature weights from the current partition of data in each iteration. The weight of a feature is determined by the sum of the within-cluster dispersions of the feature. The larger the sum, the smaller the feature weight. The weights produced by the W-k-means algorithm measure the importance of the corresponding features in clustering. The small weights reduce or eliminate the effect of insignificant (or noisy) features. Therefore, the feature weights can be used in feature selection. Since the k-means clustering process is not fundamentally changed in W-k-means, the efficiency and convergency of the clustering process remain.

A further extension of this approach is to calculate a weight for each feature in each cluster [4]. This is called subspace k-means clustering because the important features in each cluster identify the subspace in which the cluster is discovered. Since the subsets of important features are different in different clusters, subspace clustering is achieved. Subspace clustering has wide applications in text clustering, bio-informatics, and customer behavior analysis, where high-dimensional data are involved. In this chapter, subspace k-means clustering is also discussed.

10.2 Feature Weighting in k-Means

Given a dataset $\boldsymbol{X}$ with M records and N features, the k-means clustering algorithm [16] searches for a partition of $\boldsymbol{X}$ into k clusters that minimizes the sum of the within-cluster dispersions of all features. The clustering process is conducted as follows:

1. Randomly select k distinct records as the initial cluster centers.
2. For each record in $\boldsymbol{X}$, calculate the distances between the record and each cluster center, and assign the record to the cluster with the shortest distance.
3. Repeat the above step until all records have been assigned to clusters. For each cluster, compute a new cluster center as the mean (average) of the feature values.
4. Compare the new cluster centers with the previous centers. If the new centers are the same as the previous centers, stop the clustering process; otherwise, go back to Step 2.

In the above standard k-means clustering process, all features are treated the same in the calculation of the distances between the data records and the

cluster centers. The importance of different features is not distinguishable. The formal presentation of the k-means clustering algorithm can be found in [13].

To identify the importance of different features, a weight can be assigned to each feature in the distance calculation. As such, the feature with a large weight will have more impact on determining the cluster a record is assigned to. Since the importance of a feature is determined by its distribution in the dataset, the feature weights are data dependent.

To automatically determine the feature weights, we add one step to the standard k-means clustering process to calculate the feature weights from the current partition of the data in each iteration. During the clustering process, weights are updated automatically until the clustering process converges. Then, the final weights of the features can indicate which features are important in clustering the data and which are not.

Formally, the process is to minimize the following objective function:

$$P(U, Z, W) = \sum_{l=1}^{k} \sum_{i=1}^{M} \sum_{j=1}^{N} u_{i,l} w_j^{\beta} d(x_{i,j}, z_{l,j}) \tag{10.1}$$

subject to

$$\begin{cases} \sum\limits_{l=1}^{k} u_{i,l} = 1, \quad 1 \le i \le M \\ u_{i,l} \in \{0, 1\}, 1 \le i \le M, 1 \le l \le k \\ \sum\limits_{j=1}^{N} w_j = 1, \quad 0 \le w_j \le 1 \end{cases} \tag{10.2}$$

where

- U is an $M \times k$ partition matrix, $u_{i,l}$ is a binary variable, and $u_{i,l} = 1$ indicates that record i is allocated to cluster l.
- $Z = \{Z_1, Z_2, ..., Z_k\}$ is a set of k vectors representing the k-cluster centers.
- $W = [w_1, w_2, ..., w_N]$ is a set of weights.
- $d(x_{i,j}, z_{l,j})$ is a distance or dissimilarity measure between object i and the center of cluster l on the jth feature. If the feature is numeric, then

$$d(x_{i,j}, z_{l,j}) = (x_{i,j} - z_{l,j})^2 \tag{10.3}$$

If the feature is categorical, then

$$d(x_{i,j}, z_{l,j}) = \begin{cases} 0 \ (x_{i,j} = z_{l,j}) \\ 1 \ (x_{i,j} \neq z_{l,j}) \end{cases} \tag{10.4}$$

- β is a parameter.

The above optimization problem can be solved by iteratively solving the following three minimization problems:

1. P_1: Fix $Z = \hat{Z}$ and $W = \hat{W}$; solve the reduced problem $P(U, \hat{Z}, \hat{W})$.
2. P_2: Fix $U = \hat{U}$ and $W = \hat{W}$; solve the reduced problem $P(\hat{U}, Z, \hat{W})$.
3. P_3: Fix $U = \hat{U}$ and $Z = \hat{Z}$; solve the reduced problem $P(\hat{U}, \hat{Z}, W)$.

P_1 is solved by

$$\begin{cases} u_{i,l} = 1 \text{ if } \sum_{j=1}^{N} w_j^{\beta} d(x_{i,j}, z_{l,j}) \leq \sum_{j=1}^{N} w_j^{\beta} d(x_{i,j}, z_{t,j}) \text{ for } 1 \leq t \leq k \\ u_{i,t} = 0 \text{ for } t \neq l \end{cases} \tag{10.5}$$

and P_2 is solved for the numeric features by

$$z_{l,j} = \frac{\sum_{i=1}^{M} u_{i,l}\, x_{i,j}}{\sum_{i=1}^{M} u_{i,l}} \qquad \text{for } 1 \leq l \leq k \text{ and } 1 \leq j \leq N \tag{10.6}$$

If the feature is categorical, then

$$z_{l,j} = a_j^r \tag{10.7}$$

where a_j^r is the mode of the feature values in cluster l [13].

The solution to P_3 is given in the following theorem.

Theorem 1. Let $U = \hat{U}$ and $Z = \hat{Z}$ be fixed,
(i) When $\beta > 1$ or $\beta \leq 0$, $P(\hat{U}, \hat{Z}, W)$ is minimized iff

$$\hat{w}_j = \begin{cases} 0 & \text{if } D_j = 0 \\ \dfrac{1}{\sum_{t=1}^{h} \left[\frac{D_j}{D_t}\right]^{\frac{1}{\beta-1}}} & \text{if } D_j \neq 0 \end{cases} \tag{10.8}$$

where

$$D_j = \sum_{l=1}^{k} \sum_{i=1}^{M} \hat{u}_{i,l} d(x_{i,j}, z_{l,j}) \tag{10.9}$$

and h is the number of features with $D_j \neq 0$.

(ii) When $\beta = 1$, $P(\hat{U}, \hat{Z}, W)$ is minimized iff

$$\hat{w}_{j'} = 1 \quad \text{and} \quad \hat{w}_j = 0, \quad j \neq j'$$

where $D_{j'} \leq D_j$ for all j.

The proof is given in [14].

Theorem 1 shows that, given a data partition, a larger weight is assigned to a feature with a smaller sum of the within-cluster dispersions and a smaller weight to a feature with a larger sum of the within-cluster dispersions. Therefore, the feature weight is reversely proportional to the sum of the within-cluster dispersions of the feature.

The real weight w_j^β of feature x_j in the distance calculation (see (1.5)) is also dependent on the value of β. In using W-k-means, we can choose either $\beta < 0$ or $\beta > 1$ for the following reasons:

- When $\beta = 0$, W-k-means is equivalent to k-means.
- When $\beta = 1$, w_j is equal to 1 for the smallest value of D_j. The other weights are equal to 0. Although the objective function is minimized, the clustering is made by the selection of one variable. It may not be desirable for high-dimensional clustering problems.
- When $0 < \beta < 1$, the larger D_j, the larger w_j, and similarly for w_j^β. This is against the variable weighting principal, so we cannot choose $0 < \beta < 1$.
- When $\beta > 1$, the larger D_j, the smaller w_j and the smaller w_j^β. The effect of variable x_j with large D_j is reduced.
- When $\beta < 0$, the larger D_j, the larger w_j. However, w_j^β becomes smaller and has less weighting to the variable in the distance calculation because of negative β.

10.3 W-k-Means Clustering Algorithm

The algorithm to solve (10.1) is an extension to the standard k-means algorithm [13, 21].

Algorithm - (The W-k-means algorithm)

Step 1. Randomly choose an initial $Z^0 = \{Z_1, Z_2, ..., Z_k\}$ and randomly generate a set of initial weights $W^0 = [w_1^0, w_2^0, ..., w_N^0]$ ($\sum_{j=1}^{N} w_j = 1$). Determine U^0 such that $P(U^0, Z^0, W^0)$ is minimized. Set $t = 0$;

Step 2. Let $\hat{Z} = Z^t$ and $\hat{W} = W^t$, solve problem $P(U, \hat{Z}, \hat{W})$ to obtain U^{t+1}. If $P(U^{t+1}, \hat{Z}, \hat{W}) = P(U^t, \hat{Z}, \hat{W})$, output $(U^t, \hat{Z}, \hat{W})$ and stop; otherwise, go to Step 3;

Step 3. Let $\hat{U} = U^{t+1}$ and $\hat{W} = W^t$, solve problem $P(\hat{U}, Z, \hat{W})$ to obtain Z^{t+1}. If $P(\hat{U}, Z^{t+1}, \hat{W}) = P(\hat{U}, Z^t, \hat{W})$, output $(\hat{U}, Z^t, \hat{W})$ and stop; otherwise, go to Step 4;

Step 4. Let $\hat{U} = U^{t+1}$ and $\hat{Z} = Z^{t+1}$, solve problem $P(\hat{U}, \hat{Z}, W)$ to obtain W^{t+1}. If $P(\hat{U}, \hat{Z}, W^{t+1}) = P(\hat{U}, \hat{Z}, W^t)$, output $(\hat{U}, \hat{Z}, W^t)$ and stop; otherwise, set $t = t + 1$ and go to Step 2.

Theorem 2. The above algorithm converges to a local minimal solution in a finite number of iterations.

The proof is given in [14].

Since the W-k-means algorithm is an extension to the k-means algorithm by adding a new step to calculate the variable weights in the iterative process, it does not seriously affect the scalability in clustering large data; therefore, it is suitable for data mining applications. The computational complexity of the algorithm is $O(tNMk)$, where t is the total number of iterations required for performing Step 2, Step 3 and Step 4; k is the number of clusters; N is the number of features; and M is the number of records.

10.4 Feature Selection

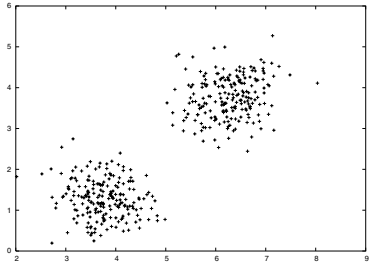
One of the drawbacks of the standard k-means algorithm is that it treats all features equally when deciding the cluster memberships. This is not desirable if the data contain a large number of diverse features. A cluster structure in a high-dimensional dataset is often confined to a subset of features rather than the entire feature set. Inclusion of all features can only obscure the discovery of the cluster structure.

The W-k-means clustering algorithm can be used to select the subset of features for clustering in real-world applications. In doing so, the clustering work can be divided in the following steps. The first step is to use W-k-means to cluster the dataset or a sample of the dataset to produce a set of weights. The second step is to select a subset of features according to the weight values and remove the unselected features from the dataset. The third step is to use W-k-means or another clustering algorithm to cluster the dataset to produce the final clustering result.

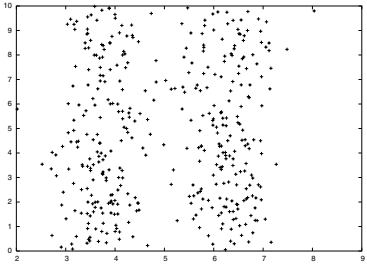
Figure 10.1 shows a dataset with three features (x_1, x_2, x_3) and two clusters in the subset of features (x_1, x_2). Feature x_3 is noise in a uniform distribution. We can see the two clusters in the plot of Figure 10.1(a) but cannot see any cluster structure in the plots of Figure 10.1(b) and Figure 10.1(c). If we did not know that the two clusters were existing in the subset of features (x_1, x_2), we would find it difficult to discover them from the dataset using the standard k-means algorithm. However, we can use W-k-means to cluster this

dataset and obtain the weights of the three features as 0.47, 0.40, and 0.13, respectively. From these weights, we can easily identify the first two features (x_1, x_2) as important features. After removing the data of feature x_3, we can run the standard k-means algorithm to discover the two clusters from the subset of the features (x_1, x_2) as shown in Figure 10.1(d).

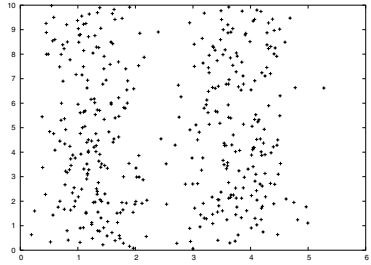
In fact, we can get the final result of Figure 10.1(d) directly from the first run of W-k-means in this simple example. Real datasets often have features in the hundreds and records in the hundreds of thousands, such as the customer datasets in large banks. In such situations, several runs of W-k-means are needed to identify the subset of important features.



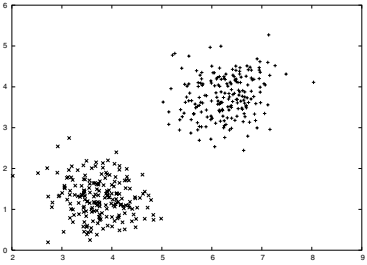
(a) Two clusters in the subset of features x_1, x_2.



(b) Plot of the subset of features x_1, x_3.



(c) Plot of the subset of features x_2, x_3.



(d) Two discovered clusters plotted in the subset of features x_1, x_2.

FIGURE 10.1: Feature selection from noise data.

10.5 Subspace Clustering with k-Means

Subspace clustering refers to the process of identifying clusters from subspaces of data, with each subspace being defined by a subset of features. Different clusters are identified from different subspaces of data. Subspace clustering is required when clustering high-dimensional data such as those in text mining, bio-informatics, and e-commerce.

Subspace clustering can be achieved by feature weighting in k-means. Instead of assigning a weight to each feature for the entire dataset, we assign a weight to each feature in each cluster. As such, if there are N features and k clusters, we will obtain $N \times k$ weights. This is achieved by rewriting the objective function (10.1) as follows:

$$P(U, Z, W) = \sum_{l=1}^{k} \sum_{i=1}^{M} \sum_{j=1}^{N} u_{i,l} w_{l,j}^{\beta} d(x_{i,j}, z_{l,j}) \tag{10.10}$$

subject to

$$\begin{cases} \sum\limits_{l=1}^{k} u_{i,l} = 1, \ \ 1 \le i \le M \\ u_{i,l} \in \{0, 1\}, 1 \le i \le M, \ \ 1 \le l \le k \\ \sum\limits_{j=1}^{N} w_{lj} = 1, \ \ 0 \le w_{lj} \le 1 \end{cases} \tag{10.11}$$

where W is a $k \times N$ weight matrix and the other notations are the same as in (10.1).

In a similar fashion, (10.10) can be reduced to three subproblems that are solved iteratively.

The subproblem P_1 is solved by

$$\begin{cases} u_{i,l} = 1 \text{ if } \sum\limits_{j=1}^{N} w_{lj}^{\beta} d(x_{i,j}, z_{l,j}) \le \sum\limits_{j=1}^{N} w_{lj}^{\beta} d(x_{i,j}, z_{t,j}) \text{ for } 1 \le t \le k \\ u_{i,t} = 0 \text{ for } t \ne l \end{cases} \tag{10.12}$$

The subproblem P_2 is solved with (10.6) or (10.7), depending on the data types.

The solution to the subproblem P_3 is given in the following theorem.

Theorem 3. Let $U = \hat{U}$ and $Z = \hat{Z}$ be fixed. When $\beta > 1$ or $\beta \le 0$, $P(\hat{U}, \hat{Z}, W)$ is minimized iff

$$\hat{w}_{lj} = \frac{1}{\sum_{t=1}^{N} \left[\frac{D_{lj}}{D_{lt}}\right]^{\frac{1}{\beta-1}}} \tag{10.13}$$

where

$$D_{lj} = \sum_{i=1}^{M} \hat{u}_{i,l} d(x_{i,j}, z_{l,j}) \tag{10.14}$$

and N is the number of features with $D_{lj} > 0$.

In subspace clustering, if $D_{lj} = 0$, we cannot simply assign a weight 0 to feature j in cluster l. $D_{lj} = 0$ means all values of feature j are the same in cluster l. In fact, $D_{lj} = 0$ indicates that feature j may be an important feature in identifying cluster l. $D_{lj} = 0$ often occurs in real-world data such as text data and supplier transaction data. To solve this problem, we can simply add a small constant σ to the distance function to make $\hat{w}_{lj}$ always computable, i.e.,

$$D_{lj} = \sum_{i=1}^{M} \hat{u}_{i,l}(d(x_{i,j}, z_{l,j}) + \sigma) \tag{10.15}$$

In practice, σ can be chosen as the average dispersion of all features in the dataset. It can be proved that the subspace k-means clustering process converges [4].

10.6 Text Clustering

A typical application of subspace clustering is text mining. In text clustering, text data are usually represented in the vector space model (VSM). A set of documents is converted to a matrix where each row indicates a document and each column represents a term or word in the vocabulary of the document set. Table 10.1 is a simplified example of text data representation in VSM. Each column corresponds to a term and each line represents a document. Each entry value is the frequency of the corresponding term in the related document.

If a set of text documents contains several classes, the documents related to a particular class, for instance *sport*, are categorized by a particular subset of terms, corresponding to a subspace of the vocabulary space. Different document classes are categorized by different subsets of terms, i.e., different subspaces. For example, the subset of terms describing the *sport* class is different from the subset of terms describing the *music* class. As such, k-means subspace clustering becomes useful for text data because different clusters can be identified from different subspaces through the weights of the terms.

TABLE 10.1: A simple example of text representation.

	t_0	t_1	t_2	t_3	t_4
x_0	1	2	3	0	6
x_1	2	3	1	0	6
x_2	3	1	2	0	6
x_3	0	0	1	3	2
x_4	0	0	2	1	3
x_5	0	0	3	2	1

TABLE 10.2: Summary of the six text datasets.

Dataset	Source	n_d	Dataset	Source	n_d
A2	alt.atheism	100	B2	talk.politics.mideast	100
	comp.graphics	100		talk.politics.misc	100
A4	comp.graphics	100	B4	comp.graphics	100
	rec.sport.baseball	100		comp.os.ms-windows	100
	sci.space	100		rec.autos	100
	talk.politics.mideast	100		sci.electronics	100
A4-U	comp.graphics	120	B4-U	comp.graphics	120
	rec.sport.baseball	100		comp.os.ms-windows	100
	sci.space	59		rec.autos	59
	talk.politics.mideast	20		sci.electronics	20

Besides, the weights can also be used to select the key words for semantic representations of clusters.

10.6.1 Text Data and Subspace Clustering

Table 10.2 lists the six datasets built from the popular *20-Newsgroups* collection.[1] The six datasets have different characteristics in sparsity, dimensionality, and class distribution. The classes and the number of documents in each class are given in the columns "Source" and "n_d." The classes in the datasets A2 and A4 are semantically apart, while the classes in the datasets B2 and B4 are semantically close. Semantically close classes have more overlapping words. The number of documents in the datasets A4-U and B4-U are different, indicating unbalanced class distributions.

These datasets were preprocessed using the *Bow* toolkit.[2] The preprocessing steps included removing the headers, the stop words, and the words that occurred in less than three documents or greater than the average number of documents in each class, as well as stemming the remaining words with the Porter stemming function. The standard $tf \cdot idf$ term weighting was used to represent the document vector.

Table 10.3 shows the comparisons of accuracy in clustering these datasets with the subspace k-means, the standard k-means, and four subspace clustering algorithms: PROCLUS [1], HARP [23], COSA [10], and LAC [8]. The

TABLE 10.3: Comparisons of accuracies of the subspace k-means with the standard k-mean and other four subspace clustering algorithms.

	A2	B2	A4	B4	A4-U	B4-U
Subspace k-means	0.9599	0.9043	0.9003	0.8631	0.9591	0.9205
Standard k-means	0.895	0.735	0.6	0.5689	0.95	0.8729
PROCLUS	0.7190	0.6604	0.6450	0.4911	0.5239	0.5739
HARP	0.8894	0.6020	0.5073	0.3840	0.4819	0.3364
COSA	0.5781	0.5413	0.3152	0.3621	0.4159	0.3599
LAC	0.9037	0.7981	0.6721	0.5816	0.9473	0.7363

	weight intervals	word number
0~1:	(0,1e-08]	8
1~2:	(1e-08,1e-07]	280
2~3:	(1e-07,1e-06]	433
3~4:	(1e-06,1e-05]	188
4~5:	(1e-05,1)	32

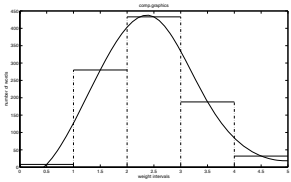


FIGURE 10.2: Distribution of words in different ranges of weights.

accuracy is calculated as the number of correctly classified documents divided by the total number of documents in a dataset. We can see that the subspace k-means performed better than the standard k-means and the other four subspace clustering algorithms on all datasets. This is due to the subspace nature of the text clusters, so the subspace k-means is more suitable in text clustering.

10.6.2 Selection of Key Words

Another advantage of using the subspace k-means in text clustering is that the weights produced by the algorithm can be used to identify the important terms or words in each cluster. The higher the weight value in a cluster, the more important the term feature in discovering the cluster. We can divide the range of the weight values into intervals and plot the distribution of features against the weight intervals as shown in Figure 10.2.

After the distribution is obtained, we remove the terms with the extremely large weights because they correspond to the terms with zero frequency in the cluster, i.e., the term did not occur in the cluster. Few terms with the extremely large weights correspond to the terms with equal frequency in each document of the cluster. Such terms can be easily identified in postprocessing.

Taking dataset B4 as an example, after preprocessing we got 1,322 feature words. We used the subspace k-means to cluster it into four clusters. Each cluster has more than 300 words with zero frequency. These words were removed from the clusters.

Figure 10.2 shows distribution of the remaining words in cluster *Computer*

Graphics of the dataset B4 against the weight intervals. Since we limit the sum of the weights for all features in a cluster to 1, the weights for most words are relatively small. Using a weight threshold, we identified 220 words with relatively larger weights. This is less than 17% of the total words. These are the words categorizing the cluster. From these words, we need to identify a few that will enable use to interpret the cluster.

Figure 10.3 show the plots of the term weights in four clusters. The horizontal axis is the index of the 220 words and the vertical lines indicate the values of the weights. We can observe that each cluster has its own subset of key words because the lines do not have big overlaps in different clusters. The classes *Computer Graphics* and *Microsoft Windows* overlap a little, which indicates that the semantics of the two classes are close to each other. Similarly, the classes *Autos* and *Electronics* are close.

We extracted 10 words from each cluster, which had the largest weights and were nouns. They are listed on the right side in Figure 10.3. We can see that these noun words indeed represent the semantic meaning of the clusters. For example, the words *graphic, color, image*, and *point* are good descriptions of the cluster *Computer Graphics*. Comparing the word distribution on the left, these words are identifiable from their large weight values. This shows that the weights, together with the word function, are useful in selecting the key words for representing the meanings of clusters.

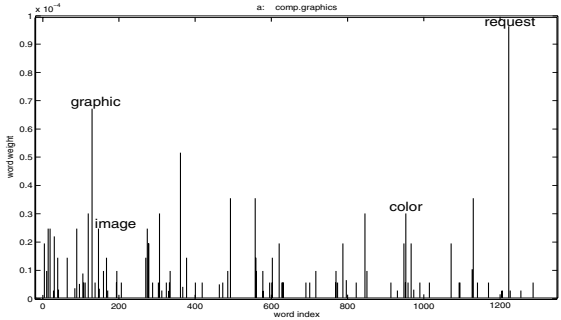
We can also observe that some words have large weights in more than one cluster. For example, the word *request* has large weight values in two classes, *Computer Graphics* and *Microsoft Windows*. Such words indicate that the two classes are semantically close.

10.7 Related Work

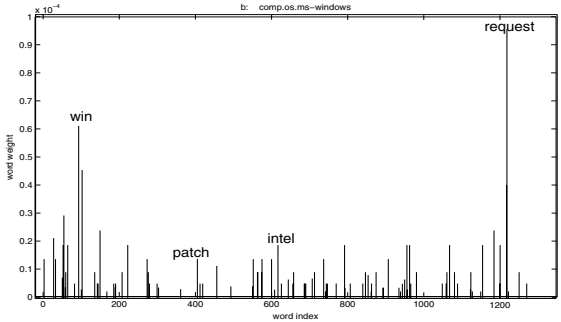
Feature selection has been an important research topic in cluster analysis [5, 6, 7, 9, 10, 11, 12, 17, 18, 19, 20].

Desarbo et al. [7] introduced the first method for variable weighting in k-means clustering in the SYNCLUS algorithm. The SYNCLUS process is divided into two stages. Starting from an initial set of weights, SYNCLUS first uses the k-means clustering to partition the data into k clusters. It then estimates a new set of optimal weights by optimizing a weighted mean-square, stress-like cost function. The two stages iterate until they converge to an optimal set of weights. The algorithm is time consuming computationally [12], so it cannot process large datasets.

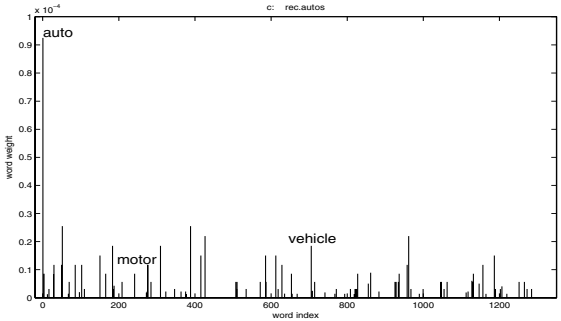
De Soete [5, 6] proposed a method to find optimal variable weights for ultrametric and additive tree fitting. This method was used in hierarchical clustering methods to solve variable weighting problems. Since the hierar-



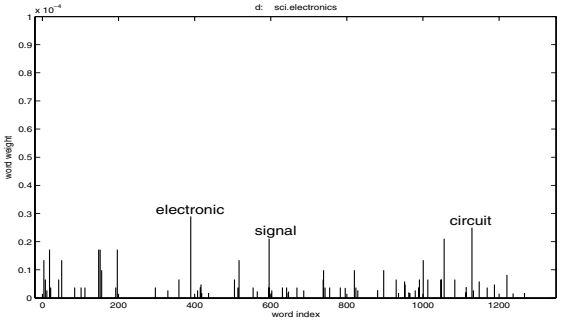
graphic	$6.70466e-05$
color	$2.98961e-05$
image	$2.45266e-05$
icon	$1.42196e-05$
laser	$9.52604e-06$
scope	$9.52604e-06$
point	$5.41076e-06$
sheet	$4.94495e-06$
plain	$3.21929e-06$
gui	$2.20811e-06$



win	$6.13444e-05$
intel	$2.14806e-05$
patch	$1.90001e-05$
logic	$1.15958e-05$
pc	$9.37718e-06$
buffer	$9.37718e-06$
demo	$8.34777e-06$
function	$5.32089e-06$
company	$5.32089e-06$
database	$3.91727e-06$



auto	$9.25063e-05$
vehicle	$1.8565e-05$
motor	$1.18095e-05$
driver	$9.01719e-06$
park	$8.57334e-06$
repair	$5.74717e-06$
mile	$4.15965e-06$
door	$3.23471e-06$
show	$3.21888e-06$
manufacture	$1.94154e-06$



electronic	$2.89103e-05$
circuit	$2.49422e-05$
signal	$2.10053e-05$
chip	$1.33768e-05$
volume	$9.80421e-06$
thread	$6.51865e-06$
charge	$3.67175e-06$
raster	$2.6509e-06$
science	$2.2915e-06$
technology	$1.91447e-06$

FIGURE 10.3: The noun words with large weights extracted from each cluster of the dataset B4. We can see that these words indeed represent the semantic meaning of the corresponding clusters.

chical clustering methods are computationally complex, De Soete's method cannot handle large datasets. Makarenkov and Legendre [18] extended De Soete's method to optimal variable weighting for k-means clustering. The basic idea is to assign each variable a weight w_i in calculating the distance between two objects and find the optimal weights by optimizing the cost function $L_p(w_1, w_2, ..., w_p) = \sum_{k=1}^{K} (\sum_{i,j=1}^{n_k} d_{ij}^2/n_k)$. Here, K is the number of clusters, n_k is the number of objects in the kth cluster, and d_{ij} is the distance between the ith and the jth objects. The Polak-Ribiere optimization procedure is used in minimization, which makes the algorithm very slow. The simulation results in [18] show that the method is effective in identifying important variables but not scalable to large datasets.

Modha and Spangler [20] very recently published a new method for variable weighting in k-means clustering. This method aims to optimize variable weights in order to obtain the best clustering by minimizing the ratio of the average within-cluster distortion over the average between-cluster distortion, referred to as the generalized Fisher ratio Q. To find the minimal Q, a set of feasible weight groups was defined. For each weight group, the k-means algorithm was used to generate a data partition and Q was calculated from the partition. The final clustering was determined as the partition having the minimal Q. This method of finding optimal weights from a predefined set of variable weights may not guarantee that the predefined set of weights would contain the optimal weights. Besides, it is also a practical problem to determine the predefined set of weights for high-dimensional data.

Friedman and Meulman [10] recently published a method to cluster objects on subsets of attributes. Instead of assigning a weight to each variable for the entire dataset, their approach is to compute a weight for each variable in each cluster. As such, $p * L$ weights are computed in the optimization process, where p is the total number of variables and L is the number of clusters. Since the objective function is a complicated, highly non-convex function, a direct method to minimize it has not been found. An approximation method is used to find clusters on different subsets of variables by combining conventional distance-based clustering methods with a particular distance measure. Friedman and Meulman's work is related to the problem of subspace clustering [3]. Scalability is a concern because their approximation method is based on the hierarchical clustering methods.

Projected clustering is another method for feature selection of high-dimensional data. *PROCLUS* is the first algorithm [1]. It starts with a set of initial cluster centers discovered from a small data sample. The initial centers are made as far apart from each other as possible. For each center, a set of data points within a distance δ to the center is identified as the center *locality* L_i. Here, δ is the minimal distance between the center and other centers. For each L_i, the average distance between the points in L_i and the center is computed in each dimension. The subset of dimensions whose average

distances are smaller than the average distance of all dimensions is considered as the candidate subspace for cluster i. After all candidate subspaces are identified, the clusters are discovered from the subspaces using the distance measures on subsets of dimensions. A few extensions have been made recently [2, 15, 22].

10.8 Discussions

k-means clustering is an important technique in data mining and many other real-world applications. In current practice, when using k-means, feature selection is either done manually using business domain knowledge or carried out in separate steps using statistical methods or data exploration. This is time consuming and difficult to make a right selection. Automated feature selection by feature weighting within the clustering process provides an easy solution. When handling very large data, a sample can be first clustered and features with large weights selected as the dimensions for clustering the whole dataset. Since the k-means clustering process is not changed much, this k-means feature weighting algorithm is efficient in clustering large data. Comparatively, other feature weighting methods for clustering as mentioned in the previous section are not scalable to large data.

Subspace clusters in high-dimensional data is a common phenomenon in many real-world applications, such as text mining, bio-informatics, e-business, supply chain management, and production scheduling/planning in manufacturing. In this chapter, we have demonstrated that the featuring weighting method in k-means can be extended to subspace clustering and the experimental results on text data are satisfactory. However, some further research problems remain. One is how to specify parameters β and σ when using this algorithm. To understand this, a sensitivity study needs to be conducted. The other one is a well-known problem: how to specify k, the number of clusters. To investigate this problem, a subspace cluster validation method needs to be developed. In the next step, we will work on solutions to these problems.

Acknowledgment

Michael Ng and Yunming Ye's work was supported by the National Natural Science Foundation of China (NSFC) under grant No.60603066.

Notes

1 http://kdd.ics.uci.edu/databases/20newsgroups/20newsgroups.html.

2 http://www.cs.cmu.edu/mccallum/bow.

References

[1] C. Aggarwal, C. Procopiuc, J. Wolf, P. Yu, and J. Park. Fast algorithms for projected clustering. In *Proc. of ACM SIGMOD*, pages 61–72, 1999.

[2] C. Aggarwal and P. Yu. Finding generalized projected clusters in high dimensional spaces. In *Proc. of ACM SIGMOD*, pages 70–81, 2000.

[3] R. Agrawal, J. Gehrke, D. Gunopulos, and P. Raghavan. Automatic subspace clustering of high dimensional data for data mining applications. In *Proc. of ACM SIGMOD*, pages 94–105, 1998.

[4] Y. Chan, W. K. Ching, M. K. Ng, and J. Z. Huang. An optimization algorithm for clustering using weighted dissimilarity measures. *Pattern Recognition*, 37(5):943–952, 2004.

[5] G. De Soete. Optimal variable weighting for ultrametric and additive tree clustering. *Quality and Quantity*, 20(3):169–180, 1986.

[6] G. De Soete. OVWTRE: A program for optimal variable weighting for ultrametric and addtive tree fitting. *Journal of Classification*, 5(1):101–104, 1988.

[7] W. S. Desarbo, J. D. Carroll, L. A. Clark, and P. E. Green. Synthesized clustering: A method for amalgamating clustering bases with differential weighting variables. *Psychometrika*, 49(1):57–78, 1984.

[8] C. Domeniconi, D. Papadopoulos, D. Gunopulos, and S. Ma. Subspace clustering of high dimensional data. In *Proc. of SIAM International Conference on Data Mining*, 2004.

[9] E. Fowlkes, R. Gnanadesikan, and J. Kettenring. Variable selection in clustering. *Journal of Classification*, 5(2):205–228, 1988.

[10] J. H. Friedman and J. J. Meulman. Clustering objects on subsets of attributes with discussion. *Journal of the Royal Statistical Society: Series B*, 66(4):815–849, 2004.

[11] R. Gnanadesikan, J. Kettenring, and S. Tsao. Weighting and selection of variables for cluster analysis. *Journal of Classification*, 12(1):113–136, 1995.

[12] P. E. Green, J. Carmone, and J. Kim. A preliminary study of optimal variable weighting in k-means clustering. *Journal of Classification*, 7(2):271–285, 1990.

[13] Z. Huang. Extensions to the k-means algorithms for clustering large data sets with categorical values. *Data Ming and Knowledge Discovery*, 2(3):283–304, 1998.

[14] Z. Huang, M. K. Ng, H. Rong, and Z. Li. Automated variable weighting in k-means type clustering. *IEEE Transactions on Pattern Analayis and Machine Intelligence*, 27(5):657–668, 2005.

[15] M. L. Liu. Iterative projected clustering by subspace mining. *IEEE Transactions on Knowledge and Data Engineering*, 17(2):176–189, 2005.

[16] J. MacQueen. Some methods for classification and analysis of multivariate observation. In *Proc. of the 5th Berkeley Symposium on Mathematical Statistica and Probability*, pages 281–297, 1967.

[17] V. Makarenkov and P. Leclerc. An algorithm for the fitting of a tree metric according to a weighted least-squares criterion. *Journal of Classification*, 16(1):3–26, 1999.

[18] V. Makarenkov and P. Leclerc. Optimal variable weighting for ultrametric and additive trees and k-means partitioning: methods and software. *Journal of Classification*, 18(2):245–271, 2001.

[19] G. Milligan. A validation study of a variable weighting algorithm for cluster analysis. *Journal of Classification*, 6(1):53–71, 1989.

[20] D. S. Modha and W. S. Spangler. Feature weighting in k-means clustering. *Machine Learning*, 52(3):217–237, 2003.

[21] S. Selim and M. Ismail. K-means-type algorithms: a generalized convergence theorem and characterization of local optimality. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6(1):81–87, 1984.

[22] J. Yang, W. Wang, H. Wang, and P. Yu. δ-clusters: capturing subspace correlation in a large data set. In *Proc. of ICDE*, pages 517–528, 2002.

[23] K. Y. Yip, D. W. Cheung, and M. K. Ng. A practical projected clustering algorithm. *IEEE Transactions on knowledge and data engineering*, 16(11):1387–1397, 2004.