0-1 SEMIDEFINITE PROGRAMMING FOR CLUSTER ANALYSIS

0-1 Semidefinite Programming for Cluster Analysis with Application to Customer Segmentation

By

Huarong Chen, B.Sc.

A Thesis

Submitted to the School of Graduate Studies

in Partial Fulfillment of the Requirements

for the Degree

Master of Science

McMaster University

MASTER OF SCIENCE(2006) McMaster University

COMPUTING AND SOFTWARE Hamilton, Ontario

TITLE: 0-1 Semidefinite Programming for Cluster Analysis with Application to Customer Segmentation

AUTHOR: Huarong Chen, B.Sc.

SUPERVISOR: Dr. Jiming Peng

NUMBER OF PAGERS: xv, 127

Abstract

In general, clustering involves partitioning a give data set into subsets based on the closeness or similarity among the data. Clustering analysis has been widely used in many applications arising from different disciplines, including market analysis, image segmentation, pattern recognition and web mining.

Recently, a new optimization model, the so called 0-1 semidefinite programming(SDP) has been introduced by Peng and Xia in [2]. It has been proved that several scenarios of clustering, such as classical K-means clustering, normalized-cut clustering, balanced clustering and semi-supervised clustering can be embedded into the 0-1 SDP model.

In this thesis, we try to extend the 0-1 SDP model to the scenario of weighted K-means clustering, where the instances in the data set are associated with some weights indicating the importance of the instance. We also develop a hierarchical approach to attack the unified 0-1 SDP model, in which each binary separation is achieved by the refined weighted K-means method in one dimensional space. Moreover, we apply the approach developed in this thesis to a particular industrial application, where the task is to extract a model to predict the children information of customers based on their buying behaviors. During the process of the model building, clustering analysis was applied as the first step to group customers with similar children information, and then the link between the segmentation of customers and their shopping behaviors was discovered.

Numerical results based on our approach are reported in the thesis as well.

Acknowledgements

I would first like to express my sincere thanks and deep appreciation to my supervisor, Dr. Jiming Peng, for his constant support, thoughtful guidance throughout my research. I also thank him for his very careful reading and insightful suggestions during the writing of this thesis.

I am grateful to Dr. Jiming Peng, Dr. Sanzheng Qiao, and Dr. Kamran Sartipi for their agreement to be a committee member and for their careful review of this thesis and for their valuable comments.

I also like to express my gratitude to my friends, Jiaping Zhu for her helpful discussion on this work, and Wei Xu for his help with lanPMO software.

Finally, I want to thank my parents and my wife. Without their love, understanding, and encouragement, this work would not have been done.

Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

This chapter will first gives an introduction to clustering, and then discuss the existing clustering methods, including partitioning and hierarchical clustering. Finally, the organization of the thesis is presented.

1.1 Introduction to Clustering

Clustering is the process to group objects that seem to fall naturally together based on some similarity/dissimilarity measurements, such that objects within the group are similar to each other and objects that belong to different groups are dissimilar. There are many kinds of clustering problems and algorithms, resulting from various choices of the measurements among entities in a data set. For a comprehensive introduction to the clustering, we refer to the book[11;

35], and for more recent results, see survey papers [12] and [37]. In this section, we just give a brief introduction to clustering.

To illustrate the concept of clustering, we take a point data set from [8] as example. The data set consists of 130 points showed in Figure1.1(a), it can be mainly partitioned into two groups. The left group has 90 points points displayed in circular rings, which can be further divided into outer and inner rings. The right group contains other 40 points that also can be further partitioned into upper and lower parts. Figure1.1(b) shows the desired clustering result of points. In the later chapter of Numerical Experiments, we will give the detail process of clustering.

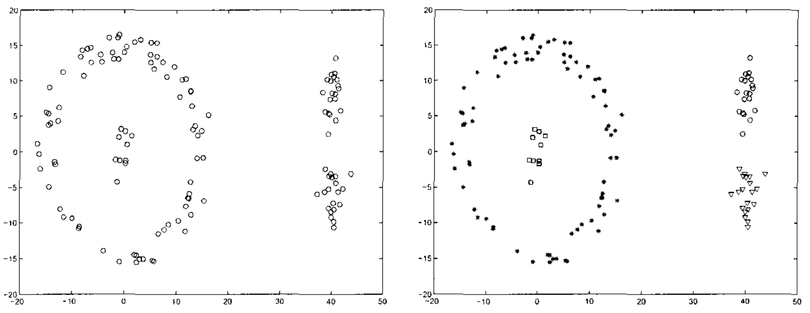


Figure 1.1: Data Points Clustering

Unlike classification, clustering is also known as unsupervised learning since the class label of each instance is unknown, and the number of clusters is need to be learned during the analysis. After grouping the given data, labels are assigned to instances. It means that the class label is data driven, solely learned from data. In order to measure the result of clustering, validation of clusters is needed. It is frequently approached by three ways. The first way is to visualize the clusters. For example, visualization is commonly used to demonstrate the effect of image segmentations. The second way is to measure the cohesion within a cluster. In other words, it is to test how objects are correlated to others within a cluster. A cluster that has a high value of of cohesion is considered better than a cluster that has a lower value. The last way is to check the separation of clusters. The difference between clusters should be significant, such that clusters are well-separated and represented in different manner.

Currently clustering analysis has being widely studied in a number of research communities, including machine learning, statistics, social science, optimization and computational geometry. Here we list some typical examples.

Market Analysis In business, clustering can help marketers discover distinct customer groups based on purchasing patterns, such that the profitable market share could be achieved effectively. For example, to determine the good location of a new business store, to predict the customer shopping taste, and etc[16].

Image Segmentation Many similarity measurements are used to partition an input image into subgraphs, each of which is considered to be homogeneous with respect to some image property of interests, such as the spatial location of each pair of pixels, the feature information based on the intensity, color or texture information[8].

Web Mining Web Mining has been popular in document analysis, search engine and knowledge discovery in large volume of web documents, it may consist of text, images, audio, video, or structured records such as lists and tables[38].

Character Recognition To identify lexemes in hand written text for the purpose of writer-independent handwriting recognition. The success of handwriting recognition is vitally dependent on its acceptance by users[12].

Biology Clustering Mainly used to build groups of genes with related expression patterns or group homologous sequences into gene families[20].

1.2 Clustering Method

In general, clustering methods can be mainly categorized into partitioning method and hierarchical method based on the similarity/dissimilarity measurement adopted in the clustering analysis. In this section, we give a brief review of two main clustering approaches.

1.2.1 Partitioning Method

The partitioning method performs a disjoint cluster analysis, the observations are partitioned into clusters such that every observation belongs to one and only one cluster. The clusters do not form a tree structure as they do in the hierarchical method. In practice, the Euclidean distance is mostly adopted for similarity measurement of observations, cluster centers are based on least-square estimation.

K-means Algorithm

Among the various partitioning methods, the classic K-means shown in Algorithm (1) is by far the most popular clustering method used in scientific and industrial areas. Many variations of k-type clustering algorithms are extended from it, including K-medoids, K-modes and weighted K-means. The essential of K-means Algorithm is to use a criteria that minimizes the sum-of-squared

Euclidean distance from each object to its assigned cluster center, so called MSSC. More precisely, for a given data set S of n points in $d-$dimensional space denoted by

$$S = \{v_i : \quad v_i \in \Re^d \;\; i = 1, \cdots, n\},$$

the task of MSSC is to find an assignment of the n points into k disjoint clusters $\mathcal{C} = (C_1, \cdots, C_k)$ centered at cluster centers $\mathbf{c}_j$ $(j = 1, \cdots, k)$ based on the total sum-of-squared Euclidean distances from each point v_i to its assigned cluster centroid c_j, i.e.,

$$\min \quad f(S, C) = \sum_{j=1}^{k} \sum_{v_i \in C_j} \|v_i - c_j\|^2 \tag{1.1}$$

Algorithm 1 K-means Clustering Algorithm

```
Step 1: Choose $k$ cluster centers randomly generated in a domain containing all the points,
Step 2: Assign each point to the closest cluster center,
Step 3: Recompute the cluster centers using the current cluster memberships,
Step 4: If a convergence criterion is met, stop; Otherwise go to step 2.
```

K-means algorithm is efficient to handle the large data set since the complexity is only $O(tkn)$, where t is the number of times we run the algorithm, k is the number of clusters and n is the number of instances. It gradually improve the cluster centers and reduce the sum of the square error (1.1) at each iteration until the error can not be reduced anymore. Although K-means clustering is widely applied in practice, it also has some drawbacks listed as follows.

1. K-means is sensitive to initial choice of cluster centers, the clustering can be very different by starting from different centers.

2. K-means tends to converge to a local optimum, in most cases K-means can not find the global minimum of the measurement used in the model.

3. There is no approximation bound available for K-means and its variants.

4. K-means only works for numeric attributes. For categorical attributes, the transformation to numerical attributes is required

5. K-means is sensitive to the outliers since outliers could significantly change the mean of clusters.

Based on the analysis of K-means, some algorithms were developed to overcome its shortcomings. For example, K-modes algorithm is extended to handle categorical attributes, K-medoids algorithm attacks the problem of outliers. In this thesis, the new framework for clustering analysis based on 0-1 semidefinite programming addresses the first three issues. Moreover, weighted K-means algorithm is introduced to take into account the importance of observations.

Weighted K-means Algorithm

Weighted K-means algorithm is an extension of the classic K-means algorithm. In K-means clustering, each observation has the same weight. In other words, K-means algorithm treats each observation equally. However, this is not a fair way to measure the importance of observation sometimes. For example, If we like to group customers based on some similarities, the number of families in a postal region signifies the distribution of the population. In such case, the importance of each instance should be considered. Otherwise, any partitions of customers are not realistic.

Similar to the classic K-means, the MSSC of weighted K-mean is calculated by

$$\min \quad f(S,C) = \sum_{j=1}^{k} \sum_{v_i \in C_j} d_i \|v_i - c_j\|^2, \tag{1.2}$$

where d_i denotes the weight of an instance v_i. In addition, the mean of all instances in a cluster is computed by taking account of weights, denoted by

$$c_j = \frac{\sum_{v_i \in C_j} d_i v_i}{\sum_{v_i \in C_j} d_i} \tag{1.3}$$

1.2.2 Hierarchical Method

Hierarchical clustering builds a a tree of clusters, also known as a *dendrogram*. Every cluster node contains child clusters. In other words, sibling clusters are covered by their common parent. For example, Figure 1.2 shows a tree structure of seven clusters labeled A, B, C, D, E, F and G, there are three clusters formed at the level with the dot line.

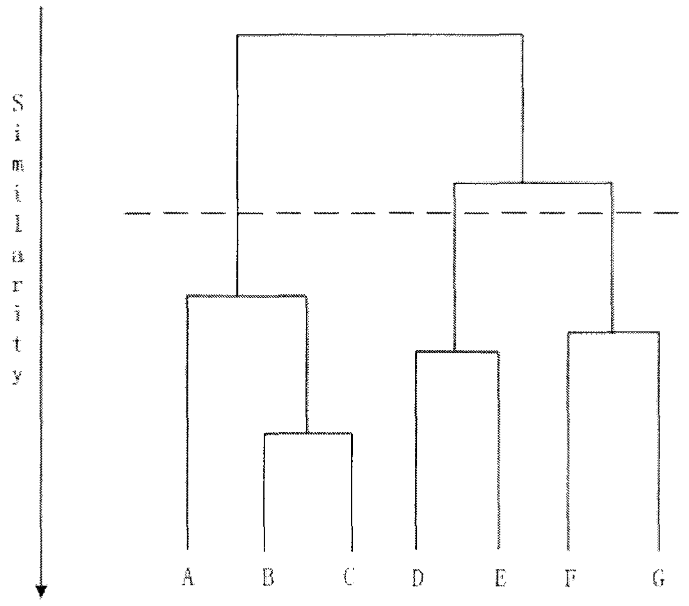


Figure 1.2: The dendrogram of clusters

In terms of the way to construct the *dendrogram* hierarchical clustering is categorized into agglomerative method and divisive method. For the agglomerative approach, every observation is treated as single clusters at the beginning and recursively merges two most appropriate clusters. The divisive method works in the reverse way. The whole data set is first perceived as one cluster, and then it is separated into two clusters, the repeating process continues until some stopping criteria are meet.

There are three commonly used methods to measure the similarity of two clusters during the merging process, the centroid method, average linkage method, and the ward's minimum-variance method. Each method has its own features and weak points. The following part gives a brief discussion of three methods.

Centroid Method

In centroid method, the distance between two clusters C_k and C_l is defined as the squared Euclidean distance between their centroids or means, shown as

$$D_{kl} = \|c_k - c_l\|^2$$

where c_k and c_l denote the center of cluster C_k and C_l respectively. The centroid method is tend to detect clusters with the spherical shape, it is less robust to outliers than other two methods.

Average linkage

In average linkage method, the distance between two clusters is defined as the average distance between each pair of observations, calculated by

$$D_{kl} = \frac{1}{N_k N_l} \sum_{i \in C_k} \sum_{j \in C_l} \|x_i - x_j\|^2$$

where N_k and N_l denote the number of observations in cluster C_k and C_l. The average linkage tends to join clusters with small variance, so that it is slightly biased toward producing clusters with the same variance.

Ward's Minimum-Variance

In ward's minimum-variance method, the distance between two clusters is computed by the ANOVA sum of squared Euclidean distance between the two clusters, i.e.,

$$D_{kl} = \frac{\|c_k - c_l\|^2}{\frac{1}{N_k} + \frac{1}{N_l}}.$$

It is easy to see that ward minimum method tends to merge clusters with a small number of observations, and it is strongly biased on producing clusters with the roughly same number of observations.

1.2.3 Other clustering methods

Besides partitioning and hierarchical clustering methods, there are also some other clustering methods proposed from different disciplines. These methods depict the ideas of different subjects and explore the natural multidisciplinary of clustering analysis. For a more comprehensive introduction, we refer to [12; 23]. In the following part, we just list some typical methods and their basic ideas.

Density Based Clustering: In the process of the grouping, density based method absorbs the density-connected objects into clusters with respect to the density-reachability. It is able to find arbitrary shape of clusters since the clustering detection is based on the density objects[16].

Graph-based Clustering: A number of clustering techniques that took a graph-based view of data are used. Data objects are represented by nodes and the proximity between two data objects is represented by the weight of the edge between the corresponding nodes[19].

Fuzzy Clustering: Unlike disjointed clustering methods, in which each instance belongs to one and only one cluster. Fuzzy clustering tries to discover the probability or grade of membership of each object in clusters by using membership function[24].

K-Nearest Neighbor(KNN)Clustering: In this approach, each unlabeled pattern is assigned to the cluster of its k nearest labeled neighbors as long as the average distance to the k neighbors is below a threshold[21].

1.3 Organization of this thesis

This thesis is organized as follows. In chapter 2, we transfer the weighted K-means clustering into a novel optimization model via semidefinite programming, in which the eigenvalues of involved matrix argument must be 0 or 1, it is so called 0-1 SDP model. We also show how the problem of k-ways normalized cut can be embedded into 0-1 SDP model. In chapter 3, we consider the approximate algorithms for solving the 0-1 SDP based on the relaxation. A new approximation method that extracts a feasible solution via PCA(Principal Component Analysis) of the projection of the affinity matrix is proposed. It has been shown that the algorithm can provide 2-approximation to the global solution of the original problem. In addition, we present a divisive hierarchical algorithm for clustering when $k \geq 3$, where each binary separation is achieved by the refined weighted K-means method in one dimensional space. In chapter 4, preliminary experiments are reported, the text problems include image segmentations and the partition of regular data sets. In chapter 5, we discuss a project sponsored by *MITACS* and *Rogers Communication Inc.*. The purpose of the project is to extract a pattern to predict the children information of customers based on their buying behaviors. Clustering analysis was applied as the first step to group customers in the perspective of children information, and then the interrelationship between the segmentation of customers and their shopping behaviors was discovered. In the last chapter, we summarize our contribution in this thesis, includes some concluding remarks.

Chapter 2

0-1 SDP Model for clustering

In this chapter, we first give a brief introduction to 0-1 semidefinite programming(SDP), and then we show how the classical K-means clustering and the k-ways normalized cut problem can be embedded into 0-1 SDP model, finally we establish the equivalence between weighted K-means clustering and the 0-1 SDP model.

2.1 0-1 Semidefinite Programming

In general, SDP refers to the problem of optimizing a linear function over the convex cone of symmetric and positive semidefinite matrices, subjected to linear equality constraints. The canonical SDP takes the following form

$$\textbf{(SDP)}\begin{cases} \min & \mathrm{Tr(WZ)} \\ S.T. & \mathrm{Tr(B_iZ) = b_i \quad for \quad i = 1, \cdots, m} \\ & Z \succeq 0 \end{cases}$$

Here Tr(.) denotes the trace of the matrix, and $Z \succeq 0$ means that Z is positive semidefinite. If we further require that the eigenvalues of the matrix argument in the above model take values 0 or 1, which implies Z is a projection matrix satisfying the relation $Z^2 = Z$, then we end up with the following problem

$$\textbf{(0-1 SDP)}\begin{cases} \min & \mathrm{Tr(WZ)} \\ S.T. & \mathrm{Tr(B_iZ) = b_i \quad for \quad i = 1, \cdots, m} \\ & Z^2 = Z, Z = Z^T \end{cases}$$

We call it 0-1 SDP owing to the similarity of the constraint $Z^2 = Z$ to the classical 0-1 requirement in integer programming. It is easy to see that the SDP model is a relaxation of the 0-1 SDP model.

2.2 0-1 SDP Model for K-means Clustering

The equivalence between the MSSC in classic K-means clustering and 0-1 SDP was first established in [2] and further discussed in [6]. However, for self-completeness, we briefly describe how K-means clustering can be embedded into the 0-1 SDP model.

Let $X = [x_{ij}] \in \Re^{n\times k}$ be the assignment matrix defined by

$$x_{ij} = \begin{cases} 1 & \text{If } v_i \text{ is assigned to cluster } C_j; \\ 0 & \text{Otherwise.} \end{cases}$$

in the feasible set $\mathcal{F}_k$ defined by

$$\mathcal{F}_k = \{X : Xe^k = e^n, x_{ij} \in \{0, 1\}\}.$$

where e is a all 1 vector in the suitable space.

By rearranging some items, the objective function (1.1) can be rewritten as

$$f(S, C) \quad = \sum_{i=1}^{n} \|v_i\|^2 \left(\sum_{j=1}^{k} x_{ij}\right) - \sum_{j=1}^{k} \frac{\|\sum_{i=1}^{n} x_{ij}v_i\|^2}{\sum_{i=1}^{n} x_{ij}}. \tag{2.1}$$

If we define

$$Z = X(X^TX)^{-1}X^T,$$

then we have

$$Z^2 = Z, Z^T = Z, Z \geq 0, Ze = e, \mathrm{Tr(Z) = k}.$$

It is easy to see that Z is a projection matrix. Based on properties of the projection matrix, we can rewrite (2.1) as

$$\mathrm{Tr}(\mathrm{WW^T(I - Z)}) = \mathrm{Tr}(\mathrm{WW^T}) - \mathrm{Tr}(\mathrm{WW^TZ})$$

where $W \in \Re^{n\times d}$ denotes the data matrix whose i-th row represents an instance $v_i \in \Re^d$.

Therefore, we have the following 0-1 SDP model for MSSC of K-means clustering.

$$
\begin{aligned}
\min \quad & \mathrm{Tr}\big(\mathrm{WW^T}(\mathrm{I} - \mathrm{Z})\big) \\
& Ze = e, \mathrm{Tr(Z)} = \mathrm{k}, \\
& Z \geq 0, Z = Z^T, Z^2 = Z.
\end{aligned}
\tag{2.2}
$$

2.3 0-1 SDP Model for graph-cut clustering

In general, graph-cut clustering is refer to partition a given graph $G(V, E)$ into disjointed subgraphs by removing some edges, such that each of which is considered to be homogeneous with respect to some interesting image properties. More rigorously, given a set of vertices $V = \{v_i \in \Re^m, i = 1 \cdots n\}$, we first define the weight matrix $W = [w_{ij}]$ where $w_{ij} = \phi(v_i, v_j)$ for a kernel function $\phi(.)$, which can be further interpreted as the weight of an edge between two pixels v_i and v_j, we then solve the graph-cut optimization problem according to the coefficient matrix.

2.3.1 Weight Matrix

Since cutting lines of the graph-cut clustering are non-linear separable in the input space, we need to transfer the input data into so-called kernel space via a nonlinear mapping and then perform the graph-cut clustering in the transfered kernel space. In the following parts, We will discuss the two popular choices of kernel functions to construct a weighted matrix, Gaussian kernel and Image kernel introduced by Shi and Malik[8].

Gaussian Kernel Matrix

The Gaussian kernel of a weight matrix is used to measure the spatial information of any two points v_i and v_j, defined as

$$w(i,j) = \exp^{-\frac{d_{ij}}{\sigma}}, \sigma > 0 \tag{2.3}$$

where $d_{ij} = \|v_i - v_j\|^2$ is Euclidean distance, σ is similar to the term of standard deviation, as it roughly translated to the sensitivity range of neighboring points. Empirically let $\sigma = r.c$, where $c = \frac{1}{n}\sum_i \min_j d_{ij}$ is the mean nearest-neighboring distance, r is an integer parameter [13].

Weight Matrix for Images

In the image segmentation, the weight matrix should reflects not only the spatial location of any two pixels, but also the likelihood of some feature factors based on the intensity, color or texture information[8]. It is defined by

$$w(i,j) = \exp^{-\frac{\|\mathcal{F}_i - \mathcal{F}_j\|^2}{\sigma_f}} * \begin{cases} \exp^{-\frac{\|v_i - v_j\|^2}{\sigma_v}} & \text{if } \|v_i - v_j\|^2 < r \\ 0 & \text{otherwise} \end{cases} \tag{2.4}$$

where the integer parameter r is the threshold of spatial location of pixels, it means that the weight w_{ij} becomes zero when any pair of pixel v_i and v_j are more than r pixels apart. $\mathcal{F}_i$ is a feature vector at the pixel v_i defined as:

- $\mathcal{F}_i = 1$, in the case of segmenting points sets,
- $\mathcal{F}_i = I(i)$, the intensity value, for segmenting brightness images,
- $\mathcal{F}_i = [v, v.s.\sin(h), v.s.\cos(h)](i)$, where h, s, v are the HSV values for color segmentations,
- $\mathcal{F}_i = [|I * f_1|, \cdots, |I * f_n|](i)$, where f_i is the DOOG filters at various scales and orientations in the case of texture segmentation.

2.3.2 Normalized Cut

Recently the normalized k-cut has caught much attention in the machine learning community. It is first introduced by Shi and Malik [8] and later investigated by Xing and Jordan[13]. The normalized cut computes the cut cost as a fraction of the total edges connections within subgraphs. In other words, the dissimilarity between subgraphs is measured by the cost of cutting edges, the similarity of a subgraph is measured by the total connections of edges in the subgraph. Mathematically, the normalized cut for a given graph $G(V, E)$ partitioned into two disjointed subgraph $V = A \bigcup B$, is defined as

$$Ncut(A, B) = \frac{S(A, B)}{d_A} + \frac{S(A, B)}{d_B}, \tag{2.5}$$

where $S(A, B) = \sum_{i \in A} \sum_{j \in B} w_{ij}$ is the total weight of edges across the subgraph A and B, $d_A = \sum_{i \in A} d_i$ is the total weight of edges within the subgraph A, $d_i = \sum_{j=1}^{n} w_{ij}$ is the total weight of edges that connected to the node v_i. It is easy to see that minimizing the function 2.5 is equivalent to maximize the following

$$\frac{S(A, A)}{d_A} + \frac{S(B, B)}{d_B}.$$

Furthermore, minimization of 2.5 can be rewritten as

$$\min \quad \frac{\mathrm{h}_\mathrm{A}^\mathrm{T}(\mathrm{W} - \mathrm{D})\mathrm{h}_\mathrm{A}}{\mathrm{h}_\mathrm{A}^\mathrm{T}\mathrm{D}\mathrm{h}_\mathrm{A}} + \frac{\mathrm{h}_\mathrm{B}^\mathrm{T}(\mathrm{W} - \mathrm{D})\mathrm{h}_\mathrm{B}}{\mathrm{h}_\mathrm{B}^\mathrm{T}\mathrm{D}\mathrm{h}_\mathrm{B}} \tag{2.6}$$

where D is diagonal matrix denoted by diag$(d_1, \cdots, d_n)$, h_A is n x 1 vector, $h(i) = 1$ if the node v_i belongs to A; otherwise, $h(i) = 0$.

2.3.3 0-1 SDP Model for Normalized Cut

Many interesting results about normalized cut have been reported [8; 13–15; 17; 18; 25]. In particular, Dhilllon at [14] showed that the normalized

k-cut problem is equivalent to the weighted K-means clustering. Xing and Jordan at [13] considered an SDP relaxation on the normalized k-cut problem. We next show that the normalized k-cut can be embedded into the 0-1 SDP model.

Let us first recall the model for the normalized k-cut problem[13] and the assignment matrix X defined before, and let $d = We^n$ and $D = \text{diag}(d)$ where W denotes the affinity matrix computed by (2.3) or (2.4) according to the type of images. The exact model for the normalized k-cut problem in [8; 13] can be rewritten as

$$\min_{X\in\mathcal{F}_k} \quad \mathrm{Tr}\left(\mathrm{D}^{-1}\mathrm{W} - (\mathrm{X}^{\mathrm{T}}\mathrm{D}\mathrm{X})^{-1}\mathrm{X}^{\mathrm{T}}\mathrm{W}\mathrm{X}\right). \tag{2.7}$$

If we define

$$Z = D^{\frac{1}{2}}X(X^TDX)^{-1}X^TD^{\frac{1}{2}},$$

it is easy to find that

$$Z^2 = Z, Z^T = Z, Z \geq 0, Zd^{\frac{1}{2}} = d^{\frac{1}{2}}, \mathrm{Tr(Z) = k}.$$

Therefore, we obtain the 0-1 SDP model for the normalized k-cut problem.

$$\begin{aligned} \min \quad & \mathrm{Tr}\left(\mathrm{D}^{-\frac{1}{2}}\mathrm{W}\mathrm{D}^{-\frac{1}{2}}(\mathrm{I}-\mathrm{Z})\right) \\ & Zd^{\frac{1}{2}} = d^{\frac{1}{2}}, \mathrm{Tr(Z) = k}, \\ & Z \geq 0, Z^2 = Z, Z = Z^T. \end{aligned} \tag{2.8}$$

2.4 0-1 SDP Model for Weighted K-means Clustering

Weighted K-means clustering is a variant of K-means clustering. Reformulation of weighted K-means clustering via semidefinite programming just requires some further efforts. However, this is never done before. In this section we show how to reformulate the MSSC of weighted K-means clustering (1.2) into 0-1 SDP model in detail.

Based on the assignment matrix X mentioned above, the weighted mean of all the points in a cluster C_j (1.3) is calculated by

$$c_j = \frac{\sum_{l=1}^{n} x_{lj} d_l v_l}{\sum_{l=1}^{n} x_{lj} d_l},$$

in which d_l denotes the weight for the point v_l.

Moreover, the MSSC for weighted K-means clustering (1.2) can be represented as

$$\min_{x_{ij}} \quad \sum_{j=1}^{k} \sum_{i=1}^{n} x_{ij} d_i \left\| v_i - \frac{\sum_{l=1}^{n} x_{lj} d_l v_l}{\sum_{l=1}^{n} x_{lj} d_l} \right\|^2 \tag{2.9}$$

$$\sum_{j=1}^{k} x_{ij} = 1 \ (i = 1, \cdots, n) \tag{2.10}$$

$$\sum_{i=1}^{n} x_{ij} \geq 1 \ (j = 1, \cdots, k) \tag{2.11}$$

$$x_{ij} \in \{0, 1\} \ (i = 1, \cdots, n;\ j = 1, \cdots, k) \tag{2.12}$$

The constraint (2.10) ensures that each point v_i is assigned to one and only one cluster, and (2.11) ensures that there are exactly k clusters. By rearranging some items, the objective function (2.9) can be rewritten as

$$f(S,C) \quad = \sum_{i=1}^{n} d_i \left\|v_i\right\|^2 \left(\sum_{j=1}^{k} x_{ij}\right) - \sum_{j=1}^{k} \frac{\left\|\sum_{i=1}^{n} x_{ij} d_i v_i\right\|^2}{\sum_{i=1}^{n} x_{ij} d_i}. \qquad (2.13)$$

Let $D = diag(d_1, \cdots, d_n)$ denote the diagonal weighted matrix. The first part of the objective function (2.13) can be rewritten by

$$\mathrm{Tr}(\mathrm{DWW^T}) = \mathrm{Tr}\left(\mathrm{D^{\frac{1}{2}}WW^TD^{\frac{1}{2}}}\right).$$

Since X is an assignmentmatrix, we have

$$X^T DX = diag(\alpha_p) \quad \text{where } \alpha_p = \textstyle\sum_{i=1}^{n} x_{ip} d_i \text{ and } p \in \{1, \cdots, k\} \ ,$$

Let

$$Z = D^{\frac{1}{2}} X (X^T DX)^{-1} X^T D^{\frac{1}{2}},$$

it follows that the any element of Z is

$$z_{ij} = \beta_{ij} \frac{d_i^{\frac{1}{2}} d_j^{\frac{1}{2}}}{\alpha_p}, \quad \text{where } i, j \in \{1, \cdots, n\}$$

where

$$\beta_{ij} = \begin{cases} 1 & \text{If } v_i \text{ and } v_j \text{ are in the same cluster } C_p, \\ 0 & \text{Otherwise;} \end{cases}$$

Let M denotes the result of the matrix multiplication of $D^{\frac{1}{2}}WW^TD^{\frac{1}{2}}$ and Z, whose elements can be represented by

$$M_{ij} = \sum_{l=1}^{n} d_i^{\frac{1}{2}} v_i v_l^T d_l^{\frac{1}{2}} \beta_{lj} \frac{d_l^{\frac{1}{2}} d_j^{\frac{1}{2}}}{\alpha_p} \quad \text{where } i, j \in \{1, \cdots, n\} \ ,$$

we have

$$\begin{aligned} \text{Tr(M)} &= \sum_{i=1}^{n}\sum_{l=1}^{n} d_i^{\frac{1}{2}} v_i v_l^T d_l^{\frac{1}{2}} \beta_{li} \frac{d_l^{\frac{1}{2}} d_i^{\frac{1}{2}}}{\alpha_p} \\ &= \frac{1}{\alpha_p} \sum_{i=1}^{n}\sum_{l=1}^{n} \beta_{li} d_i d_l v_i v_l^T \\ &= \sum_{j=1}^{k} \frac{\left\|\sum_{i=1}^{n} x_{ij} d_i v_i\right\|^2}{\sum_{i=1}^{n} x_{ij} d_i}, \end{aligned}$$

which is equivalent to the second part of the objective function (2.13). As a result, the objective function can be rewritten by

$$\text{Tr}\left(\text{D}^{\frac{1}{2}}\text{WW}^{\text{T}}\text{D}^{\frac{1}{2}}(\text{I} - \text{Z})\right).$$

From the definition of matrix Z, we have that Z is a projection matrix by satisfying $Z^2 = Z$ and $Z = Z^T$ with nonnegative elements, the eigenvalue of Z is either 1 or 0.

Since $Xe = e$, we have

$$X(X^TX)^{-1}X^Te = e.$$

This implies

$$Zd^{\frac{1}{2}} \;=\; D^{\frac{1}{2}}X(X^TDX)^{-1}X^TD^{\frac{1}{2}}d^{\frac{1}{2}}$$

$$
\begin{aligned}
&= D^{\frac{1}{2}}X(X^TDX)^{-1}X^TD^{\frac{1}{2}}D^{\frac{1}{2}}e \\
&= D^{\frac{1}{2}}X(X^TDX)^{-1}X^TDX(X^TX)^{-1}X^Te \\
&= D^{\frac{1}{2}}X(X^TX)^{-1}X^Te \\
&= d^{\frac{1}{2}},
\end{aligned}
$$

which means that $s = d^{\frac{1}{2}}$ is the eigenvector of Z corresponding to its largest eigenvalue 1. Moreover, the trace of Z should equal to the rank of assignment matrix X, the number of clusters, i.e.,

$$\mathrm{Tr(Z) = k.}$$

Therefore, we have the following 0-1 SDP model for weighted K-means clustering

$$
\begin{aligned}
\min \quad & \mathrm{Tr}\Big(\mathrm{D^{\frac{1}{2}}WW^TD^{\frac{1}{2}}(I - Z)}\Big) \qquad (2.14)\\
& Zd^{\frac{1}{2}} = d^{\frac{1}{2}}, \mathrm{Tr(Z) = k}, \\
& Z \geq 0, Z = Z^T, Z^2 = Z.
\end{aligned}
$$

Theorem 2.4.1. *Finding a global solution of problem (2.9) equals to solving the 0-1 SDP problem (2.14).*

Proof. From the construction of the 0-1 SDP model (2.14), we know that any feasible solution for (2.9) can be easily transfered into a feasible solution of problem (2.14). Therefore, it remains to show that from any feasible solution for (2.14), we can construct a feasible assignment matrix X for (2.9).

Suppose that Z is a feasible solution of (2.14). We can define a matrix $\bar{Z} = D^{-\frac{1}{2}}ZD^{\frac{1}{2}}$, it follows that

$$\bar{Z}^2 = \bar{Z}, \bar{Z}e = e, \bar{Z} \geq 0.$$

Let

$$\bar{Z}_{i_1j_1} = \max\{\bar{Z}_{ij} : 1 \leq i, j \leq n\} > 0,$$

and define the index set

$$\mathcal{J}_1 = \{i : \bar{Z}_{ij_1} > 0\}.$$

Since $\bar{Z}^2 = \bar{Z}$, we have

$$\bar{Z}_{i_1j_1} = \sum_{k=1}^{n} \bar{Z}_{i_1k}\bar{Z}_{kj_1} = \sum_{k\in\mathcal{J}_1} \bar{Z}_{i_1k}\bar{Z}_{kj_1}$$

which implies

$$\sum_{k\in\mathcal{J}_1} \frac{\bar{Z}_{i_1k}}{\bar{Z}_{i_1j_1}}\bar{Z}_{kj_1} = 1.$$

Since $\bar{Z}e = e$, we have

$$\sum_{k=1}^{n} \bar{Z}_{i_1k} = \sum_{k\in\mathcal{J}_1} \bar{Z}_{i_1k} = 1.$$

we can conclude that

$$\bar{Z}_{kj_1} = \bar{Z}_{i_1j_1}, \quad \forall k \in \mathcal{J}_1$$

$$\bar{Z}_{kj_1} = 0, \quad \forall k \notin \mathcal{J}_1.$$

This further implies

$$\bar{Z}_{i_1 j} = \sum_{k \notin \mathcal{J}_1} Z_{i_1 k} Z_{kj} = 0, \quad \forall j \notin \mathcal{J}_1.$$

Recall the definition of the matrix $\bar{Z}$, we can decompose the matrix $\bar{Z}$ into a bock matrix with the following structure

$$\bar{Z} = \begin{pmatrix} \bar{Z}_{\mathcal{J}_1 \mathcal{J}_1} & 0 \\ 0 & \bar{Z}_{\bar{\mathcal{J}}_1 \bar{\mathcal{J}}_1} \end{pmatrix}, \tag{2.15}$$

where $\bar{\mathcal{J}}_1 = \{i : i \notin \mathcal{J}_1\}$. Now we claim that $\bar{Z}_{\mathcal{J}_1 \mathcal{J}_1}$ is a submatrix with rank 1 for which all the elements in any column are equivalent. To see this, let us choose any column from the submatrix $\bar{Z}_{\mathcal{J}_1 \mathcal{J}_1}$ and consider the minimum element in that column, i.e. for a fixed $j \in \mathcal{J}_1$,

$$\bar{Z}_{i_1 j} = \min\{\bar{Z}_{ij} : i \in \mathcal{J}_1\},$$

Since $\bar{Z}^2 = \bar{Z}$ and $\bar{Z}e = e$, we have

$$\bar{Z}_{i_1 j} = \sum_{k \in \mathcal{J}_1} \bar{Z}_{i_1 k} \bar{Z}_{kj} \geq \bar{Z}_{i_1 j} \sum_{k \in \mathcal{J}_1} \bar{Z}_{i_1 k} = \bar{Z}_{i_1 j},$$

this means that the equality holds if and only if all the elements in the column are equivalent. Thus, we have $\mathrm{Tr}\big(\bar{\mathrm{Z}}_{\mathcal{J}_1 \mathcal{J}_1}\big) = \mathrm{rank}(\bar{\mathrm{Z}}_{\mathcal{J}_1 \mathcal{J}_1}) = 1$.

From the above discussion we can put all the points associated with the index set $\mathcal{J}_1$ into one cluster, and reduce the corresponding 0-1 SDP model(2.8) to a smaller problem as follows

$$\min \quad \mathrm{Tr}\Big(\mathrm{D}_{\bar{\mathcal{J}}_1}^{-\frac{1}{2}} \mathrm{W}_{\bar{\mathcal{J}}_1 \bar{\mathcal{J}}_1} \mathrm{D}_{\bar{\mathcal{J}}_1}^{-\frac{1}{2}} (\mathrm{I} - \mathrm{Z})_{\bar{\mathcal{J}}_1 \bar{\mathcal{J}}_1}\Big) \tag{2.16}$$

$$Z_{\bar{\mathcal{J}}_1\bar{\mathcal{J}}_1} d^{\frac{1}{2}}_{\bar{\mathcal{J}}_1} = d^{\frac{1}{2}}_{\bar{\mathcal{J}}_1}, \mathrm{Tr}(Z_{\bar{\mathcal{J}}_1\bar{\mathcal{J}}_1}) = \mathrm{k} - 1,$$

$$Z_{\bar{\mathcal{J}}_1\bar{\mathcal{J}}_1} \geq 0, Z^2_{\bar{\mathcal{J}}_1\bar{\mathcal{J}}_1} = Z_{\bar{\mathcal{J}}_1\bar{\mathcal{J}}_1}, Z_{\bar{\mathcal{J}}_1\bar{\mathcal{J}}_1} = Z^T_{\bar{\mathcal{J}}_1\bar{\mathcal{J}}_1}.$$

By repeating the above process, we can reconstruct all the clusters from a solution of problem (2.14). This establishes the equivalence between model (2.9) and (2.14)

It should be noted that the scalar vectors d introduced in (2.8) and (2.14) are different. In (2.8), d denotes the sum of each row of the coefficient matrix, whereas d presents the weight of each instance in (2.14). These results shows the power of our 0-1 SDP model, it is the underlying framework for numerous different-first-look clustering algorithms.

Chapter 3

Approximation Algorithms for 0-1 SDP

This chapter consists of four parts. In the first section, a general scheme of approximation for solving 0-1 SDP is addressed. In the second section, we propose a new approximation method based on the SVD of the coefficient matrix in the projection space. In the third section, the evaluation of the new approximation is elaborated. In the last section, we present a divisive hierarchical algorithm, where each binary separation is achieved by the refined weighted K-means method in one dimensional space based on the bi-clustering

3.1 Relaxation Algorithm for Solving 0-1 SDP

In this section, we discuss how to solve the 0-1 SDP model for clustering. For simplification of our discussion, we restrict ourselves to the following unified mode

$$\begin{aligned} \min \quad & \mathrm{Tr}(\mathrm{W}(\mathrm{I}-\mathrm{Z})) \\ & Zs = s, \mathrm{Tr}(\mathrm{Z}) = \mathrm{k}, \\ & Z \geq 0, Z = Z^T, Z^2 = Z, \end{aligned} \tag{3.1}$$

where s is a positive scalar vector satisfying $\|s\| = 1$. Throughout the paper, we further assume that the underlying matrix W is positive semidefinite. It is worthwhile mentioning that the affinity matrix defined by (2.3) or (2.4) is positive semidefinite, which guarantee the coefficient matrix derived from the the affinity matrix of the undertaking graph is positive semidefinite[8].

3.1.1 Approximation Algorithm Based on Relaxation

Since finding the global solution of problem (3.1) is NP-hard as proved in [8], we starts with a generic scheme of approximation algorithms for (3.1), which is depicted in Algorithm 2.

The relaxation step has an important role in the whole algorithm. For example, if the approximation solution obtained from Step 2 is feasible for

Algorithm 2 Approximation Algorithm Based on Relaxation

```
Step 1: Choose a relaxation model for (3.1),
Step 2: Solve the relaxed problem for an approximate solution,
Step 3: Use a rounding procedure to extract a feasible solution to (3.1)
from the approximate solution.
```

(3.1), then it is exactly an optimal solution of (3.1). On the other hand, when the approximation solution is not feasible regarding (3.1), we have to use a rounding procedure to extract a feasible solution.

3.1.2 0-1 SDP Relaxation

Various relaxations and rounding procedures have been proposed for solving (3.1) in the literature. For example, in [2], Peng ans Xia considered a relaxation of the classical K-means clustering based on linear programming and a rounding procedure was also proposed in the that work. Xing and Jordan [13] considered the SDP relaxation for normalized k-cuts and proposed a rounding procedure based on the singular value decomposition of the solution Z of the relaxed problem, i.e., $Z = U^TU$. In their approach, every row of U^T is cast as a point in the new space, and then the weighted K-means clustering is performed over the new set of those points in $\Re^k$. Similar works for spectral clustering can also be found in [15; 17; 18; 25; 28] where the singular value decomposition of the underlying matrix W is used and a K-means-type clustering based on the eigenvectors of W is performed.

The idea of using the singular value decomposition of the underlying matrix W is natural in the so-called principal component analysis (PCA) [26]. In [22], the link between PCA and K-means clustering was also explored and simple bounds were derived. In particular, Drineas et'al [32] proposed to use singular value decomposition to form a subspace, and then perform K-means clustering in the subspace $\Re^k$. They proved that the solution obtained by solving the K-means clustering in the reduced space can provide a 2-approximation to the solution of the original K-means clustering.

There are several different ways to relax the 0-1 SDP model (3.1). First of all, the argument Z is stipulated to be a projection matrix, i.e., $Z^2 = Z$, which implies that the matrix Z is a positive semidefinite matrix whose eigenvalues are either 0 or 1. A straightforward relaxation to (3.1) is replacing the requirement $Z^2 = Z$ by the relaxed condition

$$I \succeq Z \succeq 0.$$

Moreover, the nonnegativity of Z indiates that there exists exactly one nonnegative eigenvector corresponding to its largest eigenvalue(see Theorem 1.3.2 of [3]). Recall the constraint $Zs = s$, it follows immediately that s is the nonnegative eigenvector of Z corresponding to the largest eigenvale 1. In this circumstance, the constraint $Z \preceq I$ becomes superfluous and can be waived withoud any influence. Therefore, we need only to consider the following relaxed 0-1 SDP problem.

$$\begin{aligned} \min \quad & \mathrm{Tr(W(I-Z))} \\ & Zs = s, \mathrm{Tr(Z)} = \mathrm{k}, \\ & Z \geq 0, Z \succeq 0. \end{aligned} \tag{3.2}$$

The above problem is feasible and bounded below. We can apply many existing optimization solvers such as interior-point methods to solve (3.2). It is known that an approximate solution to (3.2) can be found in polynomial time. However, we should point out that although there exist theoretically polynomial algorithm for solving (3.2), most of the present optimization solvers are unable to handle the problem in large size efficiently.

Another interesting relaxation to (3.1) is to include $I - Z \succeq 0$ that could be implied by the constraint $Z^T = Z$ and $Zs = s$ as we pointed early in this section, and further removing the nonnegative requirement on the elements of Z, we obtain the following simple SDP problem

$$\begin{aligned} \min \quad & \mathrm{Tr(W(I-Z))} \\ & \mathrm{Tr(Z)} = \mathrm{k},\ \mathrm{I} \succeq \mathrm{Z} \succeq 0, \end{aligned} \tag{3.3}$$

3.2 A New Approximation Method

In this section, we propose a new approximation algorithm based on another relaxation form of the model (3.1). Let us recall that in the relaxed model (3.2), we stipulate that s is an eigenvector of the final solution matrix Z. Since we already know this fact in advance, we can keep such a simple constraint in our relaxed model, and only remove the nonnegative requirement in (3.2). Therefore, we obtain the following SDP relaxation

$$\begin{aligned} \min \quad & \mathrm{Tr(W(I-Z))} \\ & Zs = s, \mathrm{Tr(Z) = k}, \\ & (I - Z) \succeq 0. \end{aligned} \tag{3.4}$$

In the sequel we discuss how to solve the above problem. For any feasible solution of (3.4), let us define

$$\bar{Z} = (I - ss^T)Z. \tag{3.5}$$

It is easy to see that

$$\bar{Z} = (I - ss^T)Z = (I - ss^T)Z(I - ss^T),$$

i.e., $\bar{Z}$ represents the projection of the matrix Z onto the null subspace of s. Moreover, since $\|s\| = 1$, it is easy to verify that

$$\mathrm{Tr}(\bar{Z}) = \mathrm{Tr}(Z) - 1 = k - 1.$$

Let $\bar{W}$ denote the projection of the matrix W onto the null space of s, i.e.,

$$\bar{W} = (I - ss^T)W(I - ss^T). \tag{3.6}$$

Then, we can reduce (3.4) to

$$\begin{aligned} \min \quad & \mathrm{Tr}(\bar{W}(I - \bar{Z})) \qquad (3.7)\\ & \mathrm{Tr}(\bar{Z}) = k - 1, \\ & I \succeq \bar{Z} \succeq 0. \end{aligned}$$

which can be solved by singular value decomposition of the symmetric matrix W. Denote

$$W = Udiag(\lambda_1, \cdots, \lambda_n)U^T,$$

where λ_i are the eigenvalues of W in decreasing order, and U is an orthogonal matrix whose i-column is the eigenvector corresponding to to λ_i. The optimal solution of (3.7) can be achieved if and only if

$$\mathrm{Tr}(\bar{W}\bar{Z}) = \sum_{i=1}^{k-1} \lambda_i. \tag{3.8}$$

This gives us an easy way to solve (3.7) and correspondingly (3.4).

Theorem 3.2.1. *Let Z^* be the global optimal solution of (3.1), and $\lambda_1, \cdots, \lambda_{k-1}$ be the first largest eigenvalues of the matrix $\bar{W}$, then we have*

$$\mathrm{Tr(W(I - Z^*)) \geq Tr(W) - s^TWs} - \sum_{\mathrm{i=1}}^{\mathrm{k-1}} \lambda_{\mathrm{i}}.$$

Proof. Let us denote $\bar{Z}^*$ as an optimal solution for (3.7), from (3.8) we have

$$\bar{Z} = \sum_{i=1}^{k-1} u_i u_i^T.$$

Additionally, we know that (3.5) and $Zs = s$, which implies

$$Z = ss^T + \bar{Z}.$$

Therefore, we have

$$\begin{aligned}
\mathrm{Tr(W(I - Z^*)} \quad & = \mathrm{Tr(W) - Tr\big(W(ss^T + \bar{Z}^*)\big)} \\
& = \mathrm{Tr(W) - s^TWs - Tr\big(W\bar{Z}^*\big)} \\
& \geq \mathrm{Tr(W) - s^TWs - Tr\big(W\bar{Z}\big)} \\
& = \mathrm{Tr(W) - s^TWs} - \sum_{\mathrm{i=1}}^{\mathrm{n}} \sum_{\mathrm{j=1}}^{\mathrm{k-1}} \lambda_{\mathrm{i}} \mathrm{u_i u_i^T u_j u_j^T} \\
& = \mathrm{Tr(W) - s^TWs} - \sum_{\mathrm{i=1}}^{\mathrm{k-1}} \lambda_{\mathrm{i}}.
\end{aligned}$$

This finishes the proof of the theorem.

From our above discussion, the algorithmic scheme for solving (3.4) can be described in Algorithm 3.

Algorithm 3 SVD of the Project Coefficient Matrix

```
Step 1: Calculate the projection $\bar{W}$ via (3.6);
Step 2: Use singular value decomposition method to compute the first $k-1$ largest eigenvalues of the matrix $\bar{W}$ and their corresponding eigenvectors $u_1, \cdots, u_{k-1}$,
Step 3: Set
    $Z = ss^T + \sum_{i=1}^{k-1} u_i u_i{}^T.$
```

Note that solving the relaxed problem (3.7) can not provide a solution for the original problem (3.1). In the sequel we propose a rounding procedure to extract a feasible solution for (3.1) from a solution of the relaxed problem (3.7) provided by the SVD of the projection coefficient matrix. Let

$$U_{k-1} = \sum_{i=1}^{k-1} u_i u_i^T$$

be the solution obtained from the projected coefficient matrix, and

$$U_k = ss^T + U_{k-1}, \quad \bar{W}_{k-1} = \bar{W} U_{k-1}.$$

We can formulate a matrix in $\Re^{n \times (k-1)}$ whose i-th column is $\lambda_i^{\frac{1}{2}} u_i$. Then we cast each row in such a matrix as a point in $\Re^{k-1}$, and thus we obtain a data set of n points in $\Re^{k-1}$. After that we perform clustering task for the new data set. In other words, we need to solve the 0-1 SDP model (3.1) with a new coefficient matrix $\bar{W}_{k-1}$. Finally, we partition all the points in the original space based on the obtained clusters from the new data set. The whole algorithm can be describe as follows.

Algorithm 4 Approximation Algorithm based on the SVD of the Project Coefficient Matrix

```
**Step 1**: Calculate the projection of the matrix $W$ onto the null space of $s$, i.e.,
    $\bar{W} = (I - ss^T)W(I - ss^T);$
**Step 2**: Use singular value decomposition method to compute the first $k-1$ largest eigenvalues of the matrix $\bar{W}$ and their corresponding eigenvectors $u_1, \cdots, u_{k-1}$, and then compute $\bar{W}_{k-1}$;
**Step 3**: Solve problem (3.1) with the coefficient matrix $\bar{W}_{k-1}$ and assign all the points in the original space based based on the obtained assignment.
```

3.3 Estimation of Approximate Solution

We next progress to estimate the solution obtained from Algorithm 4.

Theorem 3.3.1. *Suppose that Z^* is a global solution to problem (3.1) and $\bar{Z}$ is the solution provided by Algorithm 4. Then, we have*

$$\mathrm{Tr}\big(\mathrm{W}(\mathrm{I} - \bar{\mathrm{Z}})\big) \leq 2\mathrm{Tr}(\mathrm{W}(\mathrm{I} - \mathrm{Z}^*)).$$

Proof. Let Z^* be a global solution to (3.1) and $\bar{Z}$ is the solution provided by Algorithm 4. From the choices of U_{k-1} and U_k it follows

$$\mathrm{Tr}\big((\mathrm{I} - \mathrm{U_k})\bar{\mathrm{W}}_{\mathrm{k-1}}\big) = 0; \tag{3.9}$$

$$\mathrm{Tr}\big(\mathrm{U_k}(\bar{\mathrm{W}} - \bar{\mathrm{W}}_{\mathrm{k-1}})\big) = 0. \tag{3.10}$$

From Theorem 3.2.1, we have

$$\mathrm{Tr}(\mathrm{W}(\mathrm{I} - \mathrm{Z}^*)) \geq \mathrm{Tr}(\mathrm{W}(\mathrm{I} - \mathrm{U})). \tag{3.11}$$

It follows

$$\mathrm{Tr}\big(\mathrm{W}(\mathrm{I} - \bar{\mathrm{Z}})\big) = \mathrm{Tr}\big(\mathrm{W}(\mathrm{I} - \mathrm{U} + \mathrm{U} - \bar{\mathrm{Z}})\big) \leq \mathrm{Tr}(\mathrm{W}(\mathrm{I} - \mathrm{Z}^*)) + \mathrm{Tr}\big(\mathrm{W}(\mathrm{U} - \bar{\mathrm{Z}})\big).$$

The above relation implies that if

$$\mathrm{Tr}\big(\mathrm{W}(\mathrm{U} - \bar{\mathrm{Z}})\big) \leq \mathrm{Tr}(\mathrm{W}(\mathrm{I} - \mathrm{Z}^*)), \tag{3.12}$$

then

$$\mathrm{Tr}\big(\mathrm{W}(\mathrm{I} - \bar{\mathrm{Z}})\big) \leq 2\mathrm{Tr}(\mathrm{W}(\mathrm{I} - \mathrm{Z}^*)),$$

i.e., in the worst case, the solution provided by Algorithm 4 is a 2-approximation to the original problem (3.1).

In what follows we prove (3.12), which can be equivalently stated as

$$\mathrm{Tr}\big(\mathrm{W}(\mathrm{I} - \mathrm{Z}^* + \bar{\mathrm{Z}} - \mathrm{U})\big) \geq 0. \tag{3.13}$$

By the choices of $Z^*, \bar{Z}$ and U, it is easy to verify

$$(I - Z^* + \bar{Z} - U)s = 0, \tag{3.14}$$

$$(I - ss^T)(I - Z^* + \bar{Z} - U)(I - ss^T) = (I - Z^* + \bar{Z} - U). \tag{3.15}$$

It follows immediately that

$$\begin{aligned}
\mathrm{Tr}\big(W(I - Z^* + \bar{Z} - U)\big) &= \mathrm{Tr}\big(\bar{W}(I - Z^* + \bar{Z} - U_k)\big) \\
&= \mathrm{Tr}\big(\bar{W}_{k-1}(I - Z^* + \bar{Z} - U_k)\big) \\
&\quad + \mathrm{Tr}\big((I - Z^* + \bar{Z} - U_k)\big)(\bar{W} - \bar{W}_{k-1}) \\
&= \mathrm{Tr}\big((\bar{Z} - Z^*)\bar{W}_{k-1}\big) + \mathrm{Tr}\big((I - Z^* + \bar{Z})(\bar{W} - \bar{W}_{k-1})\big) \\
&\geq \mathrm{Tr}\big(\bar{W}_{k-1}(\bar{Z} - Z^*)\big)
\end{aligned}$$

where the last equality is given by (3.9) and (3.10), and the last inequality is implied by the fact that $I - Z^* + \bar{Z} \succeq 0, \bar{W} - \bar{W}_{k-1} \succeq 0$ Recall that $\bar{Z}$ is the global solution of problem (3.7) with the coefficient matrix $\bar{W}_{k-1}$ while Z^* is only a feasible solution of (3.7), we therefore have

$$\mathrm{Tr}\big(\bar{W}_{k-1}(\bar{Z} - Z^*)\big) \geq 0,$$

which further implies (3.12). This finishes the proof of the theorem.

3.4 Divisive Hierarchical Approach

In this section, we will first focus on solving the problem (3.1) in the special case where the affinity matrix $W = vv^T$ with $v \in \Re^n$ and $k = 2$ by applying the refined weighted K-means method in one dimensional space, and then we propose a divisive hierarchical algorithm based on the binary separation.

3.4.1 Refined Weighted K-means in One Dimensional Space

Suppose that W is a positive semidefinite coefficient matrix in the 0-1 SDP model (3.1) with $W = VV^T, V \in \Re^{n\times d}$ and s is a positive scaling vector. If we cast all the rows ($v_i, i = 1, \cdots, n$) of the matrix V as vector in $\Re^d$, then solving the 0-1 SDP model(3.1) equals to solving the following weighted k-means clustering problem

$$\min_{c_1,\cdots,c_k} \sum_{i=1}^{n} \min\{s_i^2\|\frac{v_i}{s_i} - c_1\|^2, \cdots, \|\frac{v_i}{s_i} - c_k\|^2\}. \tag{3.16}$$

Note that for given candidate center $c_1, \cdots, c_k$, the weighted K-means assigns the scaled data points precisely in the manner as the classical K-means does. This allows us to use the so-called *Voronoi Partitions* in computation geometry [29] to find the global optimal solution of problem (3.1). Based on Theorem 3 and Theorem 5 in [29], the algorithm runs in $O(n^{dk+1})$ time.

In what follows we discuss a special case of (3.1) when $W = vv^T$ with $v \in \Re^n$ and k=2. In such the case, the (3.16) can be stated as

$$\min_{c_1,c_2} \sum_{v_i \in C_1} s_i^2 \| \frac{v_i}{s_i} - c_1 \|^2 + \sum_{v_i \in C_2} s_i^2 \| \frac{v_i}{s_i} - c_1 \|^2. \tag{3.17}$$

Since the weighted center of a cluster is calculated by

$$c_i = \frac{\sum_{v_i \in C_i} s_i^2 (\frac{v_i}{s_i})}{\sum_{v_i \in C_i} s_i^2},$$

minimizing of (3.17) is equivalent to maximize the following

$$\max_{c1,c2} \frac{(\sum_{i \in C1} s_i v_i)^2}{\sum_{i \in C1} s_i^2} + \frac{(\sum_{i \in C2} s_i v_i)^2}{\sum_{i \in C2} s_i^2}.$$

In addition, we can use the ordering of the scaled data points to design a more efficient algorithm to solve the bi-clustering problem, which is depicted in Algorithm (5). Obviously the complexity of Algorithm (5) is $O(n \log n)$, this allows us to deal with large-scale data set efficiently. We also point out that a similar idea had been employed by Shi and Malik [8] in their seminal paper on normalized k-cut for image segmentation. In that case, the 0-1 SDP model takes the form as in (2.8) with $k = 2$. Since $d^{\frac{1}{2}}$ is the eigenvector corresponding to the largest eigenvalue of the underlying coefficient matrix, Shi and Malik proposed to use the eigenvector corresponding the second largest eigenvalue of the coefficient matrix to cluster the data set. Shi and Malik also suggested several simple heuristics to solve the underlying 0-1 SDP model in

Algorithm 5 Refined weighted K-means in One Dimensional space

```
Step 1: Input the data set $\mathcal{V} = \{v_1, v_2, \cdots, v_n\}$ and scalar vector $s$, calculate the $\bar{v} = \frac{v_i}{s_i}, i = 1, \cdots, n$;
Step 2: Sort the sequence $\bar{v}_i$, such that $\bar{v}_{i_1} \geq \bar{v}_{i_2} \cdots \geq \bar{v}_{i_n}$, where $\{i_1, \cdots, i_n\}$ is a permutation of the index set $\{1, \cdots, n\}$;
Step 3:
for $l = 1$ to $n$ do
    calculate the center of two partitions,
        $C_1^l = \{\bar{v}_{i_1}, \cdots, \bar{v}_{i_l}\}, C_2^l = \{\bar{v}_{i_l+1}, \cdots, \bar{v}_n\}$,
    and calculate the objective function
        $f(l) = \frac{(\sum_{i \in C_1^l} s_i v_i)^2}{\sum_{i \in C_1^l} s_i^2} + \frac{(\sum_{i \in C_2^l} s_i v_i)^2}{\sum_{i \in C_2^l} s_i^2}$.
    if $f(l) < f(l-1)$ then
        output $C_1 = \{\bar{v}_{i_1}, \cdots, \bar{v}_{i_{l-1}}\}, C_2 = \{\bar{v}_{i_l}, \cdots, \bar{v}_{i_n}\}$ as the final solution
    end if
end for
```

one dimensional space. However, none of these heuristics in [8] can ensure to locate the global solution of the subproblem. On the other hand, in our algorithm we need to perform a sorting first and then find the best breaking point in term of the objective function. As we shall see from our later discussion, our algorithm allows us to obtain an approximation with guaranteed quality.

Theorem 3.4.1. *Suppose that $W = vv^T$ with $v \in \Re^d$ and $k = 2$. Then the 0-1 SDP model (3.1) can be solved by the refined weighted K-means in one dimensional space.*

Proof. To prove the theorem, we first recall the interrelation between the 0-1 SDP (3.1) and so-called weighted K-means stated in before, which implies that there exist two distinct weighted centers corresponding to the optimal solution of problem (3.1), denoted by C_1^* and C_2^* respectively. Without loss of generality, we can assume that $C_1^* > C_2^*$. From (3.17), we know that the optimal partition can be obtained by assigning the scaled data points to these two centers based on the distances from every point to the two points. Let us denote the final clusters by (C_1, C_1^*), (C_2, C_2^*). Since the data set is in one dimensional space, it is easy to see that

$$\bar{v}_1 \in C_1, \bar{v}_2 \in C_2,$$

which implies

$$\bar{v}_1 > \bar{v}_2.$$

This further implies that the optimal partition can be obtained by making use of the ordering information of the scaled data points. The proof of the theorem is finished.

Although the solution obtained from (5) is 2-approximation guaranteed, there may be a better assignment of points existed. The procedure of local search works for that purpose. Suppose that C_1 and C_2 are output solution of (5), we have

```
Procedure of Local Search
Repeat
    1. Calculate the set
         $\mathcal{L}_1 = \{v_i$ : the objective value is decreased if moving $v_i$ from $C_1$ to $C_2\}$,
         $\mathcal{L}_2 = \{v_i$ : the objective value is decreased if moving $v_i$ from $C_2$ to $C_1\}$;
    2. Update $C_1$ and $C2$
         $C_1 = C_1 \setminus \mathcal{L}_1 \cup \mathcal{L}_2,\ C_2 = C_2 \setminus \mathcal{L}_2 \cup \mathcal{L}_1$;
Until $\mathcal{L}_1$ and $\mathcal{L}_2$ are empty.
```

3.4.2 Divisive Hierarchical Clustering

Hierarchical clustering is mainly categorized into agglomerative methods and divisive methods in terms of the way to construct the tree of clusters. We have briefly given an introduction of an agglomerative clustering in the first chapter. Also, three commonly used methods to measure the similarity of two clusters during the merging process are discussed. In this section, we present a divisive clustering method. Usually divisive hierarchical clustering is more difficult to develop than agglomerative clustering. In [27], Hansen provided an algorithm to partition a data set divisively in low dimension, the algorithm runs in $O(n^{d+1}\log n)$time, where d is the dimension of the space to which the entities belong.

From Theorem (3.3.1) and (3.4.1), we know that the Algorithm (5) provides a 2-approximation solution for the 0-1 SDP model (3.1) for $k = 2$. When $k \geq 3$, we employ the divisive hierarchical approach to partition a data set into k clusters as follows. We first cut the data set into a two parts by Algorithm (5). Then, we separate each subset into two parts based on the similar strategy. After that, the best cut is selected in term of the reduction of the value of the objective function. The process continues until the number of clusters k is reached or all subsets can not be divided any more. Algorithm(6) shows the whole procedure.

Graph Model

In this subsection, we use Figure (3.1) to illustrate how the divisive hierarchical algorithm works. When $k = 2$, the data set is divided into two parts by using the refined weighted K-means clustering in one dimension. When $k = 3$, both groups are performed a binary separation through the same strategy,

Algorithm 6 Divisive Hierarchical Clustering for $k \geq 3$

```
**Step 1**: Calculate the affinity matrix for a given graph or a data set by (2.4) or (2.3);
**Step 2**: **Repeat**
    1. Calculate the projection of the coefficient matrix onto the null space of $s$, i.e.,
           $(I - ss^T)W(I - ss^T)$,
       and obtain the corresponding largest eigenvector $v$,
    2. Use $s = d^{\frac{1}{2}}$ and $v$ to obtain a binary separation of the graph or the data set by the Algorithm (5),
    3. Run the procedure of local search if necessary,
    4. Choose a suitable sub-partition to do further binary separation such that the objective value is minimal,
    5. Mark the sub-graph as unpartitionable if the size of sub-partition is too small or there is no splitting point found,
**Until** $k$ is reached or all subsets are unpartitionable.
```

the further separation is selected in term of the reduction of the value of the objective function. For example, the left separation with solid line is chosen, the right separation with dot line is stored in the memory for the next iteration.

The process continues until the number of clusters is reached.

It is worth to point out that to get i-cut, we only need to perform the binary separation for two subsets which are newly obtained from the (i-1)-th cut.

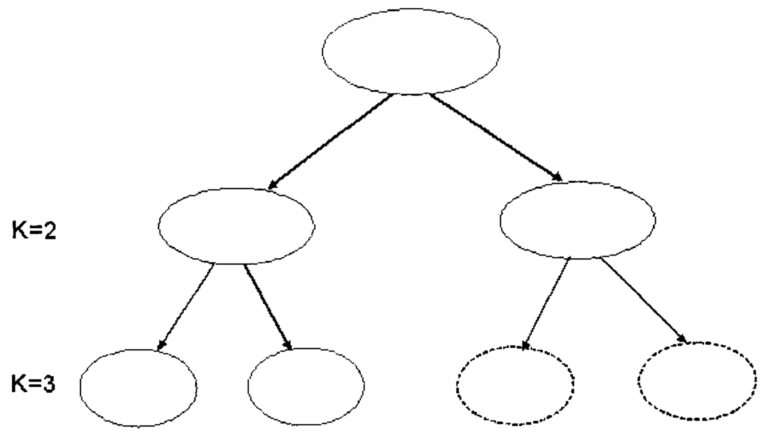


Figure 3.1: Graph for the Divisive Hierarchical Approach

Complexity

In the sequel we discuss briefly the complexity of Algorithm (4). First, we discuss the complexity on computing the largest eigenvector of the coefficient matrix. If we use the power method [9] to calculate the largest eigenvector, it takes $O(n^2)$ time. If the matrix W has a certain sparsity, then we can use its sparse structure to speed up process, especially this is for image segmentation. In the context of the classical K-means and weighted K-means clustering, we can sue the structure of the underlying matrix W to improve the process. Recall that we have $W = W_v W_v^T$ where $W_x \in \Re^{n\times d}$ is a matrix such that each row represents a point in $\Re^d$. In such case, it is not necessary to calculate the eigenvalues and corresponding eigenvectors of the matrix W directly. It can be proceed in the following way. We first compute a matrix $\bar{W} = W_v^T W_v \in \Re^{d\times d}$, which takes $O(nd^2)$ time. It is easy to see that the matrix $\bar{W}$ has the same spectrum as that of the matrix W. Thus, we can compute a singular decomposition o $\bar{W}$ directly such that

$$\bar{W} = V diag(\lambda_1, \cdots, \lambda_d) V^T,$$

where $V \in \Re^{d\times d}$ is an orthogonal matrix such that every column is an eigenvector of $\bar{W}$. This takes $O(d^3)$. We then calculate the matrix $U = W_v V$ in $O(nd^2)$. One can easily verify that the i-th column of the matrix U is an eigenvector of W corresponding to the eigenvalue λ_i. Therefore, the total computational cost to calculate the eigenvalues and their corresponding eigenvector is $O(nd^2 + d^3)$. If d is not very large, say $m < 1000$ (which is true for most data set in practice), then we can obtain the eigenvalues and eigenvectors of W very quickly.

Next we discuss the complexity in solving the subproblems by Algorithm (6). The binary clustering can be done in $O(n \log n)$ time by using the Algorithm (5). Therefore, the total time complexity of the algorithm will be $O(kn \log n + kn^2)$, in which the procedure of local search is not under the consideration. This allows us to cope with relatively large data set in high dimension.

Chapter 4

Numerical Experiments

In this chapter, we report some preliminary experiments based on the algorithm discussed before. It includes two parts. In the first part, test problems are focus on image segmentation, which is solved by using 0-1 SDP model for normalized cut. In the second part, we partition regular data sets via 0-1 SDP model for K-means clustering.

4.1 Numerical experiments for image segmentation

We have implemented our algorithms in matlab 6.5 combined with C language. The tests are done on a PC with AMD Athlon 1.24G CPU and 512M RAM. In our implementation, the largest eigenvalue and its corresponding eigenvector of the coefficient matrix are computed by Lanczos tridiagonalization with the modified partial orthogonalization [10].

For comparison, we have also tested the algorithm in [8] where the authors used a simple heuristics to attack the subproblem of Algorithm (4) in one-dimensional space. Also, we use SDPCut and Ncut to denote the algorithm in the thesis and the algorithm in [8] for convenience.

We have tested our algorithm on several examples in the literature. These includes that the data point set from [8], an image from the Berkeley segmentation data set and benchmark [1], and two images from Adobe Photoshop for image segmentation. Due to the limit of the computational facility in our experiments, we resized all images proportional to their original sizes. To compare the performance of our algorithm on the test problems, the numerical results are summarized in tables and figures. In the tables we list the values of the objective function obtained by our algorithm (SDPCut) and the algorithm(Ncut) in [8] respectively. The CPU time used to obtain the partitioning is also reported. The figure visualizes the segmentations that can help us to better understand the practical performance of two algorithms.

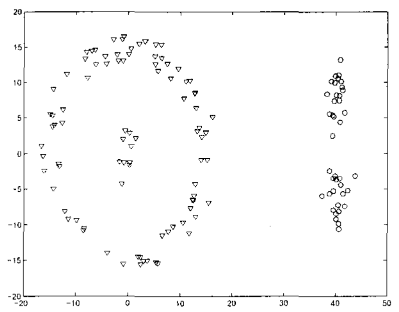
[1]http://www.eecs.berkeley.edu/Research/Projects/CS/vision/grouping/segbench/

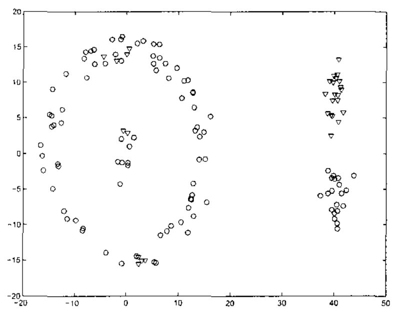
Point Data Set

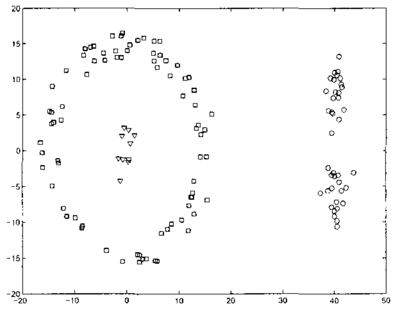
The first test problem is the point data set in [8], which consists of 130 points in two-dimensional space, the detail description of the points has been given in the first chapter. Table (4.1) records the values of the objective functions and CPU time for both algorithm for k from 2 to 4. Although both algorithms have exactly same partition for the point data set when $k = 4$, the values of the objective function provided by SDPCut are better than what provided by Ncut when $k = 2, 3$. The numerical improvement has been confirmed by figure (4.1), which shows that the cut provided by our algorithm separate the data set into two parts clearly when $k = 2$, while the separation provided by Ncut is unclear. When $k = 3$, though both methods manage to separate the data set, the final clusters are different.

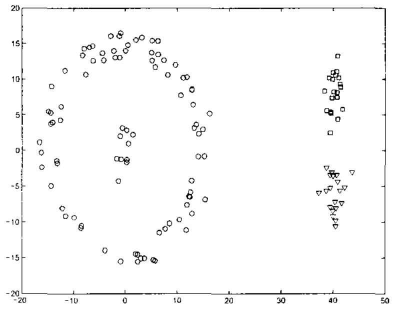
	Objective Value		CPU Time (s)	
K	SDPCut	Ncut	SDPCut	Ncut
2	0.000000000	0.134910574	0.44	0.27
3	0.000666557	0.002806832	0.53	0.23
4	0.003473389	0.003473389	0.63	0.27

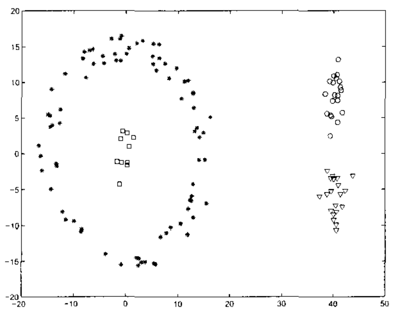
Table 4.1: Performance of Circle data


(a) SDPCut K=2


(b) Ncut K=2


(c) SDPCut K=3


(d) Ncut K=3


(e) SDPCut K=4

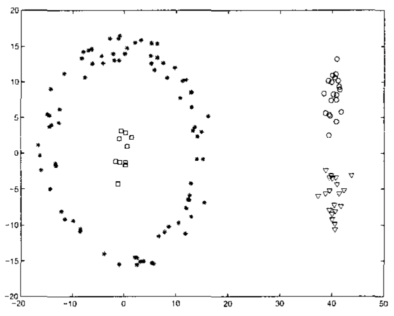

(f) Ncut K=4

Figure 4.1: Partitions of circle data point set

Woman Face

The second test problem is from the Berkeley segmentation data set and benchmark. Figure (4.2) shows is an image of a woman's face with two hands on her cheek in gray format. The original size is 481×321 pixels and it was resized as 139×91 pixels in the experiments. The similarity measure between each pair of pixels can be summarized in $n \times n$ weighted matrix W based on brightness and spatial features, where n is the total number of pixels in the image. Note that the weighted matrix W contains large number of zeros and near zeros due to the spatial proximity factor

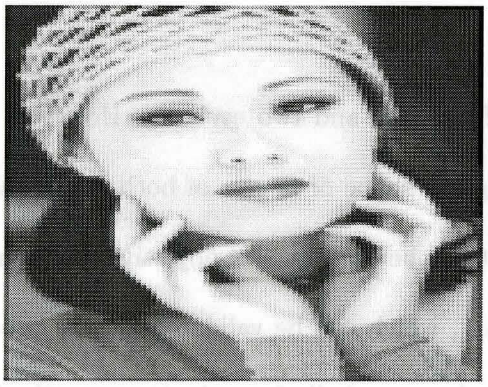


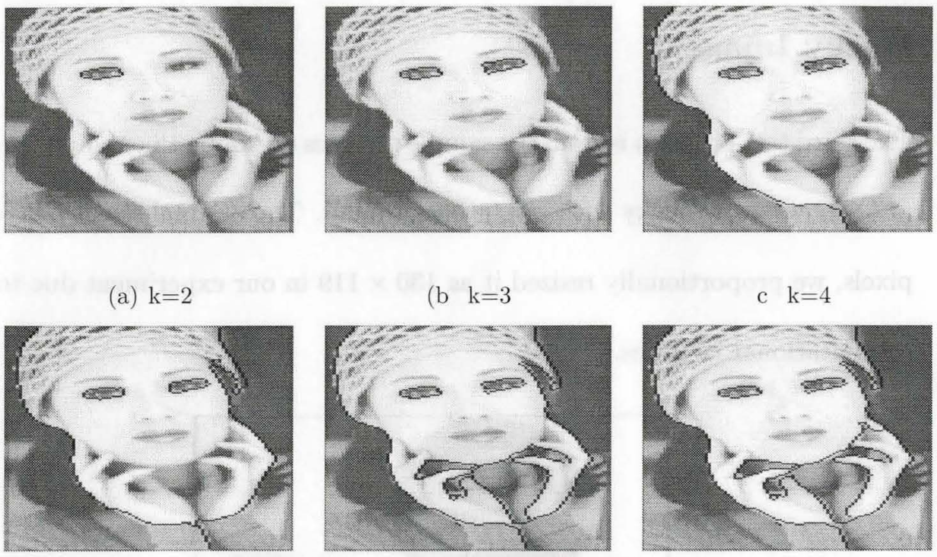
Figure 4.2 Woman Face Image

Table (4.2) records the performance of segmentations of the woman face for k from 2 to 7 Figure (4.3) and (4.4) illustrates the whole cutting process for SDPCut and Ncut respectively Both algorithm are able to find

	Objective Value		CPU Time (s)	
K	SDP Ncut	Ncut	SDP Ncut	Ncut
2	0.000373422	0.00267505	27.22	36.97
3	0.000951509	0.01056890	44.45	35.81
4	0.003592392	0.01115780	57.90	35.36
5	0.012211270	0.05035214	74.00	33.09
6	0.043300375	0.05025529	88.13	35.58
7	0.070311355	0.16297105	95.49	34.44

Table 4.2: Performance of Woman Face

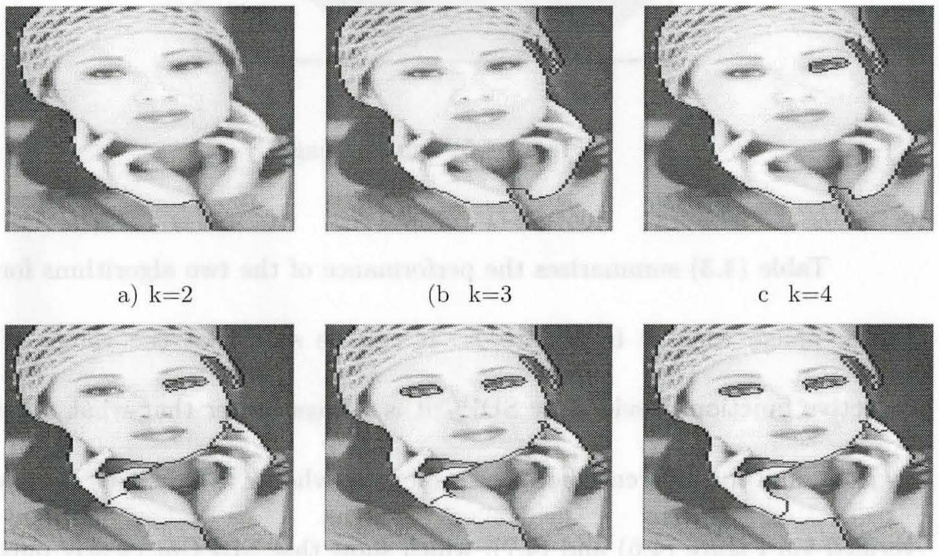
the boundary of the face and two eyeballs. However, SDPCut did a better job on extracting the shape of fingers of both hands than Ncut. It is easy to see that SDPCut has better values of objective function than Ncut, especially when $k = 2, 3, 4, 7$, which can be validated via the process of segmentations. SDPCut extracted two eyeballs from the image at the beginning. After that, it figured the outline of the left face, Next, the right face is identified. Finally, SDPcut discovered the shape of fingers of both hands. On the other hand, Ncut partitioned the image in the different sequence, it started with both sides of the face, and then two eyeballs. Moreover, Ncut made a cut somewhere in the shadow, while SDPCut outlined the shape of the fingers.



(a) k=2 (b k=3 c k=4

d) k=5 e) k=6 (f) k=7

Figure 4.3. Segmentations of the Woman Face from $k = 2$ to 7 by SDPCut



a) k=2 (b k=3 c k=4

d) k=5 e k=6 (f) k=7

Figure 4.4: Segmentations of the Woman Face from $k = 2$ to 7 by Ncut

Ducky Image

The third test problem is from the sample pictures of Adobe Photoshop. Figure (4.5) shows a toy ducky with bright background. The original size is 546×500 pixels, we proportionally resized it as 130×119 in our experiment due to the computational facilities.

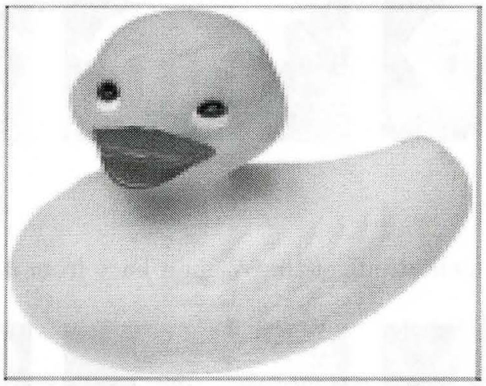


Figure 4.5· Ducky Image

Table (4.3) summarizes the performance of the two algorithms for the Ducky image when k from 2 to 7 It can be seen that the values of the objective function provided by SDPCut is always better that what obtained by Ncut, and the difference becomes prominent when $k = 6, 7$ This can also be verified via Figure (4.6) and (4.7), which show that SDPCut clearly outlined the shape of the mouth and the contour of ducky's stomach for $k = 6$ and $k = 7$ respectively, while Ncut did not.

	Objective Value		CPU Time (s)	
K	SDPCut	Ncut	SDPCut	Ncut
2	0.006035151	0.006252636	44.80	94.03
3	0.010534337	0.012925502	77.80	86.23
4	0.022542201	0.027505889	102.47	86.11
5	0.033214296	0.042012681	121.35	82.22
6	0.052768125	0.094367725	137.57	94.39
7	0.108721513	0.199546366	147.52	83.69

Table 4.3: Performance of Ducky Image

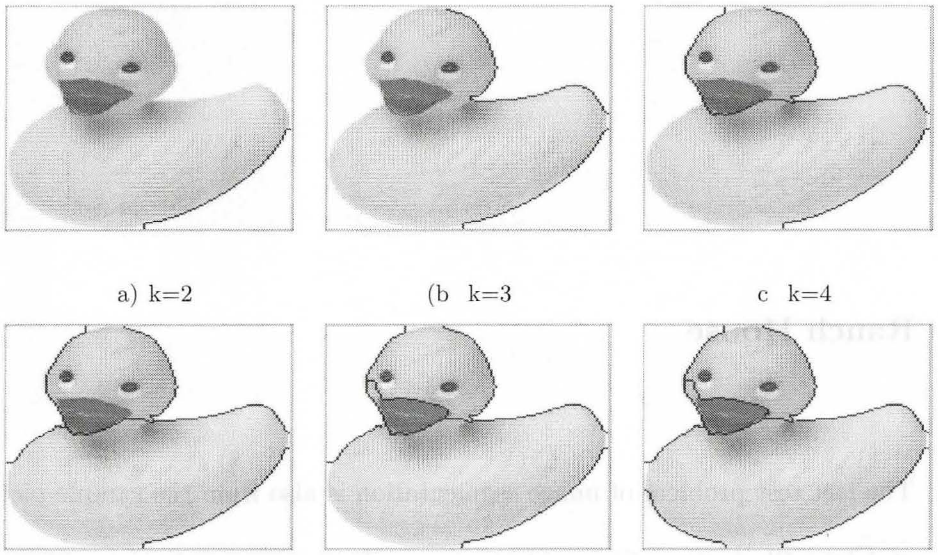
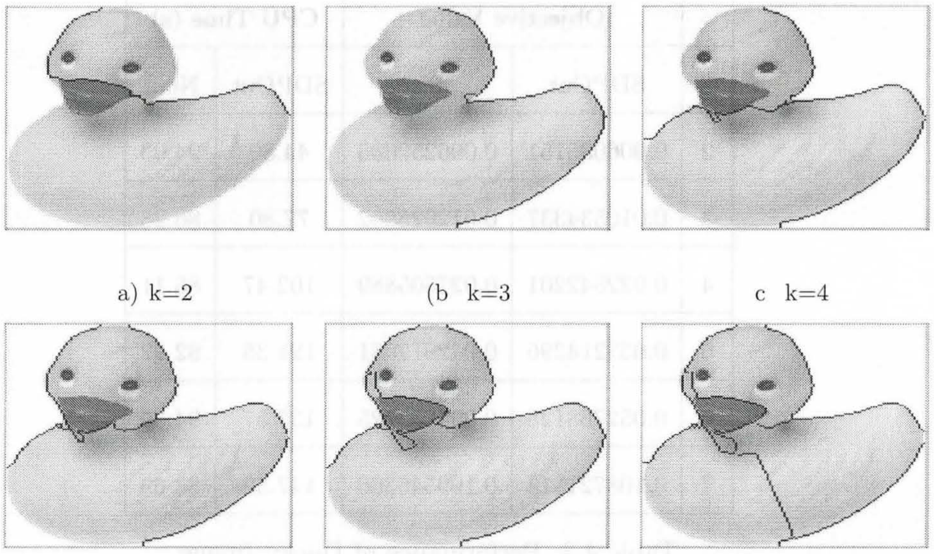


Figure 4.6: Segmentations of the Ducky from $k = 2$ to 7 by SDPCcut



a) k=2 (b k=3 c k=4

d) k=5 e k=6 (f) k=7

Figure 4.7 Segmentations of the Ducky from $k = 2$ to 7 by Ncut

Ranch House

The last test problem of image segmentation is also from the sample pictures of Adobe Photoshop. Figure (4.8) is a gray level image of the scene of a ranch house. The original size is 692×589 pixels, we proportionally resized it as 140×119 in our experiment.

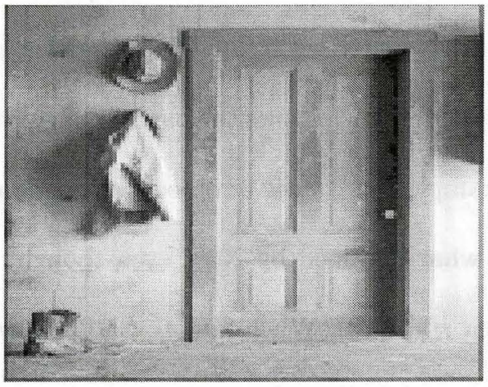
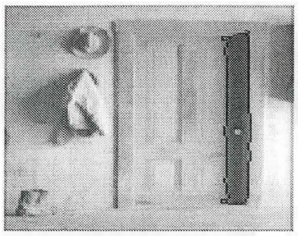


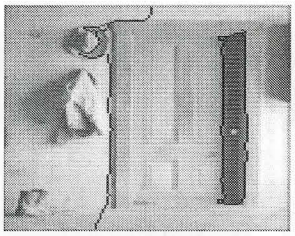
Figure 4.8. Ranch House Image

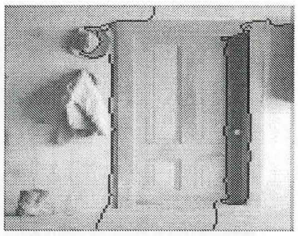
	Objective Value		CPU Time (s)	
K	SDP Ncut	Ncut	SDP Ncut	Ncut
2	0.000618508	0.000830058	50.69	94.34
3	0.014763275	0.014937043	88.19	94.55
4	0.044782877	0.067792577	112.19	90.03
5	0.078499524	0.108438306	126.78	84.45
6	0.086836495	0.173986842	138.06	97.83
7	0.095264966	0.182797317	149.09	95.44
8	0.117966244	0.235106454	161.67	99.13

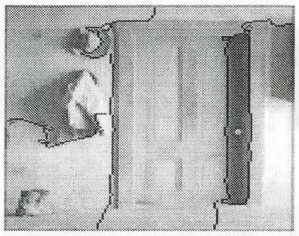
Table 4.4: Performance of Ranch House

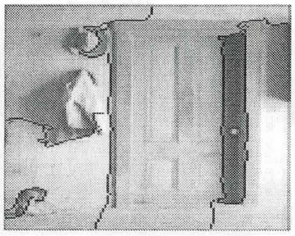
Table (4.4) gives the performance of segmentations of the Ranch House image for k from 2 to 8. It indicates that SDPCut always has better computational results compared to Ncut. The results can be demonstrated by Figure(4.9) and (4.10), which shows the partitioning process of two algorithms in detail. For example, the values of objective function provided by SDPCut are better than what obtained by Ncut even thought both algorithms have similar cuts when $k = 2, 3$. When $k = 6$, SDPCut was able to outline the contour of the shoes while Ncut drew out the shadow of the door. In addition, SDPCut gave a clearer shape of the shadow of the cloth than Ncut when $k = 8$.

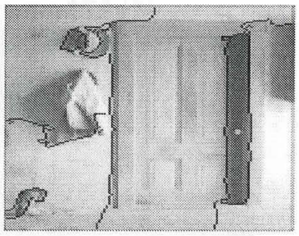

a) k=2


(b k=3


c k=4


d) k=5


e k=6


(f) k=7

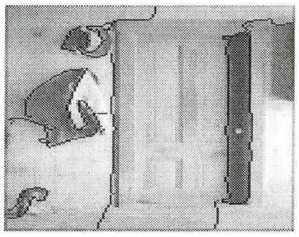

(g) k=8

Figure 4.9· Segmentations of the Ranch House from $k = 2$ to 8 by SDPCcut

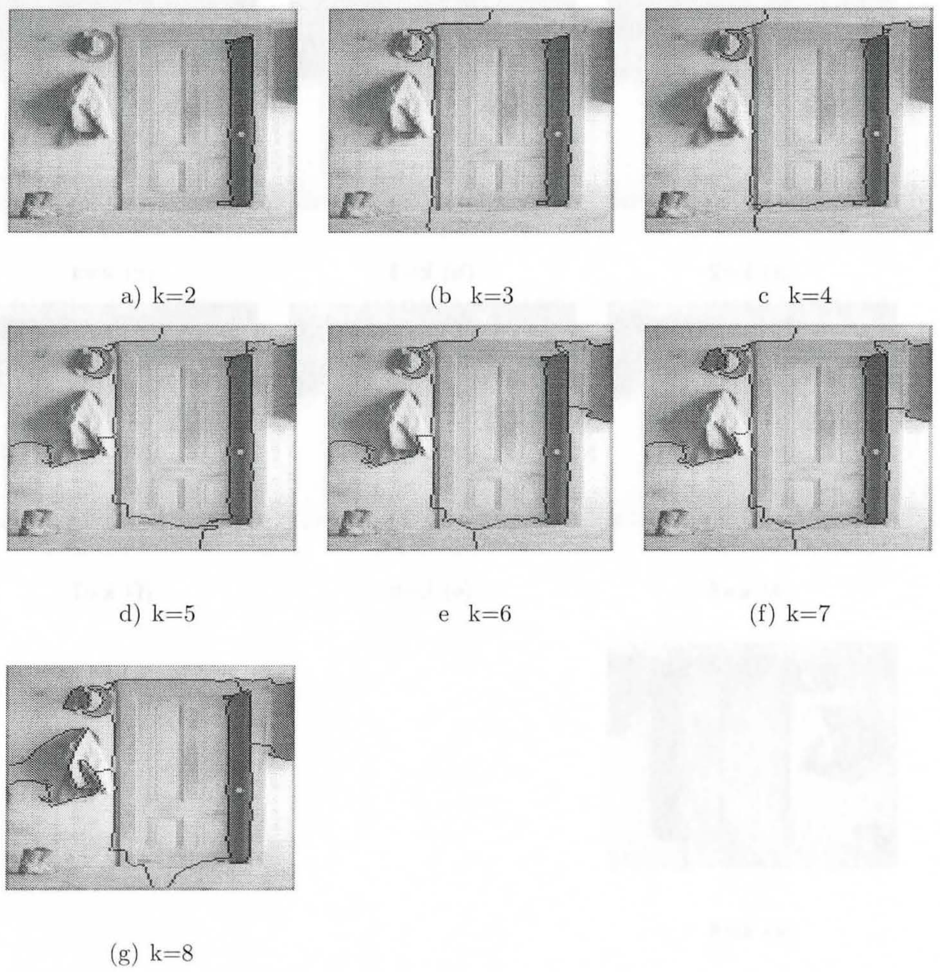


Figure 4.10: Segmentations of the Ranch House from $k = 2$ to 8 by Ncut

Local Search

We also test our algorithm by applying local search. The test was only restricted on three images in our experiments since the data point set was already separated clearly To compare the computational result on the test examples between without applying the local search and with performing the local search, we still use tables to list the values of objective function and CPU time used to obtain the partitions.

	Objective Value		CPU Time (s)	
K	SDP Ncut	Local Search	SDP Ncut	Local Search
2	0.000373422	0.000373422	27.22	37.26
3	0.000951509	0.000951509	44.45	72.60
4	0.003592392	0.003328935	57.90	175.08
5	0.012211270	0.011493237	74.00	275.32
6	0.043300375	0.041677942	88.13	353.82
7	0.070311355	0.065115482	95.49	384.52

Table 4.5: Performance of Woman Face with Local Search

Table (4.5) summaries the result of the Woman Face image. The objective value has no any changes for both algorithms when $k = 2, 3$. There are some slight improvements on the objective value while applying the local search when K from 3 to 7, but the computational time is much more than what used without applying the local search.

Table (4.6) records the testing performance of the Ducky image. The values of objective function provided by the algorithm with the local search is always better than what obtained without applying the local search, the

	Objective Value		CPU Time (s)	
K	SDPCut	Local Search	SDPCut	Local Search
2	0.006035151	0.005952198	44.80	209.75
3	0.010534337	0.010169767	77.80	279.02
4	0.022542201	0.020623691	102.47	619.64
5	0.033214296	0.030920484	121.35	671.81
6	0.052768125	0.048302935	137.57	761.28
7	0.108721513	0.102556240	147.52	776.78

Table 4.6: Performance of Ducky Image with Local Search

average improving is around 5%. However, its CPU time needed is more than five times that spent on the algorithm without the local search.

Table (4.7) shows the performance of both algorithms for the Ranch House image. The algorithm with applying local search took 24 minutes for the image segmentation when $k = 3$, which is more than 16 times that spent without local search. However, the objective value only has 4% enhancement. When $k = 6, 7, 8$, the objective values of two algorithm became almost same though there are some progressive results after applying local search in the previous segmentations.

From the above experiments, it can be seen that SDPNcut with local search can provide better local optimal with much longer running time. Also,

	Objective Value		CPU Time (s)	
K	SDP Ncut	Local Search	SDP Ncut	Local Search
2	0.000618508	0.000618508	50.69	59.80
3	0.014763275	0.014176416	88.19	1418.57
4	0.044782877	0.022385691	112.19	1559.27
5	0.078499524	0.051795732	126.78	1627.99
6	0.086836495	0.084718791	138.06	1662.52
7	0.095264966	0.093040816	149.09	1689.21
8	0.117966244	0.111911743	161.67	1724.50

Table 4.7: Performance of Ranch House with Local Search

whether apply local search during the hierarchical approach depends on how to seek the trade-off between the quality of clustering and the CPU time.

4.2 Numerical experiments for data sets

In this section, we test the 0-1 SDP Model for K-means clustering on several data sets. For convenience, we denoted the algorithm by SDPKmeans in the thesis. For comparison, we also test those data sets by SDPcut, in which the affinity matrix is obtained by (2.3). Since the objective function of both algorithm are different, we compute values of the objective function by substituting the assignment matrix X into MSSC of K-means clustering (2.1). Moreover, the procedure of local search are applied in both algorithms.

Soybean Data

The first data set we use to test is Soybean data (small) from the UCI Machine Learning Repository [2], see also [31]. This data set has 47 instances and each instances has 35 normalized attributes. It is known that the data set has 4 cluster, the global optimun has already been reported in [2] by using a linear programming mode. Table (4.8) summaries the testing result of two algorithms. It shows that SDPKmenas can find the optimum when $k = 2, 3$, but not in the case when $k = 4$. However, the optimum can be reached after applying the local search. In contrast, SDPNcut is able to discover the optimum in all k with or without executing the local search.

	SDPKmeans				SDPNcut			
K	Objective	CPU	Local Search	CPU	Objective	CPU	Local Search	CPU
2	404.4593	0.08	404.4593	0.16	404.4596	0.34	404.4596	0.45
3	246.4593	0.13	246.4593	0.23	246.4596	0.42	246.4596	0.76
4	205.9637	0.14	205.9637	0.28	230.3923	0.47	205.9637	0.86

Table 4.8: The Soybean Data

[2]http://www.ics.uci.edu/mlearn/MLRepository.html

Ruspini Data

The second test data is Ruspini data set from [34]. The data set consists of 75 points in $\Re^2$ with four groups, it is popular for illustrating clustering technique [35]. Table (4.9) lists the numerical results of two algorithms. The values of objective function provided by SDPKmeans have significant improvement after applying the local search for all k, the consumed CPU time is twice over that in SDPKmeans without performing the local search. On the other hand, the values provided by SDPNcut are exactly same with or without applying the local search.

	SDPKmeans				SDPNcut			
K	Objective	CPU	Local Search	CPU	Objective	CPU	Local Search	CPU
2	126400	0.09	123950	0.17	123950	0.36	123950	0.44
3	62790	0.13	57430	0.25	57430	0.49	57430	0.72
4	19408	0.14	15246	0.55	15246	0.52	15246	0.63

Table 4.9: The Ruspini Data

Späth postal Zones Data

The last test data is the the Späth postal zones data from [33]. This data set contains 89 entities and each entity has 3 features. It is known that the data set has 9 groups.

	SDPKmeans				SDPNcut			
K	Objective	CPU	Local Search	CPU	Objective	CPU	Local Search	CPU
2	$6.026*10^{11}$	0.13	$6.026*10^{11}$	0.17	$7.541*10^{11}$	0.36	$6.026*10^{11}$	0.48
3	$3.637*10^{11}$	0.14	$3.637*10^{11}$	0.27	$4.102*10^{11}$	0.42	$3.637*10^{11}$	0.98
4	$2.743*10^{11}$	0.17	$2.743*10^{11}$	0.36	$2.707*10^{11}$	0.52	$2.743*10^{11}$	1.05
5	$2.428*10^{11}$	0.22	$2.428*10^{11}$	0.38	$8.102*10^{10}$	0.61	$2.428*10^{11}$	1.14
6	$2.392*10^{11}$	0.23	$2.392*10^{11}$	0.42	$4.982*10^{10}$	0.64	$2.392*10^{11}$	1.19
7	$2.117*10^{11}$	0.27	$2.113*10^{11}$	0.44	$4.800*10^{10}$	0.66	$2.113*10^{11}$	1.25
8	$2.060*10^{11}$	0.28	$2.060*10^{11}$	0.52	$2.224*10^{10}$	0.78	$2.060*10^{11}$	1.33
9	$2.011*10^{11}$	0.33	$2.011*10^{11}$	0.55	$1.546*10^{10}$	0.81	$2.011*10^{11}$	1.44

Table 4.10: The Späth's Postal Zone Data

The performance of two algorithms are shown in Table (4.10). SDP-Kmeans provided the same objective values for all k with or without performing the local search. SDPNcut is unable to reach the optimum when $k = 2$. With performing the local search, the objective values was improved when $k = 2, 3$. Nevertheless, the objective values became worse on the following partitions while applying the local search. The reason is that solution discovered by SDPNcut is still local optimum even though the local search can provide better separation in i-th iteration, the quality of the next partition based on the local search may not be better than that without performing the local search.

In our experiments, we conclude that the procedure of local search should be used to derive the better solution without increasing the running time too much for SDPKmeans . For SDPNcut, it may not be a good option to perform the local search after each binary separation, we suggest that applying the local search after achieving the final partition.

Chapter 5

Case Study

In this chapter, we start discussing the project sponsored by MITACS *and* Rogers Communication Inc.. *The purpose of the project is to extract a pattern to predict the children information of customers based on their buying behaviors. The project mainly includes three parts. In the first part, we prepared the data set for the modeling, it contains the step of data collection and data preprocessing. In the second part, we applied the clustering analysis to group customers with a similar children composition in postal code level according to the Gen5 demographic data, in which the 0-1 SDP model for weighted K-means clustering(2.14) was employed to segment customers. In the last part, we discovered the link between the segmentation of customers and their shopping behaviors.*

In today's competitive market, a company want to know what it should do to meet the demands from its customers or to attract new customers from a certain region, which kind of strategies should be employed by the company to maintain a profitable market share. The important information about customers such as the customer expenditure, the customer family income, and the customers' family composition will help the company a lot in its decision making.

The purpose of the project sponsored by *MITACS* and *Rogers Communication Inc.* is to extract the interesting pattern to predict the children composition of customers based on their buying behaviors by using the combination of clustering analysis and classification. In addition, SAS language and and Enterprise Miner are required tools to build up the model. Although the model is particularly focused on customers of video stores in Calgary region, the methodologies of data mining applied in the project can be easily extended to detect other customer information.

5.1 Data Preparation

The first step of the process of modeling is data collection, it plays extremely important role of building a successful predictive model. In this section, we show how the time frame of data collection is determined and give the detail description of three data sources.

5.1.1 Time Frame

The time element is critical to model building[16]. Usually it is separated into the past, the present and the future. The past data is used as historical data to build model, it further divided into training data set, testing data set and validation data set during the modeling process. The future data is what we are trying to predict, it used to score the model. The present data is the time that takes to gather data and develop the model.

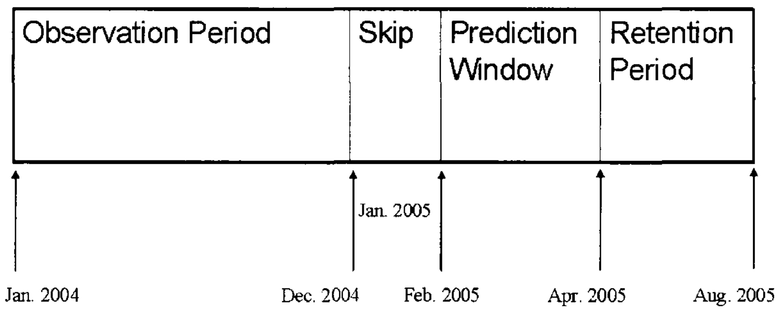


Figure 5.1: Time Frame in Data Collection

Figure (5.1) shows the time frame of the project. Since the project was started at May 2005, we assume that January 2005 is the present time to collect data. The transaction data of last 12 months are treated as the input data set to build up the model. The data between February and April are used as the scoring data set, which is to evaluate how well the model performs. Moreover, the model will be shaped by shifting the time frame one month recursively during the retention period.

5.1.2 Data Source

The project has three data sources, the customer information table, the customer transaction table and the Gen5 demographic data.

Customer Transaction Table

The transaction table contains dynamic information wrapped in 153592 customers, including 4794919 transaction records within 20 different video stores. Table (5.1) lists the variables of the transaction table. Those variables are identified by three following perspectives of customer buying behaviors.

- **Recency:** What is the customer most recent purchase?
- **Frequency:** How often has the customer made a purchase in a predefined length of time?
- **Monetary:** How much money has the customer spent over the measured time period?

Variables Name	**Description**
CUSTOMER	Customer ID (Primary Key)
SITE_ID	Store ID (Primary Key)
POSTCODE	Customer postal code
CUST_TYPE	Customer type (BAS,EMP,VIP,RGC)
TRA_DATE	Transaction Date
TRA_TIME	Transaction Time (Morning, Afternoon, Evening)
DEPARTMENT	Department of items *
SUB_DEPARTMENT	Sub_Department of Items *
CLASS	Class of items *
SUB_CLASS	Sub_Class of Items *
QUANTITY	Number of items per transaction
TOTAL_PAYMENT	Total payment of the transaction
TOTAL_DISCOUNT	Total discount of the transaction
FREQUENCY	Frequency of the transactions occurred

Table 5.1: Transaction Variables for Video Customers

Customer Information Table

The customer information table contains all the static information of video customers in the city of Calgary, such as the name of customers, the living address, the type of customers, and etc. It consists of 233875 records. Table (5.2) shows some important variables.

Variables Name	**Description**
CUSTOMER	Customer ID (Primary Key)
SITE_ID	Store ID (Primary Key)
FIRS_ NAME LAST_NAME	Customer Name
POST_CODE	Customer Postal Code
CLIENT_TYPE	Client type (BAS,EMP,VIP,RGC)
ENTER_DATE	Start Membership date
CC_TYPE Credit	card type
CC_NO	Encrypted credit card number
...	...

Table 5.2: Static Variables for Video Customers

Gen5 Demographic Data

Gen5 demographic data is a demographic and socio-economic marketing data file. It provides the census information of 2004 for over 3600 demographic variables, projected to the postal code level. The data file contains many kinds of information of households, including the information of expenditure, dwelling, education, employment and family composition. In the project we are only interested at variables that are relevant to the children information. Table (5.3) and (5.4) displays the direct and indirect variables respectively.

Variables Name	**Description**
GHWCNC	Common-law couples without children
GHWMNC	Married-couples without children
GHWCWC1	Common-law couples with 1 child at home
GHWMWC1	Married-couples with 1 child at home
GFMLPF1	Lone female parent with 1 child
GFMLPM1	Lone male parent with 1 child
GHWCWC2	Common-law couples with 2 children at home
GHWMWC2	Married-couples with 2 children at home
GFMLPF2	Lone female parent with 2 children
GFMLPM2	Lone male parent with 2 children
GHWCWC3U	Common-law couples with 3 children or more at home
GHWMWC3U	Married-couples with 3 children or more at home
GFMLPF3U	Lone female parent with 3 children or more
GFMLPM3U	Lone male parent with 3 children or more
GC05FAM	Proportion: Children 0 - 5 years in family
...	...

Table 5.3: Directed Children Variables in Gen5 Data

Variables Name	**Description**
GBANDHOUP	proportion of total private Hhds - Band Housing
GOWNDWLGP	proportion of total private Hhds - Owned
GRENTDWLP	proportion of total private Hhds - Rented
GSINGLEP	never married proportions
GMARRIEDP	legally married proportions
GWINDOWEDP	Windowed Proportions
GDIVORCEP	Divorced Proportions
GSEPARATP	Separated Proportions, but still legally married
GHHDS1PP	proportion of household size for 1 person
GHHDS2P	proportion of household size for 2 persons
GHHDS3P	proportion of household size for 3 persons
...	...

Table 5.4: Correlated Children Variables in Gen5 Data

5.2 Data Preprocessing

Data preprocessing is to smooth the raw data such that the input data becomes easy and effective to handle for the modeling by any type of processing. For example, fixing the missing value, identifying or removing outliers, resolving inconsistencies records, and etc. The essential of operations is that any processing taken should service the target of the project. Data preprocessing is a fundamental part of whole modeling. If there is not qualified data provided, no one could trust the mining result. Usually data preprocessing occupies about 70% workload in a data mining task [16]. Since the missing value can be handled by SAS automatically, it is out of our operations. In this subsection, we discuss the process of data cleaning on variable creation and transformation, outliers detection and exclusion of irrelevant information.

5.2.1 Variable Creation

In the project, the variable creation is approached from three aspects. Firstly, since different type of families have different kind of children information in the demographic data, it is not efficient way to look at every single family with the specific children composition. Thus, new variables are created to aggregate children information. For example, the variables of common-law couple with one child, married-couple with one child, lone female parent with one child, lone male parent with one child are integrated into the variable of a family with one child. Secondly, there may be exact transactions occurred during a time period. The variable called frequency is created to record such shopping behaviors in the transaction data. Finally, a shopping transaction may contain two or more items that belong to the same category. It is necessary to add new variables to summarize the total amount of payment and discount a purchasing.

5.2.2 Variable Transformation

The variable of transaction time is represented in *hhmmss* format in the transaction table. Since the numerical format is not useful to analyze customers' shopping behaviors during a time period, we convert the variable into the categorical format with three values. The following shows the mapping of the transformation.

Old Transaction Time	New Transaction Time
080000 ~ 120000	Morning
120000 ~ 170000	Afternoon
170000 ~ 240000	Evening

5.2.3 Outliers Detection

Filtering extreme values and errant data from the data set tends to produce better models. In the data preprocessing, we choose extreme percentiles as the filtering method for numerical variables, in which observations with interval variables in the extreme pth percentiles are excluded. The default values of upper and lower thresholds are specified as 0.5%. For categorical varibles, we use minimun frequency as the filtering method, any variables with a rare level that has a count less than the minimum frequency are dropped. For example, the the cutoff value of minimun level of catigrorical variables in the transaction table is set as 2. Table(5.5) gives the result of filtering process.

Table	Before Filter	After Filter	Outliers	Percentage
Demographic Data	72153	70678	1475	2.04%
Transaction Data	2327572	2286001	41571	1.79%

Table 5.5: Result of Filtering

5.2.4 Exclusion of Customer Information Data

The objective of our project is to discover the interrelationship between customers buying behaviors and their children composition. The scope of project is focused on regular customers, so that some special individuals or groups should be excluded. For example, the employees, the VIP customers, and the group of company are not under the consideration of the project. In the table of customer information, we remove such special customers from the analysis. Table (5.6) gives the result of exclusions.

Table	Before Excl.	After Excl.	Exclusives	Percentage
Customer Data	233875	230902	2973	1.27%

Table 5.6: Result of Exclusion

5.3 Clustering Analysis

On observations of customers' buying behaviors we find that there exists the interrelationship between customers' shopping tastes and their children information. For example, families with similar information of children tend to rent some videos about kids or buy candy bars. In addition, the tendency of the purchasing behaviors are varied with the different children information. Therefore, we like to know the nature grouping of customers in the perspective of children information before discovering the link between the customers' buying behaviors and their children composition. Clustering analysis is applied for that purpose. In this section, we first determine a good number of segmentation of customers. Then we partition customers on the good number of clusters, in which 0-1 SDP model for weighted K-means clustering is applied. In addition, the result is compared with the classical K-means clustering. After that we validate the partition of customers. Finally, we select the most suitable segmentation of customers for the further analysis.

5.3.1 Number of Cluster

To find the good number k of clusters that match the natural structure of customers is the first issue of clustering analysis. In the project, the potentially good number of clusters is determined by analyzing two statistic measurements from the agglomerative clustering history, called pseudo F statistic and pseudo t^2 statistic [7]. For convenience, they are denoted as PSF and $PSF2$ respectively, calculated by

$$PSF = \frac{\frac{T-P_G}{G-1}}{\frac{P_G}{n-G}}; \qquad PSF2 = \frac{B_{KL}}{\frac{W_K+W_L}{N_K+N_L-2}}$$

where T denotes the total sum-of-squared Euclidean distance in whole data set. P_G denotes the total sum-of-squared Euclidean distance over clusters at the Gth level of the hierarchy. n is the number of observations. G represents the number of clusters at Gth level of the hierarchy. N_K denotes the number of observation in cluster C_K. B_{KL} denotes $W_M - W_K - W_L$ if cluster C_K and C_L are merged into the new cluster C_M with the total sum-of-squared Euclidean distance W_M.

From the definition of two measurements, it is easy to understand that the relatively large values implies a good point of separation in PSF. For $PSF2$, a good number of cluster can be found at the first value which is markedly larger than the previous value and move back by one cluster.

Table (5.7) shows the values of two measurements for k from 1 to 15. Three merging methods, centroid, average links and ward's minimum-variance, are adopted during the merging process. By the analysis of PSF and $PSF2$, we conclude that 5 and 12 are the good stopping points indicated by the centroid method, the ward method shows that the data set can be clustered

Method	**Centroid**		**Ward**		**Average Link**	
NCL	PSF	$PSF2$	PSF	$PSF2$	PSF	$PSF2$
15	327.0	8328.0	5455.0	461.0	1682.0	44.2
14	296	692	5570	1913	1515	6049
13	310	251	5688	3472	1164	4802
12	334	33.8	5826	3385	1265	138
11	351	277	6017	1059	1297	802
10	274	999	6202	1859	486	8035
9	297	435	6470	1438	545	23.6
8	196	978	6803	3252	611	122
7	151	463	7077	2888	544	964
6	175	138	7528	1823	452	967
5	217	5.9	8045	5269	550	58.3
4	96.9	573	8089	5156	402	975
3	121	41.3	8362	7424	363	476
2	54.3	188	8665	7581	149	575
1	.	54.3	.	8665	.	149

Table 5.7: Hierarchical Clustering History

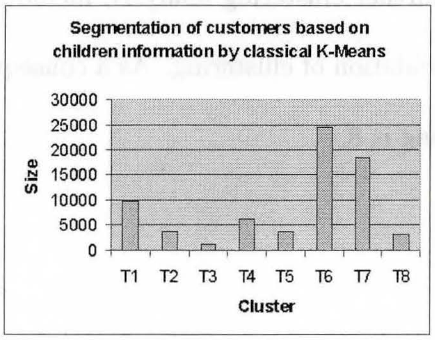
into 6 clusters or 15 clusters, the average link method suggests that the possible number of partitions is 8, 11 or 15. Because the children information is only composite of the age, the size, and the type of households, in other words that it can not be partitioned into many groups in reality, we choose 5, 6 and

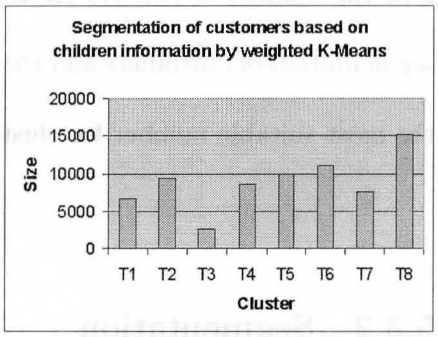
8 as the number of clusters for the further clustering analysis, including the segmentation of customers and the validation of clustering. As a consequence, the most suitable number for clustering is 8.

5.3.2 Segmentation

We use 0-1 SDP model for weighted K-means clustering (2.14) rather than the classical K-means clustering to segment customers in the perspective of children information for $k = 8$ that obtained from the last subsection. The reason is that the number of families of a postal region signifies the different distribution of the population, which means that it is not a realistic or fair way to segment customers by treating each instance of the demographic data equally. In the project, the weight attribute of instances is introduced by the number of families, which is taken into account while calculating the weighted MSSC (1.2) and the weighted center (1.3).

To compare the performance of the weighted K-means clustering on the segmentation of customers, we also partition customers by the classical K-mean clustering. The numerical result is summarized in figures and tables. Figure (5.2) visualize the distribution of the population after the segmentation of customers. Table (5.8) shows the the level of differences between clusters.





a) Classical k-means Clustering (b Weighted K-means Clustering

Figure 5.2. Distribution of the Population

From Figure (5.2), it is easy to see that the distribution of the population provided by the weighted K-Means clustering is more natural than that obtained by the classical K-Means clustering.

On the other hand, we like to test whether clusters are well separated. The level of difference is measured by R^2 that represents the proportion of variance accounted for variables from clusters, and $R^2/(1 \quad R^2)$ that indicates the ratio of between-cluster variance to within-cluster variance.

Table (5.8) compares the difference of variables that mainly represent clusters for both algorithms. It shows that the difference between clusters provided by the weighted K-means clustering is more significant than what produced by the classical K-means clustering.

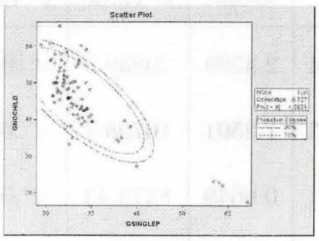
	Classical K-Means			**Weighted K-Means**			
Variable	R^2	$\frac{R^2}{(1-R^2)}$	F Value	R^2	$\frac{R^2}{(1-R^2)}$	F Value	P Value
GNOCHILD	0.4672	0.8767	7519	0.7084	2.4299	20839.9	$< .0001$
G1CHILD	0.4812	0.9275	7954.53	0.5573	1.2591	10798.4	$< .0001$
G2CHILD	0.3863	0.6295	5398.59	0.4911	0.9649	8275.43	$< .0001$
G3UCHILD	0.1258	0.1439	1234.49	0.2630	0.3568	3060.03	$< .0001$
GHHDS1PP	0.4635	0.864	7410.36	0.6353	1.7417	14937.8	$< .0001$
GHHDS2PP	0.5632	1.2896	11060.3	0.3215	0.4738	4063.14	$< .0001$
GHHDS3PP	0.4059	0.6831	5858.42	0.3641	0.5726	4910.85	$< .0001$
GHHDS45PP	0.5676	1.3124	11256.1	0.6940	2.2681	19452.0	$< .0001$
GHHDS6_PP	0.1615	0.1927	1652.38	0.2064	0.2600	2230.08	$< .0001$

Table 5.8: Difference Between Clusters

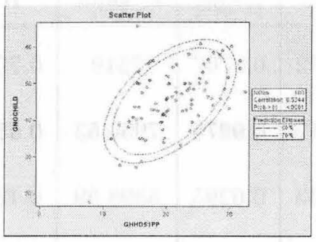
5.4 Consistency Testing

General speaking, validation of prototype-based clusters is approached by measuring the similarity within a cluster and the difference between clusters. As we see that the difference between clusters has been examined in the last subsection. In this subsection, we validate the similarity by testing the consistency of some variables that mainly represent clusters. The measurement is based on the background knowledge that has been known already. Figure (5.3) shows the test, where the level of the confidence limit is set as $70\% \sim 80\%$ with

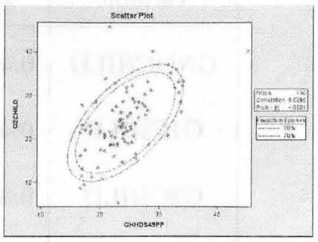
p-value less than 0.0001



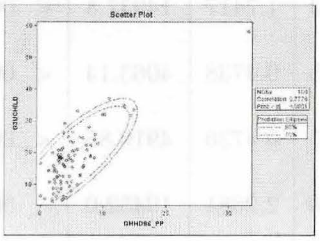
a)



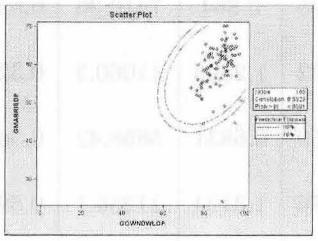
(b



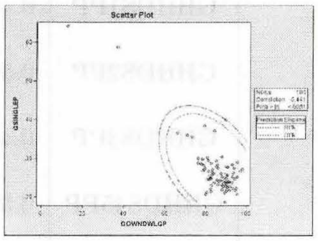
c



d)



e



(f)

Figure 5.3: Correlation of Children Variables

Now we give some explanations about what Figure(5.3) visualizes. The first four sub-figures discover the consistency of the directed children variables. Figure(a) shows the the negative correlation between the variable of single family and the variable of no children. Figure(b) gives the positive correlation between the variable of one household and the variable of no children. Figure(c) indicates that the variable of 4 or 5 households is strongly positive correlated to the variable of 2 children, the same kind correlation is verified by Figure(d) for the variable of 6 or more households and the variable of 3 or more children. Furthermore, Figure(e) shows that the married couples tend to own their house, Figure(d) indicates that the single families have less possibilities to buy houses.

Profile of Clusters

In order to better understand the segmentation of customers, the interpretation of clusters is needed. In the project, the interpretation is achieved by building a decision tree, in which the cluster label is set as the target variable, rules derived from the decision tree explain how to assign customers to the correct segmentation. The main characteristics of each cluster are summarized as follows.

T1: The families have no children, and likely to be single family.

T2: The families seem to have no or two children, more likely to have no children. If some families have children at home, their age is centered at 6 $\sim$14.

T3: The families have two children at home, their age is centered at 6 $\sim$ 14. The percentages for children with age 0 $\sim$ 5 and 18 $\sim$ 24 are very low.

T4: The families may have two children at home, but very likely to have no or one child. Children age is centered at $0 \sim 14$.

T5: The families may have one or two children at home, but more likely to have no child. If there are children at home, their age is centered at $6 \sim 14$.

T6: The families may have no or one child at home, but more likely to have two children. Their age is centered at $6 \sim 14$. The percentage for children with age $15 \sim 24$ and having 3 children or more is higher than average. Moreover, the percentage for children with age $18 \sim 24$ is higher than the average.

T7: The families may have one child at home, but more likely to have no or two children. Children age is centered at $0 \sim 14$. The percentage for children with age $18 \sim 24$ is around the average.

T8: The families tend to one or two children at home, but very likely to have one children.

5.5 Establish of Models

Each post code contains an average of fourteen households. If we assign the label of the segmentation of customers as the target value into the transaction data to form a new data set, the interrelationship between customers' buying behavior and their children composition could be discovered by applying a classification method that try to classify a set of instances or called objects into predefined classes. It is also known as supervised learning since the class label of each instance is known in advance.

The predictive pattern could be extracted from the data set in two steps. In the first step, the data set is partitioned into three components, the training set, the test set, and the evaluation set. These components are totally separated, there are no any objects in common. Typically, the training set is used to learn classification pattern which in turn to predict the future data. The testing set is used to refine the classification scheme, the refinement ensures that the classification model is more general and work well on unseen data. In the second step, the predictive model is estimated by the evaluation set.

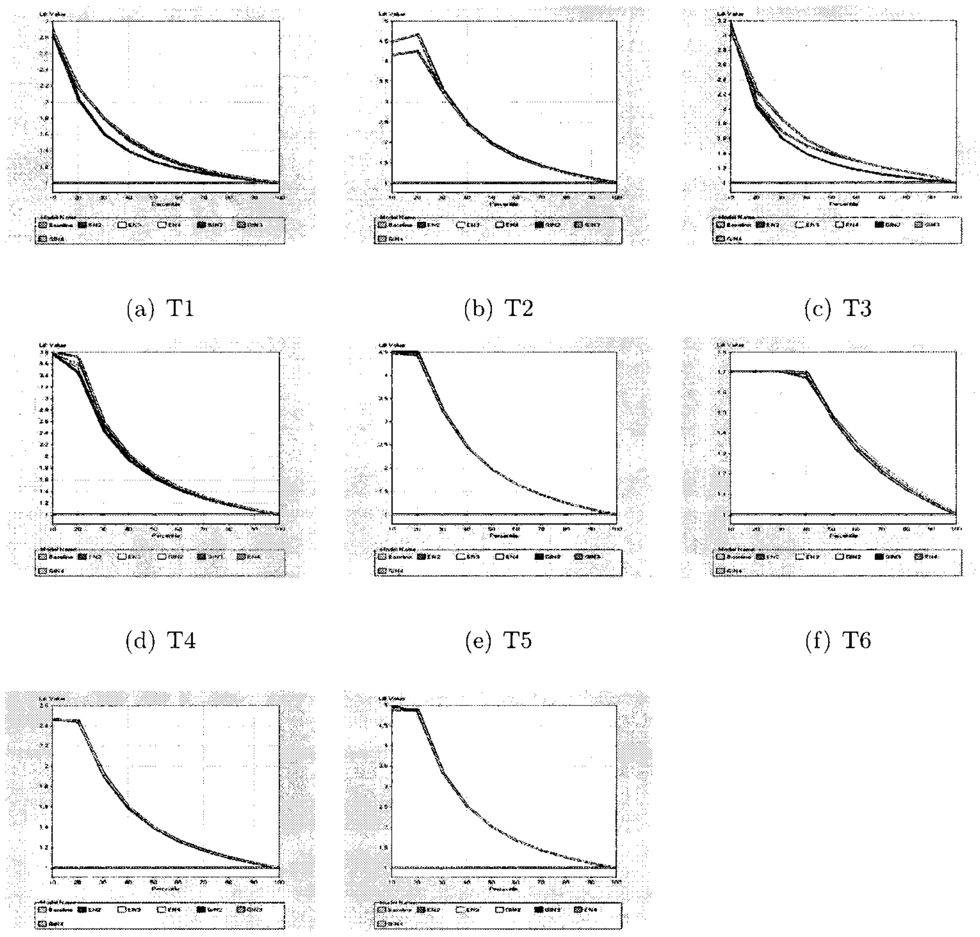
5.5.1 Model Voting

Since there are eight possibilities for values of the target, it is hard to distinguish between all the different types of output in one single model. Model voting provides the solution. The approach is to build a separate model indicating the propensity of customers' shopping behaviors according to their children information. There are many ways to combine the results. The simplest way is to choose the classification rule with the highest propensity if more than one rules are agreed at the same time. In addition, model voting simplifies a matter of adding or removing one of the propensity models.

Decision tree is adopted to build the separate model for each cluster. It is a good choice when the data mining task is classification of records or prediction of a outcome[30]. Moreover, it is primitively motivated by that the variables to describe categories of items are nominal. We built six different decision trees for each model, the decision tree is varied by the splitting measurement, Entropy[36] and Gini[30], each measurement allows 2-, 3-, or 4-way separation. After the evaluation of models, the one with the best performance is selected.

5.5.2 Estimation of Performance

The lift chart provides important insights into the models, as well as measuring the performance of models, in which, the horizontal axis represents the percentage of the data set, usually it is divided into 10 bins. The vertical axis gives the corresponding lift value, if there is no model built for the prediction, the lift value is 1, denoted as the base line. Figure(5.4) shows the lift chart for each model.



(a) T1 (b) T2 (c) T3

(d) T4 (e) T5 (f) T6

(g) T7 (h) T8

Figure 5.4: Lift Chart of Models

From Figure(5.4), we take the first two models as examples to see how to examine the performance. T1 model indicates that the top 10% of the video customers are 3.2 times more likely to respond than the overall response, the performance provided all six decision is not very good, GIN4(Gini measurement with 4-way slplitting) seems to have more consistent performance than other decision trees. T2 model shows that the top 10% of the video customers are 4.7 times more likely to respond than the overall response, all six decision trees performed very well, EN2(Entropy measurement with 2-way splitting) provides the best performance among them. By following similar analysis of other lift charts, it is easy to see that all models could provide the reasonably good performance.

5.5.3 Scoring Models

Finally, the model has to be scored. This actually introduces time-dependency into the modeling process because the score data set is more recent than the model set. The scoring data set is formed by the transaction data of 2005 from February to April, which contains 939894 records wrapped into 165894 customers. In fact, the model is going to be shaped by shifting one month repeatedly over the retention period. In such cases, measuring the model's accuracy is important in order to know when the model has to be rebuilt.

During the scoring process, the cutoff value for the probability to classify objects is set as 75%. Table (5.9) gives the numerical result of the scoring data set.

Cluster	Actual Rate(%)	Predictive Rate(%)
T1	9.28	5.00
T2	13.23	10.88
T3	3.70	1.64
T4	12.16	10.09
T5	13.97	13.08
T6	15.61	11.86
T7	10.66	7.45
T8	21.40	15.47

Table 5.9: Performance of the Scoring Data Set

From Table (5.9), we conclude that the numerical result obtained from the scoring data set matches the performance evaluated from the input data set. It help us verify that the model is able to make the prediction of the children composition of customers according to their buying behaviors.

Chapter 6

Conclusions and Future Work

In this thesis, we reformulated weighted K-means clustering via semidefinite programming where the eigenvalues of involved assignment matrix is either 0 or 1, so called 0-1 SDP. We also show how the unified framework can be employed in k-way normalized cut problem. An approximation method based on the singular value decomposition of the coefficient matrix in the projection space has been proposed to attack the underlying 0-1 SDP model. It is shown that the method can provide 2-approximation to the original problem. Moreover, we present a divisive hierarchical algorithm of clustering for $k \geq 3$, in which each binary separation is solved by the refined weighted K-means method in one dimensional space. Preliminary experiments illustrate that our new algorithms not only enjoys theoretical efficiency, but also performs well in practice.

In the case study, we have accomplished a project sponsored by *MITACS* and *Rogers Communication Inc.*. The objective of the project is to build a model to predict the children information of customers based on their buying behaviors. During the process of the model building, clustering analysis was applied as the first step to group customers with similar children information, and then the link between the segmentation of customers and their shopping behaviors was discovered.

There are several open questions regarding the new 0-1 SDP model. First, we note that there are several different ways to relax the 0-1 SDP model that have not been investigated. For example, we can solve the relaxed model (3.1) to find an approximation solution, which will give us a tighter bound that the relaxed based on SVD. However, it is unclear how to design a rounding procedure to extract a feasible solution and how to estimate the quality of the extracted solution. Secondly, the new approximation method require to solve the subproblems exactly, which turns out that it is still a challenge to estimate the quality of the solution when $k \geq 3$. More study is necessary to address these questions.

```
=================================================
```

Bibliography

[1] Peng, J. and Chen, H. (2006) 0-1 Semidefinite Programming for Graph-Cut Clustering: Modelling and Approximation. Advanced optimization Lab, Department of Computing and Software, McMaster University.

[2] Peng, J. and Xia, Y.(2004) A new theoretical framework for K-means-type clustering. To appear in *Foundation and recent advances in data mining*, Eds Chu and Lin, Springer Verlag.

[3] Berman, A. and Plemmins, RlJ. (1994). *nonnegative matrices in the mathematical science*, SIAM Classics in Applied Mathematics, Philadelphia.

[4] Sokal, R.R. and Michener, C.D. A Statistical Method for Evaluating Systematic Relationships. *The University of Kansas Scientific Bulletin* 38: 1409C1438, 1958.

[13] Xing, E.P. and Jordan, M.I. (2003) On semidefinite relaxation for nor-

malized k-cut and connections to spectral clustering. *Tech Report CSD-03-1265*, UC Berkeley.

[6] Peng, J. and Wei, Y. (2005). Approximating K-means-type clustering via semidefinite programming. Advanced optimization Lab, Department of Computing and Software, McMaster University.

[7] Cooper, M.C. and Milligan, G.W. The Effect of Error on Determining the Number of Clusters, *Proceedings of the International Workshop on Data Analysis*, 319-328, 1988.

[8] Shi, J. and Malik, J.(2000) Normalized cuts and image segmentation. *IEEE. Trans. on Pattern Analysis and Machine Intelligence*, 22, 888-905.

[9] Gulub, G. and Loan, C. V. (1996) *Matix Computation.* John Hopkins University Press.

[10] Sanzheng Qiao, Orthogonalization Techniques for the Lanczos Tridiagonalization of Complex Symmetric Matrices, *Advanced Signal Processing Algorithms, Architectures, and Implementations XIV*, 5529, 423-434, 2004.

[11] Jain, A.K., Dubes, R.C.(1988). *Algorithms for clustering data.* Englewood Cliffs, NJ:Prentice Hall.

[12] Jain, A.K., Murty, N.M. and Flynn, P.J.(1999) Data Clustering: A Review. *ACM Computing Surveys*, Vol.31 No.3, pp 264-323.

[13] Xing, E.P. and Jordan, M.I. (2003) On semidefinite relaxation for normalized k-cut and connections to spectral clustering. *Tech Report CSD-03-1265*, UC Berkeley.

[14] Dhillon, I., Guan, Y. and Kulis, B.(2004) A Unified View of Kernel k-means, Spectral Clustering and Graph Cuts. *UTCS Technical Report TR-04-25.*

[15] Gu, M., Zha, H., Ding, C., He, X. and Simon, H. (2001) Spectral relaxation models and structure analysis for k-way graph Clustering and bi-clustering. *Penn State Univ Tech Report.*

[16] Jiawei Han, Micheline Kamber. (2001) *Data Mining Concepts and Techniques*, Academic Press.

[17] Meila, M. and Shi, J. (2001) A random walks view of spectral segmentation. *Int'l Workshop on AI & Sta.*

[18] Ng, A.Y., Jordan, M.I. and Weiss, Y. (2001). On spectral clustering: Analysis and an algorithm. *Proc. Neural Info. Processing Systems, NIPS*, 14.

[19] Pang, N., Michael, S. and Vipin K.(2006) *Introduction to Data Mining*, Pearson Education Inc.

[20] Heyer, L.J., Kruglyak, S. and Yooseph, S. Exploring Expression Data: Identification and Analysis of Coexpressed Genes, Genome Research, 1106-1115.

[21] Lu, S.Y. and Fu, K.S.(1978) A sentence to sentence clustering procedure for pattern analysis. *IEEE transactions on Systems Mans and Cybernetics*, 8:381-389.

[22] Ding, C. and He, X. (2004) K-means clustering via principal component analysis. *Proceedings of the 21st International Conference on Machine Learning*, Banff, Canada.

[23] Bagirov, A.M., Rubinov, A.M., Soukhoroukova, N. V. and Yearwood, J.L. (2003). Unsupervised and supervised data classification via nonsmooth and global optimization. *Top*, 11, no.1, pp 1-93.

[24] Erminio, D. and Guerrisi, F. (2002) A Fuzzy Clustering Algorithm Based on the Analogy with Mechanical Physics, *Quality and Quantity*, vol. 36, pp. 239-257.

[25] Weiss, Y. (1999). Segmentation using eigenvectors: a unifying view. *Proceedings IEEE International Conference on Computer Vision*, 975-982.

[26] Jolliffe, I. (2002) *Principal component analysis*. Springer, 2nd edition.

[27] Hansen, P., Jaumard, B. and Mladenovic, N.(1998). Minumum sum of squares clustering in a low dimensional space. *J. Classification*, 15, 37-55.

[28] Zha, H., Ding, C., Gu, M., He, X. and Simon, H. (2002) Spectral Relaxation for K-means Clustering. In Dietterich, T., Becker, S. and Ghahramani, Z. Eds., *Advances in Neural Information Processing Systems 14*, pp. 1057-1064. MIT Press.

[29] Inaba, M., Katoh, N. and Imai, H.(1994) Applications of Weighted Voronoi Diagrams and Randomization to Variance-Based k-clustering. *Proceeding of 10th ACM Symposium on Computational Geometry*, pp 332-339.

[30] Safavian S. R. and David LandgrebeA (1991), Survey of Decision Tree Classifier Methodology, *IEEE Transactions on Systems*, vol. 21, pp. 660-674.

[31] Michalski, R.S. and Chilausky, R.L. (1980). Learning by being told and learning from examples: An experimental comparison of the two methods

of knowledge acquisition in the context of developing an expert system for soybean disease diagnosis. *International Journal of Policy Analysis and Information Systems*, 4(2), pp. 125-161.

[32] Drineas, P., Frieze, A., Kannan, R., Vempala, R. and Vinay, V. (2004). Clustering large graphs via singular value decomposition. *Machine Learning*, 56, 9-33.

[33] Späth, H. (1980) *Cluster analysis Algorithms for Data Reduction and Classification of Objects*, John Wiley and Sons, Ellis Horwood Ltd.

[34] Ruspini, E. H. (1970). Numerical methods for fuzzy clustering, *inform. Sci.*, 2, 319-350.

[35] Kaufman, L. and Peter Rousseeuw, P. (1990). Finding Groups in Data, *An Introduction to Cluster Analysis,* John Wiley.

[36] Hai Leong Chieu and Hwee Tou Ng (2002). Named Entity Recognition: A Maximum Entropy Approach Using Global Information. *Proc.: 17th Int'l Conf. Computation Linguistics (COLING 2002)*, Taipei, Taiwan.

[37] Ghosh J. (2003). Scalable Clustering. In N. Ye, Editor, The Handbook of Data Mining, Lawrence Erlbaum Associate, Inc, pp. 247-277.

[38] R. Cooley, B. Mobasher, and J. Srivastava (1997). Web Mining: Information and Pattern Discovery on the World Wide Web. Technical Report TR97-027, University of Minnesota.