ENHANCED DATA MINING SUITE USING SIGNAL PROCESSING

Project Report submitted in partial fulfillment of the requirement for the degree of

Bachelor of Technology.

in

Information Technology

under the Supervision of

Mr. Pardeep Kumar

By

SHIVAM FUTELA - 091412

SANDEEP KUMAR - 091437

TSHERING WANGCHUK – 091411

to

Jaypee University of Information and Technology

Waknaghat, Solan – 173234, Himachal Pradesh

Table of Contents

List of Figures

Certificate

This is to certify that project report entitled "Enhanced Data Mining Suite Using Signal Processing", submitted by Shivam Futela(091412), Sandeep Kumar(091437), Tshering Wangchuk (091411) in partial fulfillment for the award of degree of Bachelor of Technology in Information Technology Engineering to Jaypee University of Information Technology, Waknaghat, Solan has been carried out under my supervision.

This work has not been submitted partially or fully to any other University or Institute for the award of this or any other degree or diploma.

Date: **Supervisor's Name**

Designation

Acknowledgement

We would like to take this oppurtinity to express our sincere indebtness and sense of gratitude to all those who have contributed greatly towards the successful partial completion of our project "Enhanced Data Mining Suite Using Signal Processing" .

It would not have been possible to see through the undertaken project without the guidance and constant support of our guide Mr. Pardeep Kumar . For his coherent guidance we feel fortunate to be taught by him, who gave us his unwavering support. We owe our heartiest thanks to Brig. (Retd.) S.P.Ghrera(H.O.D.-CSE/IT Department) who've always inspired confidence in us to take initiative.

As a final note, we are grateful to CSE and IT Department of Jaypee University of Information and Technology ,who inspired us to undertake difficult tasks by their strength of understanding our calibre and our requirements and taught us to work with patience and provided constant encouragement to successfully complete the project.

Date:

Shivam Futela(091412)
SandeepKumar(091437)
Tshering Wangchuk(091411)

ABSTRACT

Data clustering is an unsupervised data analysis and data mining technique,which offers refined and more abstract views to the inherent structure of a data set by partitioning it into a number of disjoint or overlapping (fuzzy) groups. Hundreds of clustering algorithms and Signal Processing Techniques have been developed by researchers from a number of different scientific disciplines. The intention of this report is to present a special class of clustering algorithms with or without signal processing, namely partition-based methods. After the introduction and a review on iterative relocation clustering algorithms in Data Mining and Signal Processing, some illustrative results are presented.

CHAPTER 1

INTRODUCTION

PRE-PROCESSING

DATA MINING

ANOMALY DETECTION

ASSOCIATION RULE LEARNING

CLUSTER ANALYSIS

MATLAB CODE

APPLICATIONS

DATA MINING

Data mining (the analysis step of the "Knowledge Discovery in Databases" process, or KDD) is the process that attempts to discover patterns in large data sets. It utilizes methods at the intersection of artificial intelligence, machine learning, statistics, and database systems. The overall goal of the data mining process is to extract information from a data set and transform it into an understandable structure for further use.

The Knowledge Discovery in Databases (KDD) process is commonly defined with the stages:

I. Selection
II. Pre-processing
III. Transformation
IV. *Data Mining*
V. Interpretation/Evaluation.

1.1 Pre-processing

Before data mining algorithms can be used, a target data set must be assembled. As data mining can only uncover patterns actually present in the data, the target dataset must be large enough to contain these patterns while remaining concise enough to be mined within an acceptable time limit. A common source for data is a data mart or data warehouse. Pre-processing is essential to analyze the multivariate datasets before data mining. The target set is then cleaned. Data cleaning removes the observations containing noise and those with missing data.

1.2 Data mining

Data mining involves six common classes of tasks:

I. Anomaly detection (Outlier/change/deviation detection) – The identification of unusual data records, that might be interesting or data errors that require further investigation.
II. Association rule learning (Dependency modeling) – Searches for relationships between variables. For example a supermarket might gather data on customer purchasing habits. Using association rule learning, the supermarket can determine which products are

frequently bought together and use this information for marketing purposes. This is sometimes referred to as market basket analysis.

III. Clustering – is the task of discovering groups and structures in the data that are in some way or another "similar", without using known structures in the data.

IV. Classification – is the task of generalizing known structure to apply to new data. For example, an e-mail program might attempt to classify an e-mail as "legitimate" or as "spam".

V. Regression – Attempts to find a function which models the data with the least error.

VI. Summarization – providing a more compact representation of the data set, including visualization and report generation.

VII. Sequential pattern mining – Sequential pattern mining finds sets of data items that occur together frequently in some sequences. Sequential pattern mining, which extracts frequent subsequences from a sequence database, has attracted a great deal of interest during the recent data mining research because it is the basis of many applications, such as: web user analysis, stock trend prediction, DNA sequence analysis, finding language or linguistic patterns from natural language texts, and using the history of symptoms to predict certain kind of disease.

1.3Anomaly detection

Anomaly detection, also referred to as outlier detection refers to detecting patterns in a given data set that do not conform to an established normal behavior. The patterns thus detected are called anomalies and often translate to critical and actionable information in several application domains. Anomalies are also referred to as **outliers**, change, deviation, surprise, aberrant, peculiarity, intrusion, etc.

In particular in the context of abuse and network **intrusion detection**, the interesting objects are often not *rare* objects, but unexpected bursts in activity. This pattern does not adhere to the common statistical definition of an **outlier** as a rare object, and many outlier detection methods (in particular unsupervised methods) will fail on such data, unless it has been aggregated appropriately. Instead, a **cluster analysis** algorithm may be able to detect the micro clusters formed by these patterns.

1.3.1 Popular techniques for anomaly detection:

Several anomaly detection techniques have been proposed in literature. Some of the popular techniques are:

I. Distance based techniques (k-nearest neighbor, Local Outlier Factor).

II. One Class Support Vector Machines.

III. Replicator Neural Networks.

IV. Cluster analysis based outlier detection.

V. Pointing at records that deviate from learned association rules.

1.4 Association rule learning:

In data mining, association rule learning is a popular and well researched method for discovering interesting relations between variables in large databases. It is intended to identify strong rules discovered in databases using different measures of interestingness.

For example, the rule $\{\mathrm{onions}, \mathrm{potatoes}\} \Rightarrow \{\mathrm{burger}\}$ found in the sales data of a supermarket would indicate that if a customer buys onions and potatoes together, he or she is likely to also buy hamburger meat. Such information can be used as the basis for decisions about marketing activities such as, e.g., promotional pricing or product placements. In addition to the above example from market basket analysis association rules are employed today in many application areas including Web usage mining, intrusion detection, Continuous production and bioinformatics. As opposed to sequence mining, association rule learning typically does not consider the order of items either within a transaction or across transactions.

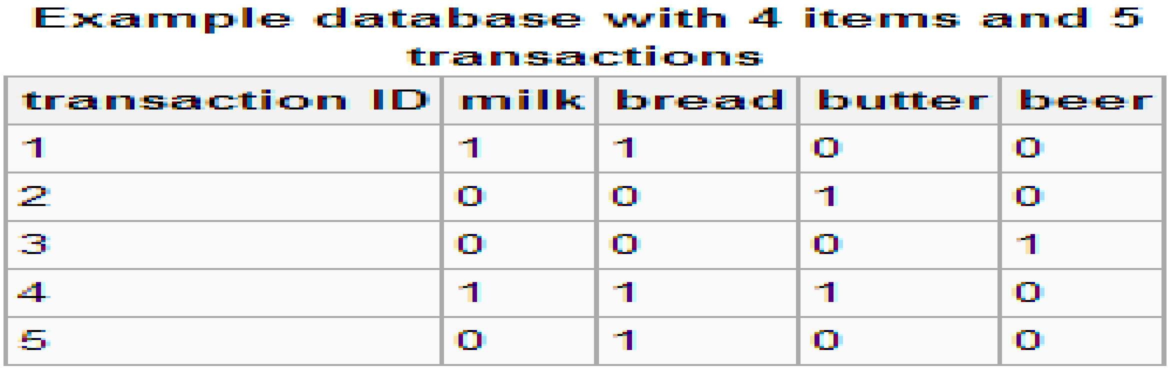
Definition:

The problem of association rule mining is defined as: Let $I = \{i_1, i_2, \ldots, i_n\}$be a set of n binary attributes called *items*. Let $D = \{t_1, t_2, \ldots, t_m\}$ be a set of transactions called the *database*. Each transaction in D has a unique transaction ID and contains a subset of the items in I. A *rule* is defined as an implication of the form $X \Rightarrow Y$ where $X, Y \subseteq I$

and $X \cap Y = \emptyset$. The sets of items (for short *itemsets*) X and Y are called *antecedent* (left-hand-side or LHS) and *consequent* (right-hand-side or RHS) of the rule respectively.

To illustrate the concepts, we use a small example from the supermarket domain. The set of items is $I = \{\mathrm{milk}, \mathrm{bread}, \mathrm{butter}, \mathrm{beer}\}$ and a small database containing the items (1 codes presence and 0 absence of an item in a transaction) is shown in the table to the right. An example rule for the supermarket could be $\{\mathrm{butter}, \mathrm{bread}\} \Rightarrow \{\mathrm{milk}\}$meaning that if butter and bread are bought, customers also buy milk.

Note: this example is extremely small. In practical applications, a rule needs a support of several hundred transactions before it can be considered statistically significant, and datasets often contain thousands or millions of transactions



Example database with 4 items and 5 transactions

transaction ID	milk	bread	butter	beer
1	1	1	0	0
2	0	0	1	0
3	0	0	0	1
4	1	1	1	0
5	0	1	0	0

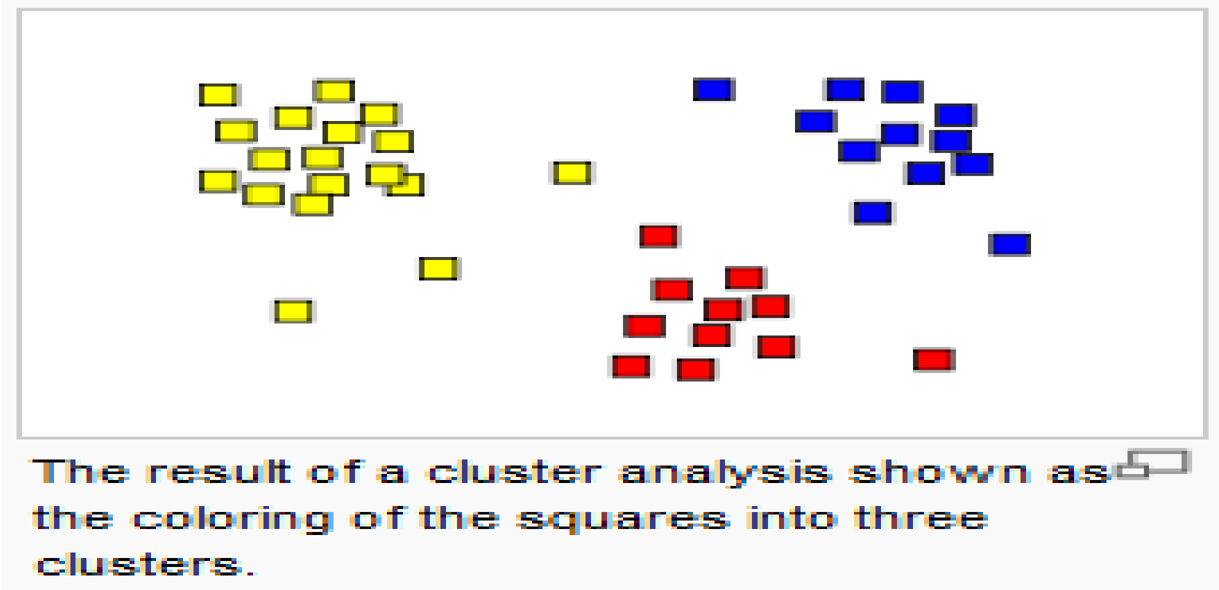
1.5 Cluster Analysis:

Cluster analysis or clustering is the task of assigning a set of objects into groups (called clusters) so that the objects in the same cluster are more similar (in some sense or another) to each other than to those in other clusters.

Clustering is a main task of explorative data mining, and a common technique for statistical data analysis used in many fields, including machine learning, pattern recognition, image analysis, information retrieval, and bioinformatics.

Cluster analysis itself is not one specific algorithm, but the general task to be solved. It can be achieved by various algorithms that differ significantly in their notion of what constitutes a cluster and how to efficiently find them. Popular notions of clusters include groups with low distances among the cluster members, dense areas of the data space, intervals or particular statistical distributions. Clustering can therefore be formulated as a multi-objective optimization problem. The appropriate clustering algorithm and parameter settings (including values such as the distance function to use, a density threshold or the number of expected clusters) depend on the individual data set and intended use of the results. Cluster analysis as such is not an automatic task, but an iterative process of knowledge discovery or interactive multi-objective optimization that involves trial and failure. It will often be necessary to modify preprocessing and parameters until the result achieves the desired properties.

Besides the term *clustering*, there are a number of terms with similar meanings, including *automatic classification*, *numerical taxonomy*, *botryology* (from Greek βότρυς "grape") and *typological analysis*. The subtle differences are often in the usage of the results: while in data mining, the resulting groups are the matter of interest, in automatic classification primarily their discriminative power is of interest. This often leads to misunderstandings between researchers coming from the fields of data mining and machine learning, since they use the same terms and often the same algorithms, but have different goals



The result of a cluster analysis shown as the coloring of the squares into three clusters.

Figure 1 Result of cluster analysis

1.5.1 Clusters and clusterings :

The notion of a "cluster" varies between algorithms and is one of the many decisions to take when choosing the appropriate algorithm for a particular problem. At first the terminology of a cluster seems obvious: a group of data objects. However, the clusters found by different algorithms vary significantly in their properties, and understanding these "cluster models" is key to understanding the differences between the various algorithms. Typical cluster models include:

I. Connectivity models: for example hierarchical clustering builds models based on distance connectivity.
II. Centroid models: for example the k-means algorithm represents each cluster by a single mean vector.
III. Distribution models: clusters are modeled using statistic distributions, such as multivariate normal distributions used by the Expectation-maximization algorithm.
IV. Density models: for example DBSCAN and OPTICS defines clusters as connected dense regions in the data space.
V. Subspace models: in Bi clustering (also known as Co-clustering or two-mode-clustering), clusters are modeled with both cluster members and relevant attributes.
VI. Group models: some algorithms (unfortunately) do not provide a refined model for their results and just provide the grouping information.
VII. Graph-based models: a clique, i.e., a subset of nodes in a graph such that every two nodes in the subset are connected by an edge can be considered as a prototypical form of cluster. Relaxations of the complete connectivity requirement (a fraction of the edges can be missing) are known as quasi-cliques.

A "clustering" is essentially a set of such clusters, usually containing all objects in the data set. Additionally, it may specify the relationship of the clusters to each other, for example a hierarchy of clusters embedded in each other. Clusterings can be roughly distinguished in:

I. hard clustering: each object belongs to a cluster or not
II. soft clustering (also: fuzzy clustering): each object belongs to each cluster to a certain degree (e.g. a likelihood of belonging to the cluster)

There are also finer distinctions possible, for example:

I. strict partitioning clustering: here each object belongs to exactly one cluster

II. strict partitioning clustering with outliers: objects can also belong to no cluster, and are considered outliers.

III. overlapping clustering (also: alternative clustering, multi-view clustering): while usually a hard clustering, objects may belong to more than one cluster.

IV. hierarchical clustering: objects that belong to a child cluster also belong to the parent cluster

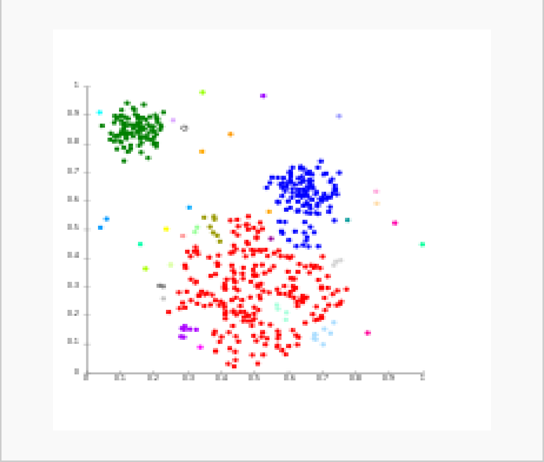
V. subspace clustering: while an overlapping clustering, within a uniquely defined subspace, clusters are not expected to overlap.

1.5.2 Connectivity based clustering (hierarchical clustering):

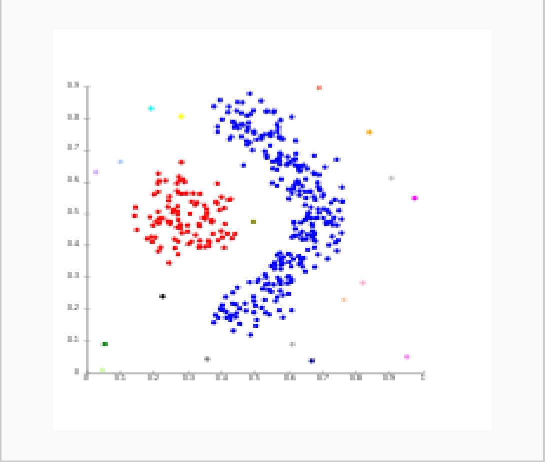
Connectivity based clustering, also known as *hierarchical clustering*, is based on the core idea of objects being more related to nearby objects than to objects farther away. As such, these algorithms connect "objects" to form "clusters" based on their distance. A cluster can be described largely by the maximum distance needed to connect parts of the cluster. At different distances, different clusters will form, which can be represented using a dendrogram, which explains where the common name "hierarchical clustering" comes from: these algorithms do not provide a single partitioning of the data set, but instead provide an extensive hierarchy of clusters that merge with each other at certain distances. In a dendrogram, the y-axis marks the distance at which the clusters merge, while the objects are placed along the x-axis such that the clusters don't mix.

Connectivity based clustering is a whole family of methods that differ by the way distances are computed. Apart from the usual choice of distance functions, the user also needs to decide on the linkage criterion (since a cluster consists of multiple objects, there are multiple candidates to compute the distance to) to use. Popular choices are known as single-linkage clustering (the minimum of object distances), complete linkage clustering (the maximum of object distances) or UPGMA ("Unweighted Pair Group Method with Arithmetic Mean", also known as average linkage clustering). Furthermore, hierarchical clustering can be agglomerative (starting with single elements and aggregating them into clusters) or divisive (starting with the complete data set and dividing it into partitions).

While these methods are fairly easy to understand, the results are not always easy to use, as they will not produce a unique partitioning of the data set, but a hierarchy the user still needs to choose appropriate clusters from. The methods are not very robust towards outliers, which will either show up as additional clusters or even cause other clusters to merge (known as "chaining phenomenon", in particular with single-linkage clustering). In the general case, the complexity is $\mathcal{O}(n^3)$, which makes them too slow for large data sets. For some special cases, optimal efficient methods (of complexity $\mathcal{O}(n^2)$) are known: SLINK for single-linkage and CLINK for complete-linkage clustering. In the data mining community these methods are recognized as a theoretical foundation of cluster analysis, but often considered obsolete. They did however provide inspiration for many later methods such as density based clustering



Single-linkage on Gaussian data. At 35 clusters, the biggest cluster starts fragmenting into smaller parts, while before it was still connected to the second largest due to the single-link effect.



Single-linkage on density-based clusters. 20 clusters extracted, most of which contain single elements, since linkage clustering does not have a notion of "noise".

H

***k*-means clustering**

In data mining, *k*-means clustering is a method of cluster analysis which aims to partition *n* observations into *k* clusters in which each observation belongs to the cluster with the nearest mean. This results in a partitioning of the data space into Voronoi cells.

The problem is computationally difficult (NP-hard), however there are efficient heuristic algorithms that are commonly employed and converge fast to a local optimum. These are usually similar to the expectation-maximization algorithm for mixtures of Gaussian distributions via an iterative refinement approach employed by both algorithms. Additionally, they both use cluster centers to model the data, however *k*-means clustering tends to find clusters of comparable spatial extent, while the expectation-maximization mechanism allows clusters to have different shapes

Description

Given a set of observations (x_1, x_2, …, x_n), where each observation is a *d*-dimensional real vector, *k*-means clustering aims to partition the *n* observations into *k* sets ($k \leq n$)
S = {S_1, S_2, …, S_k} so as to minimize the within-cluster sum of squares (WCSS):

$$\underset{\mathbf{S}}{\arg\min} \sum_{i=1}^{k} \sum_{\mathbf{x}_j \in S_i} \left\| \mathbf{x}_j - \boldsymbol{\mu}_i \right\|^2$$

History

The term "*k*-means" was first used by James MacQueen in 1967, though the idea goes back to Hugo Steinhaus in 1957. The standard algorithm was first proposed by Stuart Lloyd in 1957 as a technique for pulse-code modulation, though it wasn't published until 1982.

Standard algorithm

The most common algorithm uses an iterative refinement technique. Due to its ubiquity it is often called the *k*-means algorithm; it is also referred to as Lloyd's algorithm, particularly in the computer science community.

Given an initial set of *k* means $m_1^{(1)}$,…,$m_k^{(1)}$ (see below), the algorithm proceeds by alternating between two steps:[4]

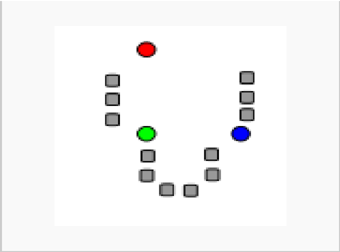
Assignment step: Assign each observation to the cluster with the closest mean (i.e. partition the observations according to the Voronoi diagram generated by the means).

$$S_i^{(t)} = \{x_p : \|x_p - m_i^{(t)}\| \le \|x_p - m_j^{(t)}\| \ \forall\ 1 \le j \le k\}$$

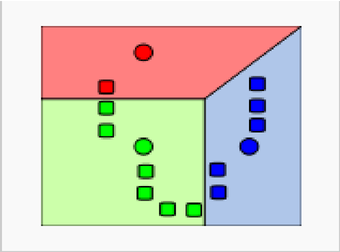
Update step: Calculate the new means to be the centroid of the observations in the cluster

$$\mathbf{m}_i^{(t+1)} = \frac{1}{|S_i^{(t)}|}\sum_{\mathbf{x}_j \in S_i^{(t)}} \mathbf{x}_j$$

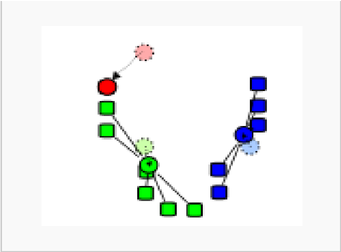
The algorithm is deemed to have converged when the assignments no longer change.



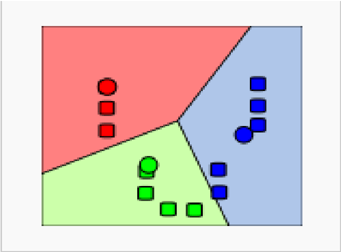
1) *k* initial "means" (in this case *k*=3) are randomly generated within the data domain (shown in color).



2) *k* clusters are created by associating every observation with the nearest mean. The partitions here represent the Voronoi diagram generated by the means.



3) The centroid of each of the *k* clusters becomes the new mean.



4) Steps 2 and 3 are repeated until convergence has been reached.

As it is a heuristic algorithm, there is no guarantee that it will converge to the global optimum, and the result may depend on the initial clusters. As the algorithm is usually very fast, it is common to run it multiple times with different starting conditions. However, in the worst case, *k*-means can be very slow to converge: in particular it has been shown that there exist certain point sets, even in 2 dimensions, on which *k*-means takes exponential time, that is $2^{\Omega(n)}$, to converge. These point sets do not seem to arise in practice: this is corroborated by the fact that the smoothed running time of *k*-means is polynomial.

The "assignment" step is also referred to as expectation step, the "update step" as maximization step, making this algorithm a variant of the *generalized* expectation-maximization algorithm.

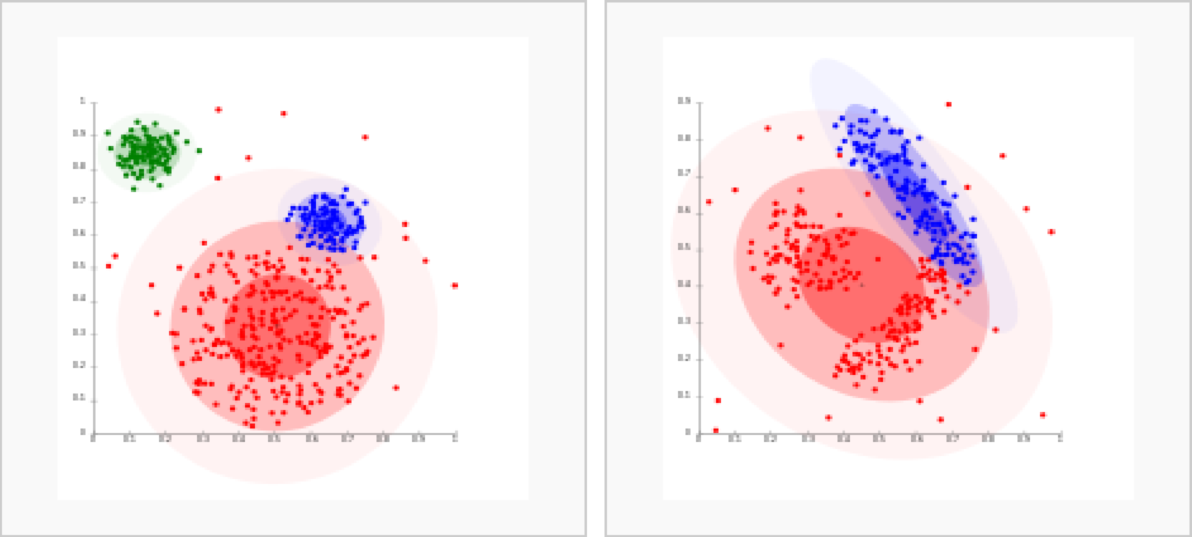
1.5.3 Distribution-based clustering

The clustering model most closely related to statistics is based on distribution models. Clusters can then easily be defined as objects belonging most likely to the same distribution. A nice property of this approach is that this closely resembles the way artificial data sets are generated: by sampling random objects from a distribution.

While the theoretical foundation of these methods is excellent, they suffer from one key problem known as over fitting, unless constraints are put on the model complexity. A more complex model will usually always be able to explain the data better, which makes choosing the appropriate model complexity inherently difficult.

The most prominent method is known as expectation-maximization algorithm (or short: *EM-clustering*). Here, the data set is usually modeled with a fixed (to avoid overfitting) number of Gaussian distributions that are initialized randomly and whose parameters are iteratively optimized to fit better to the data set. This will converge to a local optimum, so multiple runs may produce different results. In order to obtain a hard clustering, objects are often then assigned to the Gaussian distribution they most likely belong to, for soft clusterings this is not necessary.

Distribution-based clustering is a semantically strong method, as it not only provides you with clusters, but also produces complex models for the clusters that can also capture correlation and dependence of attributes. However, using these algorithms puts an extra burden on the user: to choose appropriate data models to optimize, and for many real data sets, there may be no mathematical model available the algorithm is able to optimize (e.g. assuming Gaussian distributions is a rather strong assumption on the data)



On Gaussian-distributed data, EM works well, since it uses Gaussians for modelling clusters

Density-based clusters cannot be modeled using Gaussian distributions

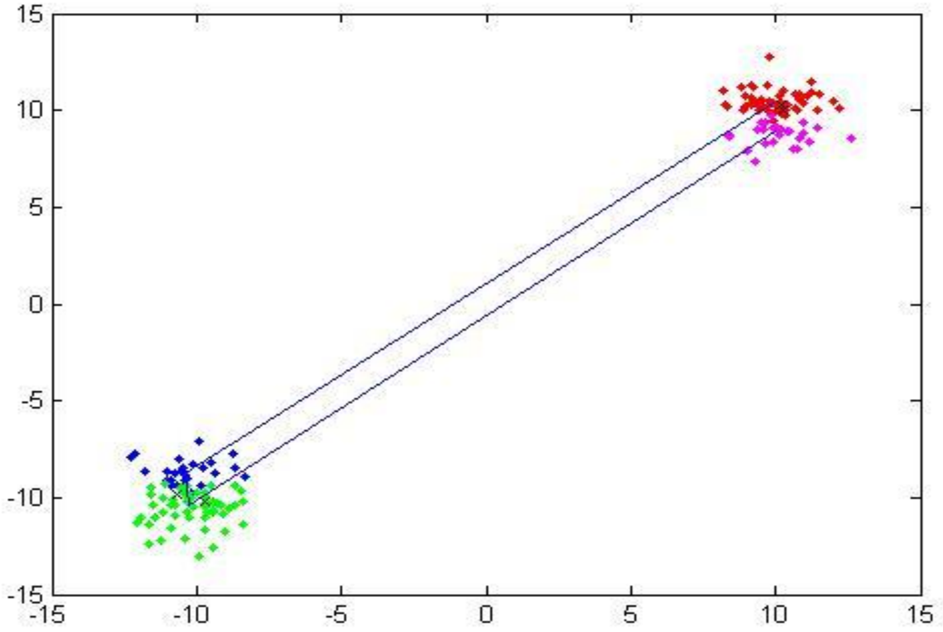
Figure 2 Gaussian Distribution

1.6MATLAB code for clustering on randomly generated data:

```
clc;
clear all;
close all;
% set-1
% Firstly defining  the data...+ones(80,4);-ones(80,4)
X = [randn(80,4); randn(80,4)];
% Now the kmeans is applied...
opts = statset('Display','final');
[cidx, ctrs] = kmeans(X, 4,'Distance','city', ...
                         'Replicates',5, 'Options',opts);
%% Now look at the cluster formation....
figure(1),plot(X(cidx==1,1),X(cidx==1,2),'r.', ...
X(cidx==2,1),X(cidx==2,2),'b.',...
X(cidx==3,1),X(cidx==3,2),'g.',...
X(cidx==4,1),X(cidx==4,2),'m.',...
ctrs(:,1),ctrs(:,2),ctrs(:,3),ctrs(:,4),'kx');
% this are the initial result for the implementation....
%%  set-2
% Firstly defining  the data...+ones(80,4);-ones(80,4)
X = [randn(80,4)+10*ones(80,4); randn(80,4)-10*ones(80,4)];
%X = [data2(:,1); data2(:,2)];
% Now the kmeans is applied...
opts = statset('Display','final');
[cidx, ctrs] = kmeans(X, 4,'Distance','city', ...
                         'Replicates',5, 'Options',opts);
```

```
%% Now look at the cluster formation....
figure(2),plot(X(cidx==1,1),X(cidx==1,2),'r.', ...
X(cidx==2,1),X(cidx==2,2),'b.',...
X(cidx==3,1),X(cidx==3,2),'g.',...
X(cidx==4,1),X(cidx==4,2),'m.',...
ctrs(:,1),ctrs(:,2),ctrs(:,3),ctrs(:,4),'kx');
% this are the initial result for the implementation....
%%  set-3
% Firstly defining  the data...+ones(80,4);-ones(80,4)
X = [randn(80,4)+5*ones(80,4); randn(80,4)-10*ones(80,4)];
%X = [data2(:,1); data2(:,2)];
% Now the kmeans is applied...
opts = statset('Display','final');
[cidx, ctrs] = kmeans(X, 4,'Distance','city', ...
                          'Replicates',5, 'Options',opts);
%% Now look at the cluster formation....
figure(3),plot(X(cidx==1,1),X(cidx==1,2),'r.', ...
X(cidx==2,1),X(cidx==2,2),'b.',...
X(cidx==3,1),X(cidx==3,2),'g.',...
X(cidx==4,1),X(cidx==4,2),'m.',...
ctrs(:,1),ctrs(:,2),ctrs(:,3),ctrs(:,4),'kx');
% this are the initial result for the implementation....
%%  set-4
% Firstly defining  the data...+ones(80,4);-ones(80,4)
X = [randn(80,4)+ones(80,4); randn(80,4)-ones(80,4)];
%X = [data2(:,1); data2(:,2)];
% Now the kmeans is applied...
opts = statset('Display','final');
[cidx, ctrs] = kmeans(X, 4,'Distance','city', ...
                          'Replicates',5, 'Options',opts);
%% Now look at the cluster formation....
```

```
figure(4),plot(X(cidx==1,1),X(cidx==1,2),'r.', ...
X(cidx==2,1),X(cidx==2,2),'b.',...
X(cidx==3,1),X(cidx==3,2),'g.',...
X(cidx==4,1),X(cidx==4,2),'m.',...
ctrs(:,1),ctrs(:,2),ctrs(:,3),ctrs(:,4),'kx');
% this are the initial result for the implementation....
```



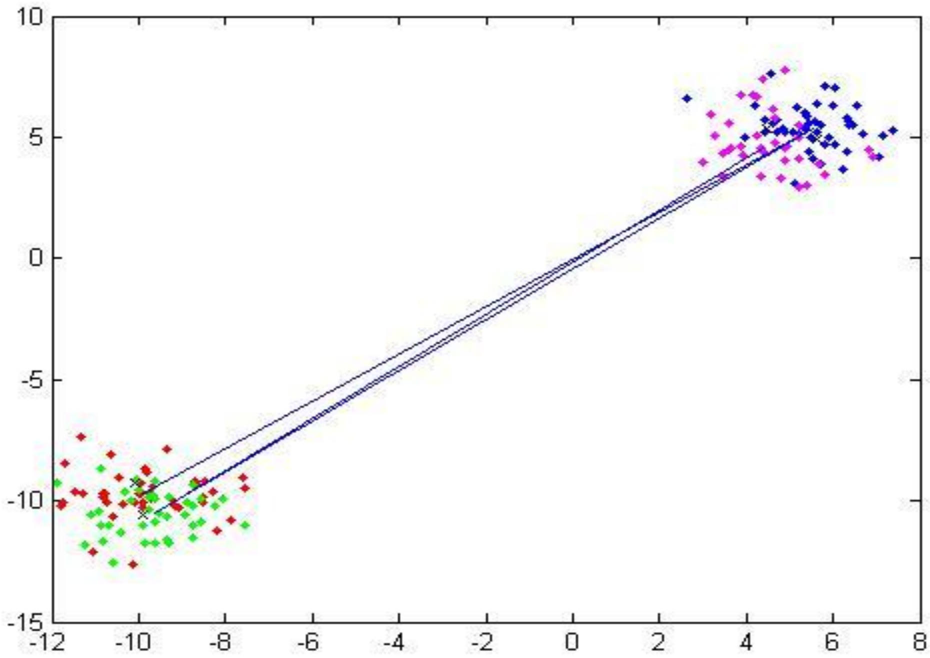


Figure 3 : Clustering on random data

1.7 Applications

I. Biology, computational biology and bioinformatics

a. Plant_and animalecology

b. cluster analysis is used to describe and to make spatial and temporal comparisons of communities (assemblages) of organisms in heterogeneous environments; it is also used in plant systematics to generate artificial phylogenies or clusters of organisms (individuals) at the species, genus or higher level that share a number of attributes

c. Transcriptomics

d. clustering is used to build groups of genes with related expression patterns (also known as co-expressed genes). Often such groups contain functionally related proteins, such as enzymes for a specific pathway, or genes that are co-regulated. High throughput experiments using expressed sequence tags (ESTs) or DNA microarrays can be a powerful tool for genome annotation, a general aspect of genomics.

e. Sequence analysis
f. clustering is used to group homologous sequences into gene families. This is a very important concept in bioinformatics, and evolutionary biology in general. See evolution by gene duplication.
g. High-throughput genotyping platforms
h. clustering algorithms are used to automatically assign genotypes.
i. Human genetic clustering
j. The similarity of genetic data is used in clustering to infer population structures.

II. Medicine
   a. Medical imaging
   b. On PET scans, cluster analysis can be used to differentiate between different types of tissue and blood in a three dimensional image. In this application, actual position does not matter, but the voxel intensity is considered as a vector, with a dimension for each image that was taken over time. This technique allows, for example, accurate measurement of the rate a radioactive tracer is delivered to the area of interest, without a separate sampling of arterial blood, an intrusive technique that is most common today.
   c. IMRT segmentation
   d. Clustering can be used to divide a fluence map into distinct regions for conversion into deliverable fields in MLC-based Radiation Therapy.

III. Business and marketing
   a. Market research
   b. Cluster analysis is widely used in market research when working with multivariate data from surveys and test panels. Market researchers use cluster analysis to partition the general population of consumers into market segments and to better understand the relationships between different groups of consumers/potential customers, and for use in market segmentation, Product positioning, New product development and Selecting test markets.
   c. Grouping of shopping items

d. Clustering can be used to group all the shopping items available on the web into a set of unique products. For example, all the items on eBay can be grouped into unique products.

IV. World wide web

a. Social network analysis

b. In the study of social networks, clustering may be used to recognize communities within large groups of people.

c. Search result grouping

d. In the process of intelligent grouping of the files and websites, clustering may be used to create a more relevant set of search results compared to normal search engines like Google. There are currently a number of web based clustering tools such as Clusty.

e. Slippy map optimization

f. Flickr's map of photos and other map sites use clustering to reduce the number of markers on a map. This makes it both faster and reduces the amount of visual clutter.

V. Computer science

a. Software evolution

b. Clustering is useful in software evolution as it helps to reduce legacy properties in code by reforming functionality that has become dispersed. It is a form of restructuring and hence is a way of directly preventative maintenance.

c. Image segmentation

d. Clustering can be used to divide a digital image into distinct regions for border detection or object recognition.

e. Evolutionary algorithms

f. Clustering may be used to identify different niches within the population of an evolutionary algorithm so that reproductive opportunity can be distributed more evenly amongst the evolving species or subspecies.

g. Recommender systems

h. Recommender systems are designed to recommend new items based on a user's tastes. They sometimes use clustering algorithms to predict a user's preferences based on the preferences of other users in the user's cluster.
i. Markov chain Monte Carlo methods
j. Clustering is often utilized to locate and characterize extreme in the target distribution.

VI. Social science
  a. Crime analysis
  b. Cluster analysis can be used to identify areas where there are greater incidences of particular types of crime. By identifying these distinct areas or "hot spots" where a similar crime has happened over a period of time, it is possible to manage law enforcement resources more effectively.
  c. Educational data mining
  d. Cluster analysis is for example used to identify groups of schools or students with similar properties.

VII. Others
  a. Mathematical chemistry
  b. To find structural similarity, etc., for example, 3000 chemical compounds were clustered in the space of 90 topological indices.
  c. Climatology
  d. To find weather regimes or preferred sea level pressure atmospheric patterns.
  e. Petroleum geology
  f. Cluster analysis is used to reconstruct missing bottom hole core data or missing log curves in order to evaluate reservoir properties.
  g. Physical geography
  h. The clustering of chemical properties in different sample locations.

1.8 Statistical classification

In machine learning and statistics, classification is the problem of identifying to which of a set of categories (sub-populations) a new observation belongs, on the basis of a training set of data containing observations (or instances) whose category membership is known. The individual observations are analyzed into a set of quantifiable properties, known as various explanatory variables, *features*, etc. These properties may variously be categorical (e.g. "A", "B", "AB" or "O", for blood type), ordinal (e.g. "large", "medium" or "small"),integer-valued (e.g. the number of occurrences of a part word in an email) or real-valued (e.g. a measurement of blood pressure). Some algorithms work only in terms of discrete data and require that real-valued or integer-valued data be *discretized* into groups (e.g. less than 5, between 5 and 10, or greater than 10). An example would be assigning a given email into "spam" or "non-spam" classes or assigning a diagnosis to a given patient as described by observed characteristics of the patient (gender, blood pressure, presence or absence of certain symptoms, etc.).

An algorithm that implements classification, especially in a concrete implementation, is known as a classifier. The term "classifier" sometimes also refers to the mathematical function, implemented by a classification algorithm that maps input data to a category.

In the terminology of machine learning, classification is considered an instance of supervised learning, i.e. learning where a training set of correctly-identified observations is available. The corresponding unsupervised procedure is known as *clustering* (or cluster analysis), and involves grouping data into categories based on some measure of inherent similarity (e.g. the distance between instances, considered as vectors in a multi-dimensional vector space).

Terminology across fields is quite varied. In statistics, where classification is often done with logistic regression or a similar procedure, the properties of observations are termed explanatory variables (or independent variables, repressors, etc.), and the categories to be predicted are known as outcomes, which are considered to be possible values of the dependent variable. In machine learning, the observations are often known as *instances*, the explanatory variables are termed *features* (grouped into a feature vector), and the possible categories to be predicted are *classes*. There is also some argument over whether classification methods that do not involve a statistical model can be considered "statistical". Other fields may use different terminology: e.g. in community ecology, the term "classification" normally refers to cluster analysis, i.e. a type of unsupervised learning, rather than the supervised learning described in this article.

1.9 Regression Analysis:

In statistics, regression analysis is a statistical technique for estimating the relationships among variables. It includes many techniques for modeling and analyzing several variables, when the focus is on the relationship between a dependent variable and one or more independent variables. More specifically, regression analysis helps one understand how the typical value of the dependent variable changes when any one of the independent variables is varied, while the other independent variables are held fixed. Most commonly, regression analysis estimates the conditional expectation of the dependent variable given the independent variables – that is, the average value of the dependent variable when the independent variables are fixed. Less commonly, the focus is on a quantile, or other location parameter of the conditional distribution of the dependent variable given the independent variables. In all cases, the estimation target is a function of the independent variables called the regression function. In regression analysis, it is also of interest to characterize the variation of the dependent variable around the regression function, which can be described by a probability distribution.

Regression analysis is widely used for prediction and forecasting, where its use has substantial overlap with the field of machine learning. Regression analysis is also used to understand which among the independent variables are related to the dependent variable, and to explore the forms of these relationships. In restricted circumstances, regression analysis can be used to infer causal relationships between the independent and dependent variables. However this can lead to illusions or false relationships, so caution is advisable; for example, correlation does not imply causation.

A large body of techniques for carrying out regression analysis has been developed. Familiar methods such as linear regression and ordinary least squares regression are parametric, in that the regression function is defined in terms of a finite number of unknown parameters that are estimated from the data. Nonparametric regression refers to techniques that allow the regression function to lie in a specified set of functions, which may be infinite-dimensional.

The performance of regression analysis methods in practice depends on the form of the data generating process, and how it relates to the regression approach being used. Since the true form of the data-generating process is generally not known, regression analysis often depends to some extent on making assumptions about this process. These assumptions are sometimes testable if many data are available. Regression models for prediction are often useful even when the assumptions are moderately violated, although they may not perform optimally. However, in many applications, especially with small effects or questions of causality based on observational data, regression methods can give misleading results.

CHAPTER 2

Knowledge Discovery in Database

Definition
Goal

KDD Process

As we march into the age of digital information, the problem of data overload looms ominously ahead. Our ability to analyze and understand massive datasets lags far behind our ability to gather and store the data. A new generation of computational techniques and tools is required to support the extraction of useful knowledge from the rapidly growing volumes of data. These techniques and tools are the subject of the emerging field of knowledge discovery in databases (KDD) and data mining. Large databases of digital information are ubiquitous. Data from the neighborhood store's checkout register, your bank's credit card authorization device, records in your doctor's office, patterns in your telephone calls, and many more applications generate streams of digital records archived in huge databases, sometimes in so-called data warehouses. Current hardware and database technology allow efficient and inexpensive reliable data storage and access. However, whether the context is business, medicine, science, or government, the datasets themselves (in raw form) are of little direct value. What is of value is the knowledge that can be inferred from the data and put to use. For example, the marketing database of a consumer goods company may yield knowledge of correlations between sales of certain items and certain demographic groupings. This knowledge can be used to introduce new targeted marketing campaigns with predictable financial return relative to unfocused campaigns. Databases are often a dormant potential resource that, tapped, can yield substantial benefits. This article gives an overview of the emerging field of KDD and data mining, including links with related fields, a definition of the knowledge discovery process, dissection of basic data mining algorithms, and an analysis of the challenges facing practitioners. Impractical Manual Data Analysis The traditional method of turning data into knowledge relies on manual analysis and interpretation. For example, in the health-care industry, it is common for specialists to analyze current trends and changes in health-care data on a quarterly basis. Databases are increasing in size in two ways: the number N of records, or objects, in the database, and the number d of fields, or attributes, per object. Databases containing on the order of N = 109 objects are increasingly common in, for example, the astronomical sciences. The number d of fields can easily be on the order of 102 or even 103 in medical diagnostic applications. Who could be expected to digest billions of records, each with tens or hundreds of fields? Yet the true value of such data lies in the users' ability to extract useful reports, spot interesting events and trends, support decisions and policy based on statistical analysis and inference, and exploit the data to achieve business, operational, or scientific goals. When the scale of data manipulation, exploration, and inference grows beyond human capacities, people look to computer technology to automate the bookkeeping. The problem of knowledge extraction from large databases involves many steps, ranging from data manipulation and retrieval to fundamental mathematical and statistical inference, search, and reasoning. Researchers and practitioners interested in these problems have been meeting since the first KDD Workshop in 1989. Although the problem of extracting knowledge from data (or observations) is not new, automation in the context of large databases opens up many new unsolved problems.

2.1 Kdd process

"

Finding useful patterns in data is known by different names (including data mining) in different communities (e.g., knowledge extraction, information discovery, information harvesting, data archeology, and data pattern processing).

"

The term "data mining "is used most by statisticians, database researchers, and more recently by the MIS and business communities. Here we use the term "KDD" to refer to the overall process of discovering useful knowledge from data. Data mining is a particular step in this process application of specific algorithms for extracting patterns (models) from data. The additional steps in the KDD process, such as data preparation, data selection, data cleaning, incorporation of appropriate prior knowledge, and proper interpretation of the results of mining ensure that useful knowledge is derived from the data. Blind application of data mining methods(rightly criticized as data dredging in the statistical literature)can be a dangerous activity leading to discovery of meaningless patterns.KDD has evolved, and continues to evolve, from the intersection of research in such fields as databases, machine learning, pattern recognition, statistics, artificial intelligence and reasoning with uncertainty, knowledge acquisition for expert systems, data visualization, machine discovery , scientific discovery, information retrieval, and high-performance computing. KDD software systems incorporate theories, algorithms, and
Methods from all of these fields. Database theories and tools provide the necessary infrastructure to store, access, and manipulate data. Data warehousing, a recently popularized term, refers to the current business trend of collecting and cleaning transactional data to make them available

for online analysis and decision support. A popular approach for analysis of data warehouses is called online analytical processing (OLAP).1 OLAP tools focus on providing multidimensional data analysis, which is superior to SQL (a standard data manipulation language) in computing summaries and breakdowns along many dimensions. While current OLAP tools target interactive data analysis, we expect they will also include more automated discovery components in the near future. Fields concerned with inferring models from data—including statistical pattern recognition, applied statistics, machine learning, and neural networks—were the impetus for much early KDD work. KDD largely relies on methods from these fields to find patterns from data in the data mining step of the KDD process. A natural question is: How is KDD different from these other fields? KDD focuses on the overall process of knowledge discovery from data, including how the data is stored and accessed, how algorithms can be scaled to massive datasets and still run efficiently, how results can be interpreted and visualized, and how the overall human-machine interaction can be modeled and supported.KDD places a special emphasis on finding understandable patterns that can be interpreted as useful or interesting knowledge. Scaling and robustness properties of modeling algorithms for large noisy datasets are also of fundamental interest.

Figure 1. Overview of the steps constituting the KDD process

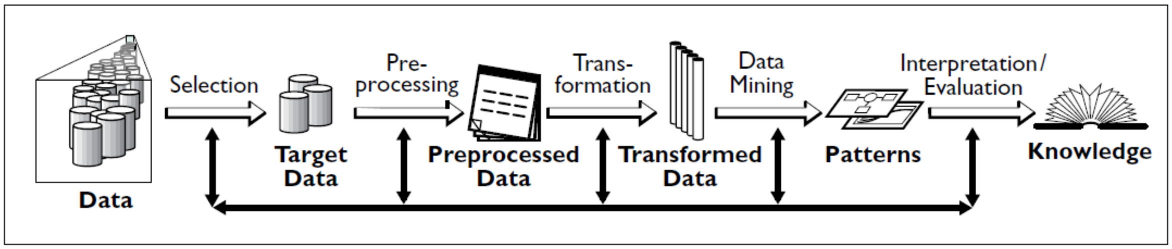


Figure 4 Overview of Kdd

2.2 Goals

I. *Selection*

II. *Preprocessing*

III. *Transformation*

IV. *Data Mining*

V. *Interpretation/Evaluation*

VI. *Target Knowledge Data*

VII. *Preprocessed Data*

VIII. *Transformed Data*

IX. *Patterns Data*

It lies in this multidisciplinary field and how they fit together. We define the KDD process as: The nontrivial process of identifying valid, novel, potentially useful, and ultimately understandable patterns in data. Throughout this article, the term pattern goes beyond its traditional sense to include models or structure in data. In this definition, data comprises a set of facts (e.g., cases in a database), and pattern is an expression in some language describing a subset of the data (or a model applicable to that subset). The term process implies there are many steps involving data preparation, search for patterns, knowledge evaluation, and refinement—all repeated in multiple iterations. The process is assumed to be nontrivial in that it goes beyond computing closed-form quantities; that is, it must involve search for structure, models, patterns, or parameters. The discovered patterns should be valid for new data with some degree of certainty. We also want patterns to be novel (at least to the system, and
preferably to the user) and potentially useful for the user or task. Finally, the patterns should be understandable—if not immediately, then after some
post processing. This definition implies we can define quantitative measures for evaluating extracted patterns. In many cases, it is possible to define measures of certainty (e.g., estimated classification accuracy) or utility (e.g., gain, perhaps in dollars saved due to better predictions or speed-up in a system's response time). Such notions as novelty and understandability are much more subjective. In certain contexts, understandability can be estimated through simplicity (e.g., number of bits needed to describe a pattern). An important notion, called interestingness, is usually taken as an overall measure of pattern value, combining validity, novelty, usefulness, and simplicity. Interestingness functions can be explicitly defined or can be manifested implicitly through an ordering placed by the KDD system on the discovered patterns or models. Data mining is a step in the KDD process consisting of an enumeration of patterns (or models) over the data, subject to some acceptable computational-efficiency limitations. Since the patterns enumerable over any finite dataset are potentially infinite, and because the enumeration of

patterns involves some form of search in a large space, computational constraints place severe limits on the subspace that can be explored by a data mining algorithm. The KDD process is outlined in Figure 1. (We did not show all the possible arrows to indicate that loops can, and do, occur between any two steps in the process; also not shown is the system's performance element, which uses knowledge to make decisions or take actions.) The KDD process is interactive and iterative (with many decisions made by the user), involving numerous steps, summarized as:

I. Learning the application domain: includes relevant prior knowledge and the goals of the application

II. Creating a target dataset: includes selecting a dataset or focusing on a subset of variables or data samples on which discovery is to be performed

III. Data cleaning and preprocessing: includes basic operations, such as removing noise or outliers if appropriate, collecting the necessary information to model or account for noise, deciding on strategies for handling missing data fields, and accounting for time sequence information and known changes, as well as deciding DBMS issues, such as data types,
schema, and mapping of missing and unknown values

IV. Data reduction and projection:includes finding useful features
to represent the data, depending on the goal of the task, and using dimensionality reduction or transformation methods to reduce the effective number of variables under consideration or to find invariant representations for the data

CHAPTER 3

DATA CLUSTERING

WHAT IS CLUSTER ANALYSIS

THE MAIN ELEMENTS OF CLUSTER ANALYSIS

K-MEANS ALGORITHM

DRAWBACK

DATA CLUSTERING

Data clustering, by definition, is an exploratory and descriptive data analysis technique, which has gained a lot of attention, e.g., in statistics, data mining, pattern recognition etc. It is an explorative way to investigate multivariate data sets that contain possibly many different data types. These data sets differ from each other in size with respect to a number of objects and dimensions, or they contain different data types etc. Undoubtedly, the data clustering belongs to the core methods of data mining, in which one focuses on large data sets with unknown underlying structure. The intention of this report is to be an introduction into specific parts of this methodology called cluster analysis. So called partitioning-based clustering methods are flexible methods based on iterative relocation of data points between clusters. The quality of the solutions is measured by a clustering criterion. At each iteration, the iterative relocation algorithms reduce the value of the criterion function until convergence. By changing the clustering criterion, it is possible to construct robust clustering methods that are more insensitive to erroneous and missing data values than classical methods. Surprisingly, most of ”real-data” is of this form . Hence, in the end of this report, an example of robust partitioning-based cluster analysis techniques is presented. Next to this introduction, various definitions for cluster analysis and clusters are discussed. Thereafter, in the third section, a principle of partitioning-based clustering is presented with numerous examples. A special treatment is given for the well-known K-means algorithm. The fourth chapter consists of discussion about robust clustering methods. In the sixth section, a novel partitioning-based method, which is robust against outliers and based on the iterative relocation principle including the treatment for missing values, is introduced. The last section contains the final summary for the report.

3.1 What is cluster analysis ?

Cluster analysis is an important element of exploratory data analysis. It is typically directed to study the internal structure of a complex data set, which can not be described only through the classical second order statistics . Already in 1967, MacQueen stated that clustering applications are considered more as an aid for investigators to obtain qualitative and quantitative understanding of a large amount of multivariate data than only a computational process that finds some unique and definitive grouping for the data. Later, due to its unsupervised, descriptive and summarizing nature, data clustering has also become a core method of data mining and knowledge discovery. Especially during the last decade, the increasing number of large multidimensional data collections have stepped up the development of new clustering algorithms . Generally speaking, the classification of different things is a natural process for human beings. There exist numerous natural examples about different classifications for living things in the world. For example, various animal and plant species are the results of unsupervised categorization processes made by humans (more precisely, domain experts), who have divided objects into separate classes by using their observable characteristics . There were no labels for the species before someone generated them. A child classifies things in an unsupervised manner as well. By observing similarities and dissimilarities between different objects, a child groups those objects into the same or different group. At the time before the computers came available, clustering tasks had to be performed manually. Although it is easy to visually perceive groups from a two- or three-dimensional data set, such ”human clustering” is not likely an inconsistent procedure, since different individuals see things in different ways. The measure of similarity, or the level and direction one is looking at the data, are not consistent between different individuals. By direction we mean the set of features (or combinations of features) that one exploits when classifying objects. For example, people can be classified into a number of groups according to the economical status or the annual alcohol consumption etc. These groupings will not necessarily capture the same individuals . The direction where the user is looking at the data set depends, for example, on her/his background (position, education, profession, culture etc.). It is clear that such things vary a lot among different individuals . Numerous definitions for cluster analysis have been proposed in the literature. The definitions differ slightly from each other in the way to emphasize the different aspects of the methodology. In one of the earliest books on data clustering, Underberg defines cluster analysis as a task, which aims to ”*finding of ”natural groups” from a data set, when little or nothing is known about the category structure*”. Bailey who surveys the methodology from the sociological perspective, defines that ”*cluster analysis seeks to divide a set of objects into a small number of relatively homogeneous groups on the basis of their similarity over N variables.*” *N* is the total number of variables in this case. Moreover, Bailey notes that ”*Conversely variables can be grouped according to their similarity across all objects.*”. Hence, the interest of cluster analysis may be in either grouping of objects or variables, or even both . On the other hand, it is not rare to reduce the number of variables before the actual object grouping, because the data can be easily compressed by substituting the correlating variables with one summarizing and representative variable. From the statistical pattern recognition perspective, Jain et al. define cluster analysis as ”*the organization of collection of patterns (usually represented as a vector of measurements, or a point in a multidimensional space) into clusters based on similarity*”. Hastie et define the goal of cluster analysis from his statistical perspective as a task ”*to partition the observations into groups (”clusters”) such that the pairwise dissimilarities between those assigned to the same cluster tend to be smaller than those in different clusters*”. Tan et al. states from data mining point of view that ”*Cluster analysis divides data into groups that are meaningful, useful, or both.*”. By meaningful they refer to clusters that capture the natural structure of a data set, whereas the useful clusters serve only as an initial setting for some other method, such as *PCA* (*principal component analysis*) or regression methods. For these methods, it may be useful to summarize the data sets beforehand. The first definition emphasizes the unknown structure of the target data sets, which is one of the key assumptions in cluster analysis. This is the main difference between clustering (*unsupervised classification*) and classification (*supervised classification*). In a classification task the category structure is known a priori, whereas the cluster analysis focuses on the object collections, whose class labels are unknown. Jain et al. suggest that the class labels and all other information about data sources, have an influence to the result interpretation, but not to the cluster formation process. On the other hand, the domain understanding is often of use during the configuration of initial parameters or correct number of clusters. The second and third definitions stress the multi-dimensionality of the data objects (observations, records etc.). This is an important notion, since the grouping of objects that possess more than three variables is no easy matter for a human being without automated methods. Naturally, most of the aforementioned definitions address the notion of similarity. Similarity is one of the key issues of cluster analysis, which means that one of the most influential elements of cluster analysis is the choice of an appropriate similarity measure. The similarity measure selection is a data-dependent problem. Anderberg does not use term ”similarity”, but instead he talk about the degree of ”natural association” among objects. Based on the aforementioned definitions and notions, the cluster analysis is summarized as ”*analysisof the unknown structure of a multidimensional data set by determining a (small) number ofmeaningful groups of objects or variables according to a chosen (dis)similarity measure*”. Inthis definition, the term meaningful is understood identically with Tan et al. Even though the visual perception of data clusters is a suitable method up tothree dimensions, in more than three dimensional space the visual perception turns to a complex task and computers become indispensable. As we know that a human classifier is an inconsistent classifier, also different algorithms produce different groupings even for the same data set. Hence, there exist not any universally best clustering algorithm.

3.2 The main elements of cluster analysis

Although the intuitive idea behind cluster analysis is simple, the successful completion of the tasks presume a large number of correct decisions and choices from several alternatives. Anderberg states that there appears to be at least nine major elements in a cluster analysis study before the final results can be attained. Because the current real world data sets contain missing values as well, we complete this element list with data presentation and missing data strategy .

I. Data presentation.
II. Choice of objects.
III. Choice of variables.
IV. What to cluster: data units or variables.
V. Normalization of variables.
VI. Choice of (dis)similarity measures.
VII. Choice of clustering criterion (objective function).
VIII. Choice of missing data strategy.
IX. Algorithms and computer implementation (and their reliability, e.g., convergence)
X. Number of clusters.
XI. Interpretation of results.

These are the most significant parts of the general clustering process. Jain et al. suggest that the strategies used in data collection, data representation, normalization and cluster validity are as important as the clustering strategy itself. According to Hastie et al. choice of the best (dis)similarity measure is even more important than the choice of clustering algorithms. This list could be also completed by validation of the resulting cluster solution . Validation is, on the other hand, closely related to the estimation of the number of clusters and to the result interpretation. For example, the visual exploration of the obtained solutions can be considered a kind of validation technique.

3.2.1 Laplace distributions

”*Do not forget that clusters are, in large part, on the eye of the beholder*.”

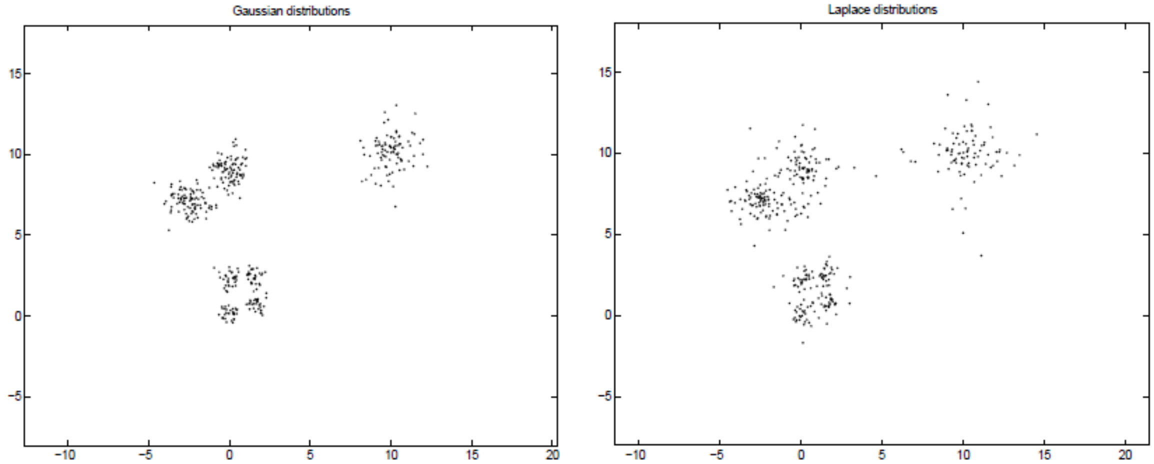


Figure 1: Illustration about the ambiguous number of clusters. On the left, there are seven clusters that are generated from a normal distribution using a set of different location and scatter parameters. Correspondingly, on the right, there are seven clusters that are drawn from the Laplace distribution using the same location and scatter parameters as in the normal case. It is not straightforward to say how many clusters there are, especially in the Laplace-case, because the clusters are inherently more spread in all directions. Although the visual recognition of clusters from a two-dimensional view is usually easy, it is hard to give a formal definition for a cluster. Many authors with contributions in the clustering literature address the lack of the universal and formal definition of a cluster. However, at the same time, they agree that giving one is an intractable problem . The notion of a cluster depends on the application and it is usually weakly defined . The goal of the cluster analysis task effects to the definition as well. Depending on the application, clusters have different shapes and size. Moreover, even the number of inherent clusters in the data is not unambiguous, because it depends on the resolution (local versus global)one is looking at the data . See Figure 1.

Typically clustering methods yield a data description in terms of clusters that possess strong internal similarities . Often one defines the cluster in terms of internal cohesion (homogeneity) and external isolation (separation). Hence, the cluster is often simply considered as a collection of objects, which are similar to one another within the same cluster and dissimilar to the objects in other clusters . An interesting connection to the software engineering is recognized, when we notice that the principle is very similar with the common software architecture rule on”*loose coupling and strong cohesion*”. Such architecture aims to localize effects caused by code modifications . The software components with a large number of mutual links can be considered close to each other. Hence, a good software architecture should contain clearly separated ”component clusters”. Some common definitions are collected from the clustering literature and given below .

”A Cluster is a set of entities which are alike, and entities from different clusters are not alike.”

”A cluster is an aggregation of points in the space such that the *distance* between two points in the cluster is less than the distance between any point in the cluster and any point not in it.”

”Clusters may be described as connected regions of a multidimensional space containing a relatively *high density* of points, separated from other such regions by a region containing a relatively low density of points.”

Although the cluster is an application dependent concept, all clusters are compared with respect to certain properties: density, variance, dimension, shape, and separation

The cluster should be a tight and compact high-density region of data points when compared to the other areas of space. From compactness and tightness, it follows that the degree of dispersion (variance) of the cluster is small. The shape of the cluster is not known a priori. It is determined by the used algorithm and clustering criteria. Separation defines the degree of possible cluster overlap and the distance to each other. Fuzzy clustering methods produce overlapping clusters by assigning the degree of the membership to the clusters for each point . Traditional partitioning clustering methods, such as K-Means, and hierarchical methods produce separated clusters , which means that each data point is assigned to only one cluster. A cluster is defined in a dimension of its variables and, if having a round shape, it is possible to determine its radius. These are the measurable features for any cluster ,but it is not possible to assign universal values or relations to them. Perhaps, the most problematic features are shape and size.

3.2.2 K-means clustering

Basically *K-means* is an iterative process that divides a given data set into *K* disjoint groups. K-means is perhaps the most widely used clustering principle, and especially, the best-known of the partitioning-based clustering methods that utilize prototypes for cluster presentation (a.k.a representative-based algorithm by Estivill- Castro [35]). Quality of K-means clustering is measured through the within-cluster **squared error criterion (e.g., [5, p.165] or [58])**

$$\min_{\mathbf{c}\in\mathbb{N}^n, \{\mathbf{m}_k\}_{i=1}^{K}\in\mathbb{R}^p} \quad \mathcal{J}, \text{ for } \mathcal{J}(\mathbf{c}, \{\mathbf{m}_k\}_{k=1}^{K}, \{\mathbf{x}_i\}_{i=1}^{n}) = \sum_{i=1}^{n} \|\mathbf{x}_i - \mathbf{m}_{(\mathbf{c})_i}\|^2 \qquad (3.1)$$

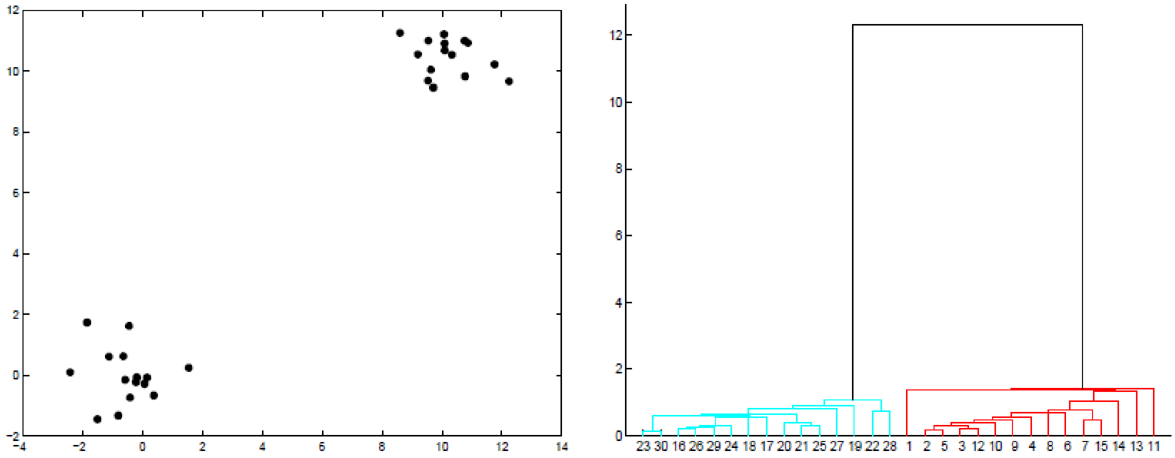


Figure 2: A sample (*n* = 30) from a two dimensional normal distribution *f*(x) = 1 2*N*2((0 0)*T ;* I2) + 1 2*N*2((10 10)*T ;* I2) in the left figure is clustered using the hierarchical single-linkage method. The result is visualized using a dendrogram tree.

$$(\mathbf{c})_i \in \{1, \ldots, K\} \quad \text{for all} \quad i \in \{1, \ldots, n\},$$

where c is a code vector for partition that contains the cluster membership for each object. m(c)*i*is the mean of the cluster, where the data point x*i* is assigned. The sample mean leads to a unique minimum of the within-cluster variance, from which it follows that the problem actually corresponds to the minimization of P*Ki*=1 trace(W*i*), where W*i* is the within-group covariance matrix of the *ith* cluster. Thus, the K means clustering is also referred to as a variance minimization technique . Actually in 1963, before the invention of any K-means algorithm, the minimum variance optimization technique was used by Ward , who proposed an hierarchical algorithm that begins with each data points as its own cluster and proceed by combining points that result in the minimum increase in the error sum of squares value (This method is later referred to both as the Ward's method, and the pairwise nearest neighbor algorithm (PNN), e.g.. As such, K-means clustering tends to produce compact clusters, but not take into account the between-cluster distances. The use of the squared *l*2-norm makes the problem formulation extremely sensitive towards large errors, which means that the formulation is non-robust in statistical sense However, due to its implementational simplicity and computational efficiency, K-means has remained its position as an extremely popular principle for many kind of cluster analysis tasks. It also requires less memory resources than, for instance, hierarchical methods, in which computation is often based on the dissimilarity matrix. By courtesy of its computational efficiency, K-means is also applied to initialization of other more expensive methods (e.g., EM algorithm . The K-means algorithm, which is used to minimize the problem of K-means, has a large number of variants which are described next.

3.3 K-means algorithms

K-means type grouping has a long history. For instance, already in 1958, Fisher investigated this problem in one-dimensional case as a *grouping problem*. At that time, algorithms and computer power were still insufficient for larger-scale problems, but the problem was shown to be interesting with concrete applications. Hence, more efficient procedures than exhaustive search was needed. The seminal versions of the K-means procedure were introduced in the Sixties by Forgy (c.f. discussion in and MacQueen These are perhaps the most widely used versions of the K-means algorithms .The main difference between Forgy's and MacQueen's

algorithms is the order, in which the data points are assigned to the clusters and the cluster centers are updated. The MacQueen's K-means algorithm updates the ”winning” cluster center immediately after every assignment of a data point and all cluster centers one more time after all data points have become assigned to the clusters. The Forgy's method updates the cluster centers only after all data points are assigned to the closest cluster centers. Moreover, another difference is that the Forgy's method iterates until converged while the MacQueen's basic algorithm performs only one complete pass through data. The starting points of the MacQueen's algorithm are often the first *K* data points in the data set. In 1982, Lloyd [88] presented a quantization algorithms for pulse-codemodulation (PCM) of analog signals.

3.4 Drawbacks

Despite the wide popularity of the ordinary K-means algorithms, there are some significant defects that have led to development of numerous alternative versions during the past years *Sensitivity to initial configuration.* Since the basic algorithms are local search heuristics and K-means cost function is non-convex, it is very sensitive to the initial configuration and the obtained partition is often only suboptimal (notthe globally best partition).

*Lack of robustness.*As the sample mean and variance are very sensitive estimate against outliers. So-called breakdown point is zero, which means that one gross errors may distort the estimate completely. The obvious consequent is that the k-means problem formulation is highly non-robust as well. *Unknown number of clusters.*Since the algorithm is a kind ”flat” or ”non-hierarchical”method , it does not provide any information about the number of clusters.

CHAPTER 4

SIGNAL PROCESSING

INTRODUCTION

TYPES OF SIGNAL POCESSING

Signal processing

is an area of systems engineering, electrical engineering and applied mathematics that deals with operations on or analysis of signals, or measurements of time-varying or spatially varying physical quantities.

Signals of interest can include

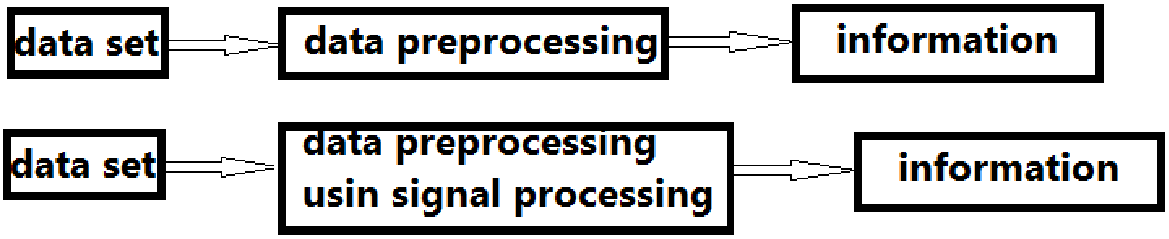
I. sound,

II. images, and sensor data for example

III. biological data such as electro-cardio-grams,

IV. control system signals,

V. telecommunication transmission signals, and many others.



Advantage of using signal processing is that we can implement dimensionality reduction which saves time.

4.1: Signal Processing Techniques

4.1.1 The Fourier transform:

named for Joseph Fourier, is a mathematical transform with many applications in physics and engineering. Very commonly, it expresses a mathematical function of time as a function of frequency, known as its frequency spectrum. The Fourier inversion theorem details this relationship. For instance, the transform of a musical chord made up of pure notes (without overtones) expressed as amplitude as a function of time, is a mathematical representation of the amplitudes and phases of the individual notes that make it up. The function of time is often called the *time domain* representation, and the frequency spectrum the *frequency domain* representation. The inverse Fourier transform expresses a frequency domain function in the time domain. Each value of the function is usually expressed as a complex number (called *complex amplitude*) that can be interpreted as a magnitude and a phase component. The term "Fourier transform" refers to both the transform operation and to the complex-valued function it produces.

In the case of a periodic function, such as a continuous, but not necessarily sinusoidal, musical tone, the Fourier transform can be simplified to the calculation of a discrete set of complex amplitudes, called Fourier series coefficients. Also, when a time-domain function is sampled to facilitate storage or computer-processing, it is still possible to recreate a version of the original Fourier transform according to the Poisson summation formula, also known as discrete-time Fourier transform.

Definition

There are several common conventions for defining the Fourier transform $\hat{f}$ of an integrable function f: R → C (Kaiser 1994, p. 29), (Rahman 2011, p. 11). This article will use the definition:

$$\hat{f}(\xi) = \int_{-\infty}^{\infty} f(x)\ e^{-2\pi i x \xi}\,dx,$$ for every real number ξ.

When the independent variable *x* represents *time* (with SI unit of seconds), the transform variable ξ represents frequency (in hertz). Under suitable conditions, f can be reconstructed from $\hat{f}$ by the inverse transform:

$$f(x) = \int_{-\infty}^{\infty} \hat{f}(\xi)\ e^{2\pi i \xi x}\,d\xi,$$ for every real number *x*.

The statement that f can be reconstructed from $\hat{f}$ is known as the Fourier integral theorem, and was first introduced in Fourier's*Analytical Theory of Heat* , although what would be considered a proof by modern standards was not given until much later .The functions f and $\hat{f}$ often are referred to as a *Fourier integral pair* or *Fourier transform pair*.

For other common conventions and notations, including using the angular frequency ω instead of the frequencyξ, see Other conventions and Other notations below. The Fourier transform on Euclidean space is treated separately, in which the variable x often represents position and ξ momentum.

The motivation for the Fourier transform comes from the study of Fourier series. In the study of Fourier series, complicated but periodic functions are written as the sum of simple waves mathematically represented by sines and cosines. The Fourier transform is an extension of the Fourier series that results when the period of the represented function is lengthened and allowed to approach infinity.

Due to the properties of sine and cosine, it is possible to recover the amplitude of each wave in a Fourier series using an integral. In many cases it is desirable to use Euler's formula, which states that $e^{2\pi i\theta}$= cos(2πθ) + i sin(2πθ), to write Fourier series in terms of the basic waves $e^{2\pi i\theta}$. This has the advantage of simplifying many of the formulas involved, and provides a formulation for Fourier series that more closely resembles the definition followed in this article. Re-writing sines and cosines as complex exponentials makes it necessary for the Fourier coefficients to be complex valued. The usual interpretation of this complex number is that it gives both the amplitude (or size) of the wave present in the function and the phase (or the initial angle) of the wave. These complex exponentials sometimes contain negative "frequencies". If θ is measured in seconds, then the waves$e^{2\pi i\theta}$ and $e^{-2\pi i\theta}$ both complete one cycle per second, but they represent different frequencies in the Fourier transform. Hence, frequency no longer measures the number of cycles per unit time, but is still closely related.

There is a close connection between the definition of Fourier series and the Fourier transform for functions f which are zero outside of an interval. For such a function, we can calculate its Fourier series on any interval that includes the points where f is not identically zero. The Fourier transform is also defined for such a function. As we increase the length of the interval on which we calculate the Fourier series, then the Fourier series coefficients begin to look like the Fourier transform and the sum of the Fourier series of *f* begins to look like the inverse Fourier transform. To explain this more precisely, suppose that *T* is large enough so that the interval [−*T*/2,*T*/2] contains the interval on which *f* is not identically zero. Then the *n*-th series coefficient c_n is given by:

$$c_n = \int_{-T/2}^{T/2} f(x)\ e^{-2\pi i(n/T)x} dx.$$

Comparing this to the definition of the Fourier transform, it follows that $c_n = \hat{f}(n/T)$ since $f(x)$ is zero outside [−*T*/2,*T*/2]. Thus the Fourier coefficients are just the values of the Fourier transform sampled on a grid of width 1/*T*. As *T* increases the Fourier coefficients more closely represent the Fourier transform of the function.

Under appropriate conditions, the sum of the Fourier series of *f* will equal the function *f*. In other words, *f* can be written:

$$f(x) = \frac{1}{T} \sum_{n=-\infty}^{\infty} \hat{f}(n/T)\ e^{2\pi i(n/T)x} = \sum_{n=-\infty}^{\infty} \hat{f}(\xi_n)\ e^{2\pi i\xi_n x}\Delta\xi,$$

where the last sum is simply the first sum rewritten using the definitions $\xi_n = n/T$, and $\Delta\xi = (n + 1)/T - n/T = 1/T$.

This second sum is a Riemann sum, and so by letting $T \to \infty$ it will converge to the integral for the inverse Fourier transform given in the definition section. Under suitable conditions this argument may be made precise .

In the study of Fourier series the numbers c_n could be thought of as the "amount" of the wave present in the Fourier series of *f*. Similarly, as seen above, the Fourier transform can be thought of as a function that measures how much of each individual frequency is present in our function *f*, and we can recombine these waves by using an integral (or "continuous sum") to reproduce the original function.

4.1.2 Fast Fourier Transform:

A fast Fourier transform (FFT) is an efficient algorithm to compute the discrete Fourier transform (DFT) and its inverse. There are many distinct FFT algorithms involving a wide range of mathematics, from simple complex-number arithmetic to group theory and number theory.

A DFT decomposes a sequence of values into components of different frequencies. This operation is useful in many fields (see discrete Fourier transform for properties and applications of the transform) but computing it directly from the definition is often too slow to be practical. An FFT is a way to compute the same result more quickly: computing a DFT of N points in the naïve way, using the definition, takes O(N^2) arithmetical operations, while an FFT can compute the same result in only O(N log N) operations. The difference in speed can be substantial, especially for long data sets where N may be in the thousands or millions—in practice, the computation time can be reduced by several orders of magnitude in such cases, and the improvement is roughly proportional to$N$ / log(N). This huge improvement made many DFT-based algorithms practical; FFTs are of great importance to a wide variety of applications, from digital signal processing and solving partial differential equations to algorithms for quick multiplication of large integers.

The most well known FFT algorithms depend upon the factorization of N, but there are FFTs with O(N log N) complexity for all N, even for prime N. Many FFT algorithms only depend on the fact that $e^{-\frac{2\pi i}{N}}$ is an Nthprimitive root of unity, and thus can be applied to analogous transforms over any finite field, such as number-theoretic transforms. Since the inverse DFT is the same as the DFT, but with the opposite sign in the exponent and a 1/N factor, any FFT algorithm can easily be adapted for it.

The FFT has been described as “the most important numerical algorithm of our lifetime”

4.1.3 Wavelet Transform

The Continuous Wavelet Transform (CWT) associated with the

mother wavelet ψ (*t*) is defined as:

$$W(a,b) = \int_{-\infty}^{\infty} y(t)\psi^{*}_{a,b}(t)dt$$

squareintegrable function, *a* is the scaling parameter, *b* is the translation parameter and , ( ) *a b* ψ *t* is the dilation and translation of the mother wavelet defined as:

$$\psi_{a,b}(t) \equiv \frac{1}{\sqrt{|a|}}\psi\left(\frac{t-b}{a}\right)$$

This CWT [9][10] provides a redundant representation of the signal in the sense that the entire support of *W*(*a*,*b*) need not be used to recover *y*(*t*) . By only evaluating the CWT at dyadic intervals, the signal can be represented compactly as:

$$y(t) = \sum_{k=-\infty}^{\infty}\sum_{j=-\infty}^{\infty} d_j(k)2^{j/2}\psi\left(2^j t - k\right)$$

where ( ) *j d k* is called the discrete wavelet transform (DWT) of *y*(*t*) associated with the wavelet is a scaling function ϕ (*t*) . The scaling function along with the wavelet creates a multi resolution analysis (MRA) of the signal. The scaling function of one level can be represented as a sum of a scaling function of the next finer level.

$$\varphi(t) = \sum_{n=-\infty}^{\infty} h(n)\sqrt{2}\varphi(2t-n)$$

The wavelet is also related to the scaling function by

$$\psi(t) = \sum_{n=-\infty}^{\infty} h_1(n)\sqrt{2}\varphi\ (2t-n)$$

scaling function used to represent the signal as

$$y(t) = \sum_{k=-\infty}^{\infty} c_{jo}(k) 2^{jo/2} \varphi(2^{jo} t - k) + \sum_{k=-\infty}^{\infty} \sum_{j=jo}^{\infty} d_j(k) 2^{j/2} \psi(2^j t - k)$$

Where *jo* represents the coarsest scale spanned by the scaling function.The scaling and wavelet coefficients of the signal *y*(*t*) can be evaluated by using a filter bank of quadrature mirror filters (QMF).

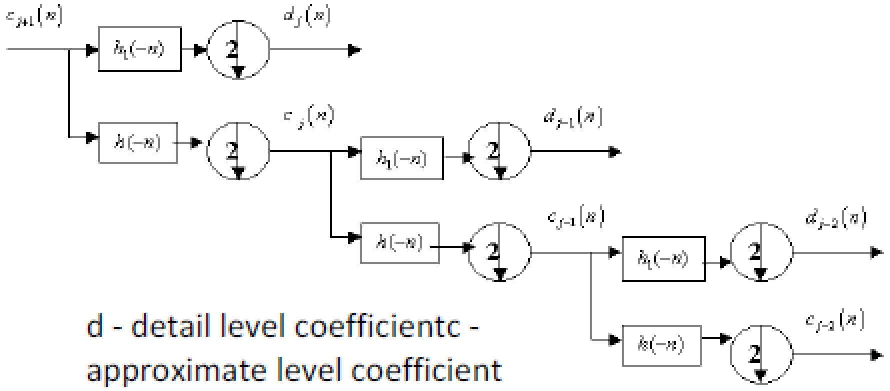
$$c_j(k) = \sum_{m=-\infty}^{\infty} c_{j+1}(m) h(m-2k)$$

$$d_j(k) = \sum_{m=-\infty}^{\infty} c_{j+1}(m) h_1(m-2k)$$

Equations and show that the coefficients at a coarser level can be attained by passing the coefficients at the finer level to their respective filters followed by a decimation of two. The decomposition process is shown in Fig.5.1. For a signal that is sampled at a frequency higher than the Nyquist frequency, the samples are used as

1 ( ) *j c* + *m*

. A three level decomposition of time series data obtained from power signal disturbances, sampled at 12.8 kHz is shown below. The approximation coefficients contain the low frequency information while the detail coefficients contain the high frequency information of the oscillatory transient.



CHAPTER 5

IRIS FLOWER DATA SET

USE OF THE DATA SET

DATA SET

Iris flower data set

The *Iris* flower data set or Fisher's *Iris* data set is a multivariate data set introduced by Sir Ronald Fisher (1936) as an example of discriminant analysis. It is sometimes called Anderson's *Iris* data set because Edgar Anderson collected the data to quantify the morphologic variation of *Iris* flowers of three related species. Two of the three species were collected in the Gaspé Peninsula "all from the same pasture, and picked on the same day and measured at the same time by the same person with the same apparatus".

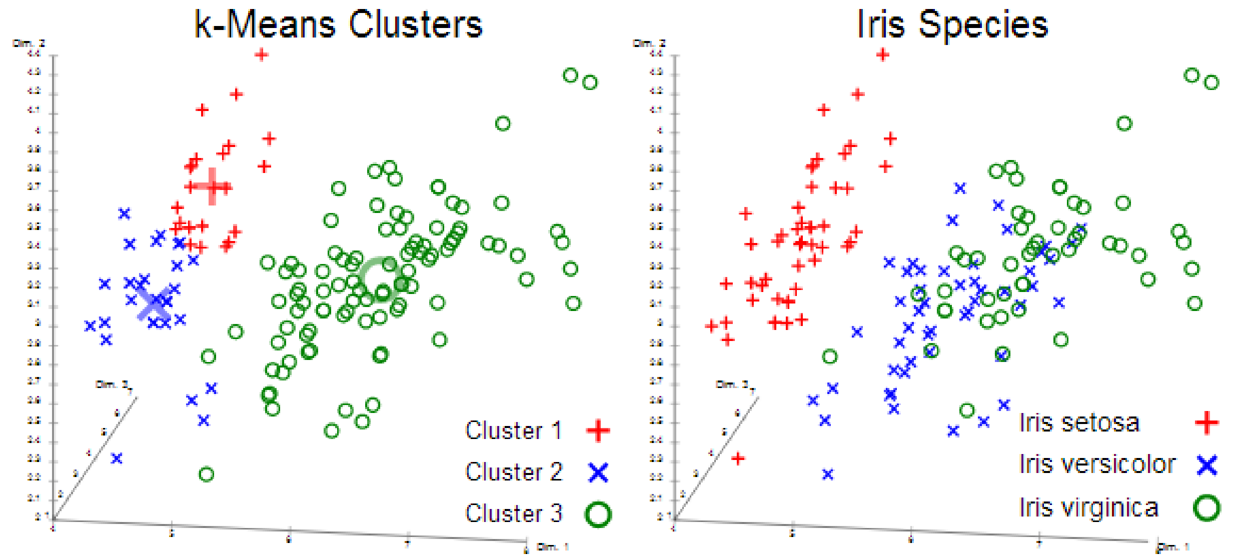
The data set consists of 50 samples from each of three species of *Iris* (*Iris setosa*, *Iris virginica* and *Iris versicolor*). Four features were measured from each sample: the length and the width of the sepals and petals, in centimetres. Based on the combination of these four features, Fisher developed a linear discriminant model to distinguish the species from each other.

5.1 Use of the dataset:

Based on Fisher's linear discriminant model, this data set became a typical test case for many classification techniques in machine learning such as support vector machines.

The use of this data set in cluster analysis however is uncommon, since the data set only contains two clusters with rather obvious separation. One of the clusters contains *Iris setosa*, while the other cluster contains both *Iris virginica* and *Iris versicolor* and is not separable without the species information Fisher used. This makes the data set a good example to explain the difference between supervised and unsupervised techniques in data mining: Fisher's linear discriminant model can only be obtained when the object species are known: class labels and clusters are not necessarily the same.

Nevertheless, all three species of Iris are separable in the projection on the nonlinear branching principal component˙ The data set is approximated by the closest tree with some penalty for the excessive number of nodes, bending and stretching. Then the so-called "metro map" is constructed. The data points are projected into the closest node. For each node the pie diagram of the projected points is prepared. The area of the pie is proportional to the number of the projected points.



k-means clustering result and actual species visualized .

5.2 Data set

Table 1 : Fisher's Iris data

Fisher's *Iris* Data				
Sepal length	**Sepal width**	**Petal length**	**Petal width**	**Species**
5.1	3.5	1.4	0.2	*I. setosa*
4.9	3.0	1.4	0.2	*I. setosa*
4.7	3.2	1.3	0.2	*I. setosa*
4.6	3.1	1.5	0.2	*I. setosa*

Fisher's *Iris* Data

Sepal length	**Sepal width**	**Petal length**	**Petal width**	**Species**
5.0	3.6	1.4	0.2	*I. setosa*
5.4	3.9	1.7	0.4	*I. setosa*
4.6	3.4	1.4	0.3	*I. setosa*
5.0	3.4	1.5	0.2	*I. setosa*
4.4	2.9	1.4	0.2	*I. setosa*
4.9	3.1	1.5	0.1	*I. setosa*
5.4	3.7	1.5	0.2	*I. setosa*
4.8	3.4	1.6	0.2	*I. setosa*
4.8	3.0	1.4	0.1	*I. setosa*
4.3	3.0	1.1	0.1	*I. setosa*
5.8	4.0	1.2	0.2	*I. setosa*
5.7	4.4	1.5	0.4	*I. setosa*
5.4	3.9	1.3	0.4	*I. setosa*

Fisher's *Iris* Data

Sepal length	**Sepal width**	**Petal length**	**Petal width**	**Species**
5.1	3.5	1.4	0.3	*I. setosa*
5.7	3.8	1.7	0.3	*I. setosa*
5.1	3.8	1.5	0.3	*I. setosa*
5.4	3.4	1.7	0.2	*I. setosa*
5.1	3.7	1.5	0.4	*I. setosa*
4.6	3.6	1.0	0.2	*I. setosa*
5.1	3.3	1.7	0.5	*I. setosa*
4.8	3.4	1.9	0.2	*I. setosa*
5.0	3.0	1.6	0.2	*I. setosa*
5.0	3.4	1.6	0.4	*I. setosa*
5.2	3.5	1.5	0.2	*I. setosa*
5.2	3.4	1.4	0.2	*I. setosa*
4.7	3.2	1.6	0.2	*I. setosa*

Fisher's *Iris* Data

Sepal length	**Sepal width**	**Petal length**	**Petal width**	**Species**
4.8	3.1	1.6	0.2	*I. setosa*
5.4	3.4	1.5	0.4	*I. setosa*
5.2	4.1	1.5	0.1	*I. setosa*
5.5	4.2	1.4	0.2	*I. setosa*
4.9	3.1	1.5	0.2	*I. setosa*
5.0	3.2	1.2	0.2	*I. setosa*
5.5	3.5	1.3	0.2	*I. setosa*
4.9	3.6	1.4	0.1	*I. setosa*
4.4	3.0	1.3	0.2	*I. setosa*
5.1	3.4	1.5	0.2	*I. setosa*
5.0	3.5	1.3	0.3	*I. setosa*
4.5	2.3	1.3	0.3	*I. setosa*
4.4	3.2	1.3	0.2	*I. setosa*

Fisher's *Iris* Data

Sepal length	**Sepal width**	**Petal length**	**Petal width**	**Species**
5.0	3.5	1.6	0.6	*I. setosa*
5.1	3.8	1.9	0.4	*I. setosa*
4.8	3.0	1.4	0.3	*I. setosa*
5.1	3.8	1.6	0.2	*I. setosa*
4.6	3.2	1.4	0.2	*I. setosa*
5.3	3.7	1.5	0.2	*I. setosa*
5.0	3.3	1.4	0.2	*I. setosa*
7.0	3.2	4.7	1.4	*I. versicolor*
6.4	3.2	4.5	1.5	*I. versicolor*
6.9	3.1	4.9	1.5	*I. versicolor*
5.5	2.3	4.0	1.3	*I. versicolor*
6.5	2.8	4.6	1.5	*I. versicolor*
5.7	2.8	4.5	1.3	*I. versicolor*

Fisher's *Iris* Data

Sepal length	**Sepal width**	**Petal length**	**Petal width**	**Species**
6.3	3.3	4.7	1.6	*I. versicolor*
4.9	2.4	3.3	1.0	*I. versicolor*
6.6	2.9	4.6	1.3	*I. versicolor*
5.2	2.7	3.9	1.4	*I. versicolor*
5.0	2.0	3.5	1.0	*I. versicolor*
5.9	3.0	4.2	1.5	*I. versicolor*
6.0	2.2	4.0	1.0	*I. versicolor*
6.1	2.9	4.7	1.4	*I. versicolor*
5.6	2.9	3.6	1.3	*I. versicolor*
6.7	3.1	4.4	1.4	*I. versicolor*
5.6	3.0	4.5	1.5	*I. versicolor*
5.8	2.7	4.1	1.0	*I. versicolor*
6.2	2.2	4.5	1.5	*I. versicolor*

Fisher's *Iris* Data

Sepal length	**Sepal width**	**Petal length**	**Petal width**	**Species**
5.6	2.5	3.9	1.1	*I. versicolor*
5.9	3.2	4.8	1.8	*I. versicolor*
6.1	2.8	4.0	1.3	*I. versicolor*
6.3	2.5	4.9	1.5	*I. versicolor*
6.1	2.8	4.7	1.2	*I. versicolor*
6.4	2.9	4.3	1.3	*I. versicolor*
6.6	3.0	4.4	1.4	*I. versicolor*
6.8	2.8	4.8	1.4	*I. versicolor*
6.7	3.0	5.0	1.7	*I. versicolor*
6.0	2.9	4.5	1.5	*I. versicolor*
5.7	2.6	3.5	1.0	*I. versicolor*
5.5	2.4	3.8	1.1	*I. versicolor*
5.5	2.4	3.7	1.0	*I. versicolor*

Fisher's *Iris* Data

Sepal length	Sepal width	Petal length	Petal width	Species
5.8	2.7	3.9	1.2	*I. versicolor*
6.0	2.7	5.1	1.6	*I. versicolor*
5.4	3.0	4.5	1.5	*I. versicolor*
6.0	3.4	4.5	1.6	*I. versicolor*
6.7	3.1	4.7	1.5	*I. versicolor*
6.3	2.3	4.4	1.3	*I. versicolor*
5.6	3.0	4.1	1.3	*I. versicolor*
5.5	2.5	4.0	1.3	*I. versicolor*
5.5	2.6	4.4	1.2	*I. versicolor*
6.1	3.0	4.6	1.4	*I. versicolor*
5.8	2.6	4.0	1.2	*I. versicolor*
5.0	2.3	3.3	1.0	*I. versicolor*
5.6	2.7	4.2	1.3	*I. versicolor*

Fisher's *Iris* Data

Sepal length	**Sepal width**	**Petal length**	**Petal width**	**Species**
5.7	3.0	4.2	1.2	*I. versicolor*
5.7	2.9	4.2	1.3	*I. versicolor*
6.2	2.9	4.3	1.3	*I. versicolor*
5.1	2.5	3.0	1.1	*I. versicolor*
5.7	2.8	4.1	1.3	*I. versicolor*
6.3	3.3	6.0	2.5	*I. virginica*
5.8	2.7	5.1	1.9	*I. virginica*
7.1	3.0	5.9	2.1	*I. virginica*
6.3	2.9	5.6	1.8	*I. virginica*
6.5	3.0	5.8	2.2	*I. virginica*
7.6	3.0	6.6	2.1	*I. virginica*
4.9	2.5	4.5	1.7	*I. virginica*
7.3	2.9	6.3	1.8	*I. virginica*

Fisher's *Iris* Data

Sepal length	Sepal width	Petal length	Petal width	Species
6.7	2.5	5.8	1.8	*I. virginica*
7.2	3.6	6.1	2.5	*I. virginica*
6.5	3.2	5.1	2.0	*I. virginica*
6.4	2.7	5.3	1.9	*I. virginica*
6.8	3.0	5.5	2.1	*I. virginica*
5.7	2.5	5.0	2.0	*I. virginica*
5.8	2.8	5.1	2.4	*I. virginica*
6.4	3.2	5.3	2.3	*I. virginica*
6.5	3.0	5.5	1.8	*I. virginica*
7.7	3.8	6.7	2.2	*I. virginica*
7.7	2.6	6.9	2.3	*I. virginica*
6.0	2.2	5.0	1.5	*I. virginica*
6.9	3.2	5.7	2.3	*I. virginica*

Fisher's *Iris* Data

Sepal length	**Sepal width**	**Petal length**	**Petal width**	**Species**
5.6	2.8	4.9	2.0	*I. virginica*
7.7	2.8	6.7	2.0	*I. virginica*
6.3	2.7	4.9	1.8	*I. virginica*
6.7	3.3	5.7	2.1	*I. virginica*
7.2	3.2	6.0	1.8	*I. virginica*
6.2	2.8	4.8	1.8	*I. virginica*
6.1	3.0	4.9	1.8	*I. virginica*
6.4	2.8	5.6	2.1	*I. virginica*
7.2	3.0	5.8	1.6	*I. virginica*
7.4	2.8	6.1	1.9	*I. virginica*
7.9	3.8	6.4	2.0	*I. virginica*
6.4	2.8	5.6	2.2	*I. virginica*
6.3	2.8	5.1	1.5	*I. virginica*

Fisher's *Iris* Data				
Sepal length	**Sepal width**	**Petal length**	**Petal width**	**Species**
6.1	2.6	5.6	1.4	*I. virginica*
7.7	3.0	6.1	2.3	*I. virginica*
6.3	3.4	5.6	2.4	*I. virginica*
6.4	3.1	5.5	1.8	*I. virginica*
6.0	3.0	4.8	1.8	*I. virginica*
6.9	3.1	5.4	2.1	*I. virginica*
6.7	3.1	5.6	2.4	*I. virginica*
6.9	3.1	5.1	2.3	*I. virginica*
5.8	2.7	5.1	1.9	*I. virginica*
6.8	3.2	5.9	2.3	*I. virginica*
6.7	3.3	5.7	2.5	*I. virginica*
6.7	3.0	5.2	2.3	*I. virginica*
6.3	2.5	5.0	1.9	*I. virginica*

Fisher's *Iris* Data

Sepal length	Sepal width	Petal length	Petal width	Species
6.5	3.0	5.2	2.0	*I. virginica*
6.2	3.4	5.4	2.3	*I. virginica*
5.9	3.0	5.1	1.8	*I. virginica*

5.3Matlab code for clustering on iris data set:

```
clc;
%clear all;
close all;
load iris;
y=zeros(1,15);
y1=zeros(1,15);
y2=zeros(1,15);
fori=1:50
    x=iris{i,1};
    c=x(1,1:15);
    y=[y;c];
end
fori=51:100
x1=iris{i,1};
     c1=x1(1,1:15);
     y1=[y1;c1];
end
fori=101:150
x2=iris{i,1};
    c2=x2(1,1:15);
    y2=[y2;c2];
end
y=y(2:end,:);
y1=y1(2:end,:);
y2=y2(2:end,:);
for j=4:4:12
y(:,j)=[' '];
y1(:,j)=[' '];
y2(:,j)=[' '];
end
y=str2num(y);
y1=str2num(y1);
y2=str2num(y2);
%% Firstly defining  the data...+ones(80,4);-ones(80,4)
X = [y;y1;y2];
%X = [data2(:,1); data2(:,2)];
% Now the kmeans is applied...
opts = statset('Display','final');
[cidx, ctrs] = kmeans(X, 3,'Distance','city', ...
                              'Replicates',5, 'Options',opts);
%% Now look at the cluster formation....
figure,h=plot(X(cidx==1,1),X(cidx==1,2),X(cidx==1,3),X(cidx==1,4),'r.', ...
                               X(cidx==2,1),X(cidx==2,2),X(cidx==2,3),X(cidx==2,4),'b.',...
                               X(cidx==3,1),X(cidx==3,2),X(cidx==3,3),X(cidx==3,4),'g.',...
ctrs(:,1),ctrs(:,2),ctrs(:,3),ctrs(:,4),'kx');
set(h,'linestyle','none');
```

Appendix - A

Clustering of oxygen isotope Without signal

```
clc
clear all
close all
load  oxygen_isotope
time=data(:,1:15);
oxy=data(:,16:25);
time2=time(:,9:15);
for i=1:length(oxy)
oxy1(i,:)=str2double(oxy(i,:));
end
for i=1:length(time2)
time3(i,:)=str2double(time2(i,:));
end
x=timeseries(oxy1,'Name','Oxygen Isotope');
%%
newdata=x.Data;
time4=x.Time;
plot(time4,newdata);
le=length(newdata);
y=newdata(1:72,:);
y1=newdata(73:144,:);
y2=newdata(145:216,:);
y3=newdata(217:288,:);
x=[y y1 y2 y3];

y=newdata(289:(288+72),:);
y1=newdata((288+73):(288+144),:);
y2=newdata((288+145):(288+216),:);
y3=newdata((288+217):(288+288),:);
x2=[y y1 y2 y3];

y=newdata((288+289):(288+288+72),:);
```

```
y1=newdata((288+288+73):(288+288+144),:);
y2=newdata((288+288+145):(288+288+216),:);
y3=newdata((288+288+217):(288+288+288),:);
x3=[y y1 y2 y3];
%% Firstly defining  the data...+ones(80,4);-ones(80,4)
X = [x;x2;x3];
%X = [data2(:,1); data2(:,2)];
% Now the kmeans is applied...
opts = statset('Display','final');
[cidx, ctrs] = kmeans(X, 3,'Distance','city', ...
                              'Replicates',5, 'Options',opts);
%% Now look at the cluster formation....
figure,h=plot(X(cidx==1,1),X(cidx==1,2),X(cidx==1,3),X(cidx==1,4),'r.', ...

X(cidx==2,1),X(cidx==2,2),X(cidx==2,3),X(cidx==2,4),'b.',...

X(cidx==3,1),X(cidx==3,2),X(cidx==3,3),X(cidx==3,4),'g.',...
                               ctrs(:,1),ctrs(:,2),ctrs(:,3),ctrs(:,4),'kx');
                           set(h,'linestyle','none');
% this are the initial result for the implementation....
```

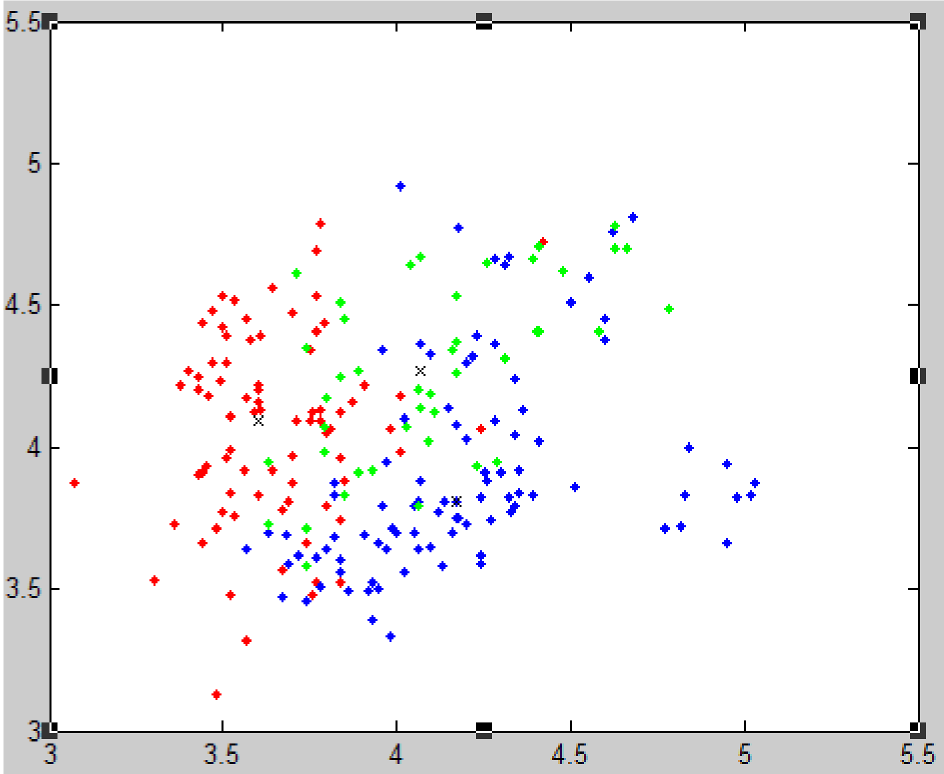


Figure 5 Result of clustering without signal processing

Appendix - B

Clustering of oxygen isotope with signal processing

```
clc
clear all
close all
load  oxygen_isotope
time=data(:,1:15);
oxy=data(:,16:25);
time2=time(:,9:15);
for i=1:length(oxy)
oxy1(i,:)=str2double(oxy(i,:));
end
for i=1:length(time2)
time3(i,:)=str2double(time2(i,:));
end
x=timeseries(oxy1,'Name','Oxygen Isotope');
%%
newdata=x.Data;
time4=x.Time;
plot(time4,newdata);
le=length(newdata);
y=newdata(1:72,:);
y1=newdata(73:144,:);
y2=newdata(145:216,:);
y3=newdata(217:288,:);
x=[y y1 y2 y3];

y=newdata(289:(288+72),:);
y1=newdata((288+73):(288+144),:);
y2=newdata((288+145):(288+216),:);
y3=newdata((288+217):(288+288),:);
x2=[y y1 y2 y3];

y=newdata((288+289):(288+288+72),:);
y1=newdata((288+288+73):(288+288+144),:);
y2=newdata((288+288+145):(288+288+216),:);
y3=newdata((288+288+217):(288+288+288),:);
```

```
x3=[y y1 y2 y3];
%% Firstly defining  the data...+ones(80,4);-ones(80,4)
X = [x;x2;x3];
s=fft(X);
s1=abs(s);
X=s1;
%X = [data2(:,1); data2(:,2)];
% Now the kmeans is applied...
opts = statset('Display','final');
[cidx, ctrs] = kmeans(X, 3,'Distance','city', ...
                              'Replicates',5, 'Options',opts);
%% Now look at the cluster formation....
figure,h=plot(X(cidx==1,1),X(cidx==1,2),X(cidx==1,3),X(cidx==1,4),'r.', ...

X(cidx==2,1),X(cidx==2,2),X(cidx==2,3),X(cidx==2,4),'b.',...

X(cidx==3,1),X(cidx==3,2),X(cidx==3,3),X(cidx==3,4),'g.',...
                               ctrs(:,1),ctrs(:,2),ctrs(:,3),ctrs(:,4),'kx');
                           set(h,'linestyle','none');
                           axis([0 20 0 20]);
% this are the initial result for the implementation....
```

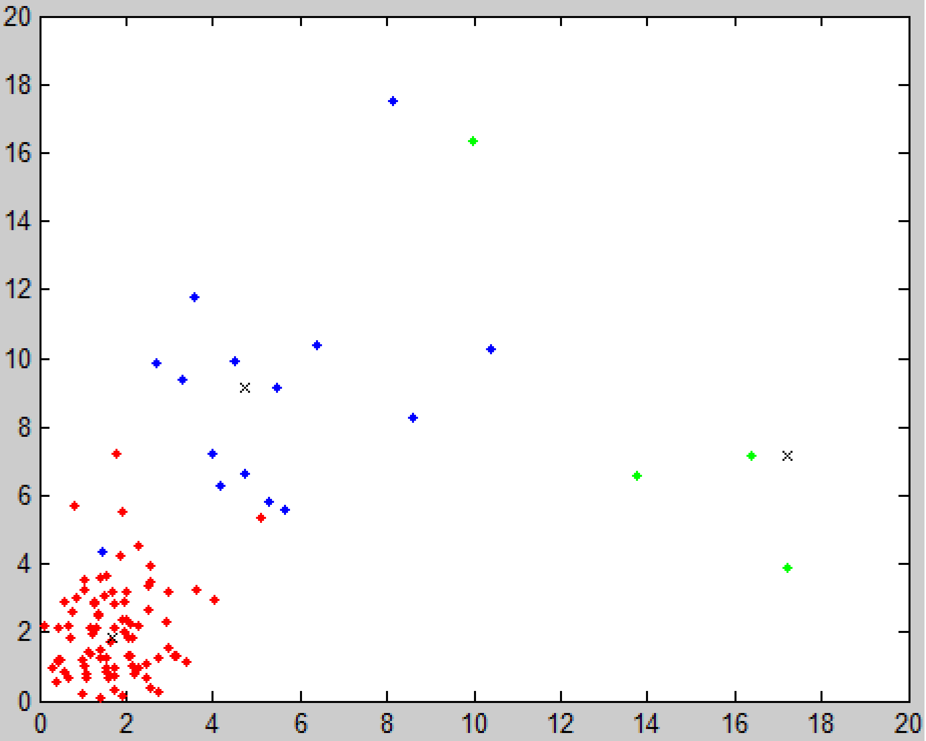


Figure 6 result of clustering with signal processing

References

1. ”Signal processing for mining information”,SSI yengar,NBlakrishnan
2. Introduction to partitioning based clustering methods with a robust example”,SamiAyramo,Tommi Karkkainen
3. The KDD Process for extracting useful information from volumes of data,Usama Fayyad,Gregory Piatetsky-Shapiro,Padhraic Smyth
4. The Challenges of clustering High Dimensional Data
   Micheal Steibach , Vipin Kumar