To Study The Efficiency of K-means Algorithm for Partitional Clustering

A thesis

Submitted in partial fulfillment of the requirement for the degree of
Master of Computer Technology of

Jadavpur University

By

Priyanka Nandi
Registration No :137124 of 2016-2017
Examination Roll No: 001610504021

Under the Guidance of

Prof. Nirmalaya Chowdhury

Department of Computer Science and Engineering

Jadavpur University, Kolkata 700032

2019

Declaration of Originality and Compliance of Academic Ethics

I hereby declare that this thesis contains literature survey and original research work by the undersigned candidate, as part of the Master of Computer Technology Degree of the Department of computer Science and Engineering. All information in this document has been obtained and presented in accordance with academic rules and ethical conduct.

I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name: Priyanka Nandi

Exam Roll No : M6TCT19021

Roll Number: 001610504021

Thesis Title: To Study The Efficiency of K-means Algorithm for Partitional Clustering

Signature with date:

TO WHOM IT MAY CONCERN

This is to certify that the work embodied in thesis entitled **“To Study The Efficiency of K-means algorithm for Partitional Clustering”** has been satisfactorily completed by Priyanka Nandi. It is a bona fide piece of work carried out under my supervision and guidance for the partial fulfillment of the requirements for the award of the degree of Master of Computer Technology of the Department of Computer Science and Engineering, a faculty of engineering and Technology, Jadavpur University, during the academic year 2018-2019.

Prof. Nirmalya Chowdhury (Guide)
Department of Computer Science
and Engineering
Jadavpur University

Forwarded By:

Prof. Mahantapas Kundu
Head of the Department
Department of Computer Science and Engineering
Jadavpur University

Prof. Chiranjib Bhattacharjee
Dean, Faculty of Engineering and Technology,
Jadavpur University

Department of Computer Science and Engineering

Faculty of Engineering and Technology

Jadavpur University, Kolkata 700032

Certificate of Approval

This is to certify that the thesis entitled **"To Study The Efficiency of K-means algorithm for Partitional Clustering"** is a bona-fide record of work carried out by **Priyanka Nandi** in partial fulfillment of the requirements for the award of the degree of **Master of Technology in Computer Technology** in the **Department of Computer Science and Engineering, Jadavpur University** during the period June 2018 to May 2019. It is understood by this approval that the undersigned do not necessarily endorse or approve any statement made, opinion expressed or conclusion therein but approve the thesis only for the purpose for which it has been submitted.

Examiners:

1.

2.

Acknowledgement

I express my sincere gratitude to Prof. Nirmalaya Chowdhury, my guide for his affectionate and valuable guidance without whose help the present work could not have been a successful one. I am also indebted to him as a Professor who introduced me to the world of Data Mining and Pattern recognition.

Also I thank Prof. Mahantapas Kundu, the head of the Department of Computer Science and Engineering, for his assistance in allowing me to work in the departmental laboratory without which my work would have been incomplete.

I would also lie to convey my sincere gratitude to all my respected teachers and faculty members of this department for their invaluable suggestion and kind cooperation.

I express my thanks to all friends of my class.

Last but not the least, the encouragement given by my Father Mr. Bankim Chandra Nandi and my mother Moly Nandi who always been a constant source of inspiration and whose encouragement is beyond linguistic expression for me.

PRIYANKA NANDI
Registration No: 137124 of 2016-2017
Exam Roll No: M6TCT19021
Department of Computer Science & Engineering
Jadavpur University

Chapter 1:
INTRODUCTION

1.1 Introduction to Pattern Recognition

What is a Pattern?

A pattern is defined by the common denominator among the multiple instance of an entity. Based on the common features among those instances of an entity a pattern can be extracted. For example, based on common features in all the finger print images we can exact a pattern of finger print. A pattern could be a finger print image, handwritten word, human face , speech signal , DNA sequence .

A pattern class is a family of patterns that share some common properties. Pattern classes are donated $\omega_1, \omega_2, \ldots . \omega_n$ where ω is the number of classes. To extract a pattern from data there are several stages described below.

1) Data cleaning :- This stage is used to remove noise and inconsistent data.
2) Data Integration :- In this stage multiple data sources may be combined
3) Data Selection:- Data relevant to the analysis task are retrieved from the data base.
4) Data Transformation:- Data are transformed or consolidated into forms appropriate for mining by performing summary or aggregation operations , for instance .
5) Data Mining:- In this process some intelligent methods like clustering , classification are applied in order to extract the data pattern .
6) Pattern Extraction and Evaluation :- In this stage patterns are extracted and evaluated to get knowledge .

1.1.1 What is Pattern Recognition?

Pattern recognition is the branch of artificial intelligence that enables a machine to recognize a “pattern” or “regularity” of a fixed structure in data. To do this we need to “teach” machine i.e. to train the machine by labeling the data. This description of pattern recognition makes it synonymous to “Machine learning”. Interestingly, it is a fact that the two fields can rarely be separated. In most cases, these two fields go hand in hand. When pattern are recognized using a “labeled data” it is termed as “Supervised learning” i.e. data is described (labels) and category (class) is known. While on the other hand, if the data is “unlabeled” it is termed as “Unsupervised learning” i.e. the category or class is unknown at the time of learning. Like machine learning, other related terms to Pattern recognition are data mining and knowledge discovery in databases as they largely overlap in their scope. Furthermore, Pattern recognition has its origins in artificial intelligence as stated above, and the term is popular in the context of computer vision. there are two approaches in pattern reignition techniques defined below.

1) Statistical Pattern Recognition

Statistical pattern recognition classifies a pattern based on a set of extracted features. Each pattern is represented in terms of d features and is viewed as a point in a dimensional feature space. The objective here is to choose those features that allow pattern vectors belonging to different categories to occupy compact and disjoint regions in d-dimensional space.

2) Structural Pattern Recognition

Structural pattern recognition attempts to classify patterns based on a set of extracted features and an underlying statistical model for the generation of these patterns. Ideally, this is achieved with a rather straight forward.

Procedure :

- Determine the feature vector,
- Train the system,
- Classify the patterns.

Unfortunately, there are also many problems where patterns contain structural and relational information that are difficult or impossible to quantify in feature vector form.

Representation: -

A pattern is represented by a set of d features , or attributes , viewed as a d-dimensional vector $x = (x_1, x_2, x_3, \dots \dots . x_d)^t$, it is called a pattern vector of feature vector .

<u>Two Modes of Pattern Recognition System :-</u>

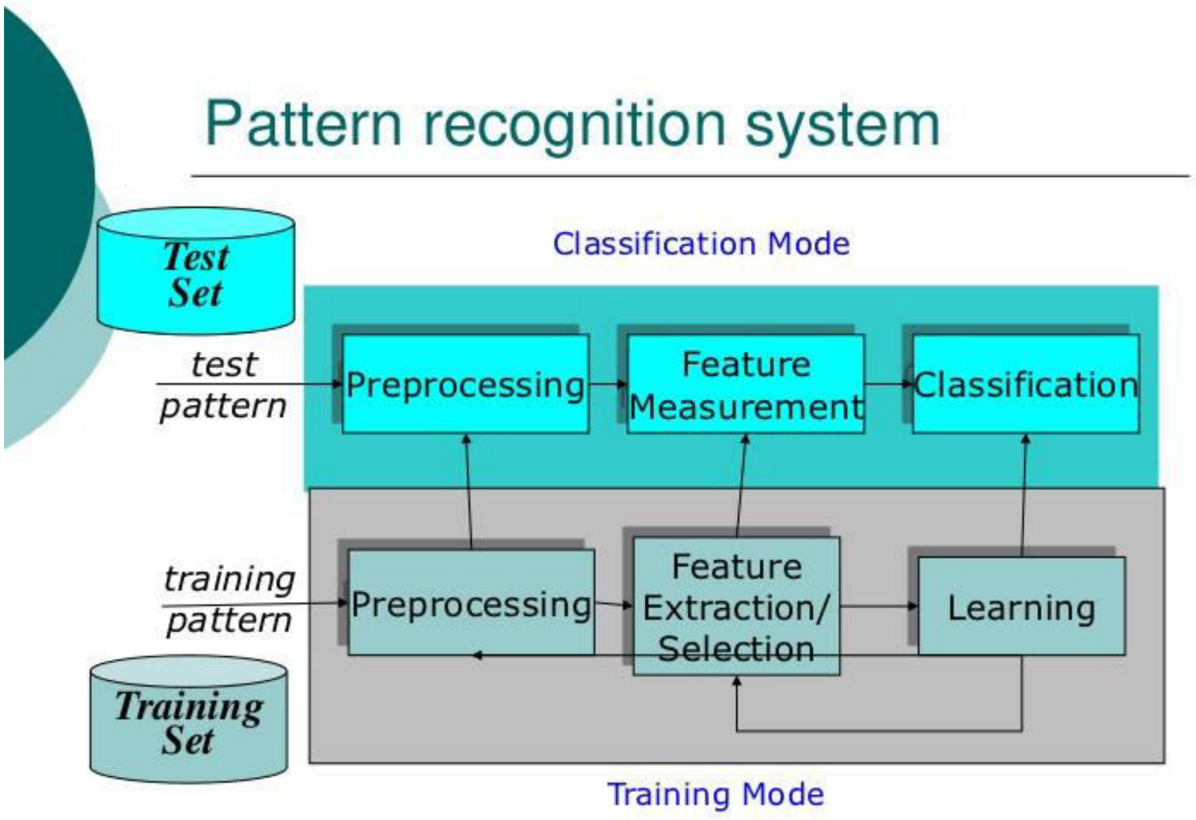


FIG-1 : Modes Of Pattern Recognition System

The problem of pattern recognition can be divided into two parts :

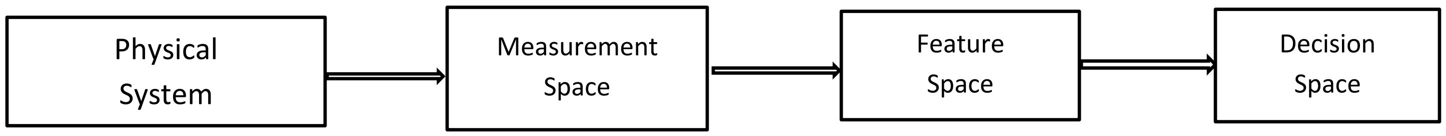
I. It is concerned with the study of recognition mechanism of patterns by human and another living organism.

II. It deals with development of theory and techniques for designing a device which can perform these recognition tasks automatically. This area is related to engineering, computer and information sciences . In this curriculum, we shall be dealing with the second part i.e the problems of automatic machine recognition of patterns.

Operating Stages in a Pattern Recognition System:

Computer recognition of patterns can be viewed as a two-fold task consisting of properties, which are represented numerically by some set of measurements constituting what is called measurement space. Feature selection and extraction in pattern recognition parlance is a process of selecting a map of the form $X = f(y)$ by which a sample $y(y_1, y_2, y_3, \dots\dots y_n)$ in a measurement space Ω_y is transformed into a point $x(x_1, x_2, x_3, \dots\dots x_n)$ in a n dimension feature space Ω_x. The pattern space, which now is transformed into the feature space is, (i) finite

dimensional, (ii) low dimensional and (iii) does contain sufficient information to satisfactorily perform the task of classification. The function
F(Y) is such that it minimizes the intra-set distance while maximizing the intra-set distances in the feature space Ω x. The process of deriving a decision rule on the basis of a finite set of labeled samples for classifying a point in the feature space corresponding to an unlabeled sample and its implementation is called 'Pattern Classification'.



1.1.3 Pattern Recognition Methods:-

Pattern Recognition

- ➔ Supervised
  - Classification
  - Regression
- ➔ Semi Supervised
- ➔ Unsupervised
  - Hierarchical
  - Non-Hierarchical

1.1.3.1 Supervised Method

Supervised methods are methods that attempt to discover the relationship between input attributes (sometimes called independent variables) and a target attribute (sometimes referred to as a dependent variable). The relationship discovered is represented in a structure referred to as a model. Usually models describe and explain phenomena, which are hidden in the dataset and can be used for predicting the value of the target attribute knowing the values of the input attributes. The supervised methods can be implemented in a variety of domains such as marketing, finance and manufacturing.

In supervised learning one is furnished with input (x1,x2,….) and output (y1,y2,……) are challenged with finding a function that approximates this behavior in a generalize fashion . the output could be a class label or a real number.

Attributes (sometimes called field, variable or feature) are typically one of two types: nominal (values are members of an unordered set), or numeric (values are real numbers). When the attribute ai is nominal, it is useful to denote by dom(ai) = {vi,1, vi,2, . . . , vi,|dom(ai)|} its domain values, where |dom(ai)| stands for its finite cardinality. In a similar way, dom(y) = {c1, . . . , c|dom(y)|} represents the domain of the target attribute. Numeric attributes have infinite cardinalities.

Steps in Supervised Learning

While there are many Statistics and Machine Learning Toolbox algorithms for supervised learning, most use the same basic workflow for obtaining a predictor model. (Detailed instruction on the steps for ensemble learning is in Framework for Ensemble Learning.) The steps for supervised learning are:

- Prepare Data
- Choose an Algorithm
- Fit a Model
- Choose a Validation Method
- Examine Fit and Update Until Satisfied
- Use Fitted Model for Predictions

Prepare Data

All supervised learning methods start with an input data matrix, usually called X here. Each row of X represents one observation. Each column of X represents one variable, or predictor. Represent missing entries with NaN values in X. Statistics and Machine Learning Toolbox supervised learning algorithms can handle NaN values, either by ignoring them or by ignoring any row with a NaN value.

You can use various data types for response data Y. Each element in Y represents the response to the corresponding row of X. Observations with missing Y data are ignored.

- For regression, Y must be a numeric vector with the same number of elements as the number of rows of X.

Choose an Algorithm

There are tradeoffs between several characteristics of algorithms, such as:

- Speed of training
- Memory usage
- Predictive accuracy on new data

- Transparency or interpretability, meaning how easily you can understand the reasons an algorithm makes its predictions

  Details of the algorithms appear in Characteristics of Classification Algorithms. More detail about ensemble algorithms is in Choose an Applicable Ensemble Aggregation Method.

Fit a Model

ALGORITHM
CLASSIFICATION TREES
REGRESSION TREES
DISCRIMINANT ANALYSIS (CLASSIFICATION)
***K*-NEAREST NEIGHBORS (CLASSIFICATION)**
NAIVE BAYES (CLASSIFICATION)
SUPPORT VECTOR MACHINES (SVM) FOR CLASSIFICATION
SVM FOR REGRESSION
MULTICLASS MODELS FOR SVM OR OTHER CLASSIFIERS
CLASSIFICATION ENSEMBLES
REGRESSION ENSEMBLES
CLASSIFICATION OR REGRESSION TREE ENSEMBLES (E.G., RANDOM FORESTS [1]) IN PARALLEL

Choose a Validation Method

The three main methods to examine the accuracy of the resulting fitted model are:

- Examine the resubstitution error. For examples, see:
  - Classification Tree Resubstitution Error
  - Cross Validate a Regression Tree
  - Test Ensemble Quality
  - Example: Resubstitution Error of a Discriminant Analysis Classifier
- Examine the cross-validation error. For examples, see:
  - Cross Validate a Regression Tree
  - Test Ensemble Quality
  - Estimate Generalization Error of Boosting Ensemble
  - Cross Validating a Discriminant Analysis Classifier
- Examine the out-of-bag error for bagged decision trees. For examples, see:
  - Test Ensemble Quality
  - Bootstrap Aggregation (Bagging) of Regression Trees Using TreeBagger
  - Bootstrap Aggregation (Bagging) of Classification Trees Using TreeBagger

Examine Fit and Update Until Satisfied

After validating the model, you might want to change it for better accuracy, better speed, or to use less memory.

- Change fitting parameters to try to get a more accurate model. For examples, see:
  - Tune RobustBoost
  - Handle Imbalanced Data or Unequal Misclassification Costs in Classification Ensembles10
  - Improving Discriminant Analysis Models
- Change fitting parameters to try to get a smaller model. This sometimes gives a model with more accuracy. For examples, see:
  - Select Appropriate Tree Depth
  - Prune a Classification Tree
  - Surrogate Splits
  - Ensemble Regularization
  - Bootstrap Aggregation (Bagging) of Regression Trees Using TreeBagger
  - Bootstrap Aggregation (Bagging) of Classification Trees Using TreeBagger
- Try a different algorithm. For applicable choices, see:
  - Characteristics of Classification Algorithms
  - Choose an Applicable Ensemble Aggregation Method

When satisfied with a model of some types, you can trim it using the appropriate compact function (compact for classification trees, compact for regression trees, compact for discriminant analysis, compact for naive Bayes, compact for SVM, compact for ECOC models, compact for classification ensembles, and compact for regression ensembles). compact removes training data and other properties not required for prediction, e.g., pruning information for decision trees, from the model to reduce memory consumption. Because*k*NN classification models require all of the training data to predict labels, you cannot reduce the size of a ClassificationKNN model.

Use Fitted Model for Predictions

To predict classification or regression response for most fitted models, use the predict method:

Ypredicted = predict(obj,Xnew)

- obj is the fitted model or fitted compact model.
- Xnew is the new input data.
- Ypredicted is the predicted response, either classification or regression

Example of Supervised Learning Algorithm

➔ Decision Tree Learning
➔ Bayesian Learning
➔ Learning using Artificial Neural Network
➔ Black Propagation

1.1.3.2 Semi-Supervised Method

Semi-supervised learning(SSL) is one of the artificial intelligence(AI) methods that have become popular in the last few months. Companies such as Google have been advancing the tools and frameworks relevant for building semi-supervised learning applications. Google Expander is a great example of a tool that reflects the advancements in semi-supervised learning applications. Conceptually, semi-supervised learning can be positioned halfway between unsupervised and supervised learning models. A semi-supervised learning problem starts with a series of labeled data points as well as some data point for which labels are not known. The goal of a semi-supervised model is to classify some of the unlabeled data using the labeled information set.

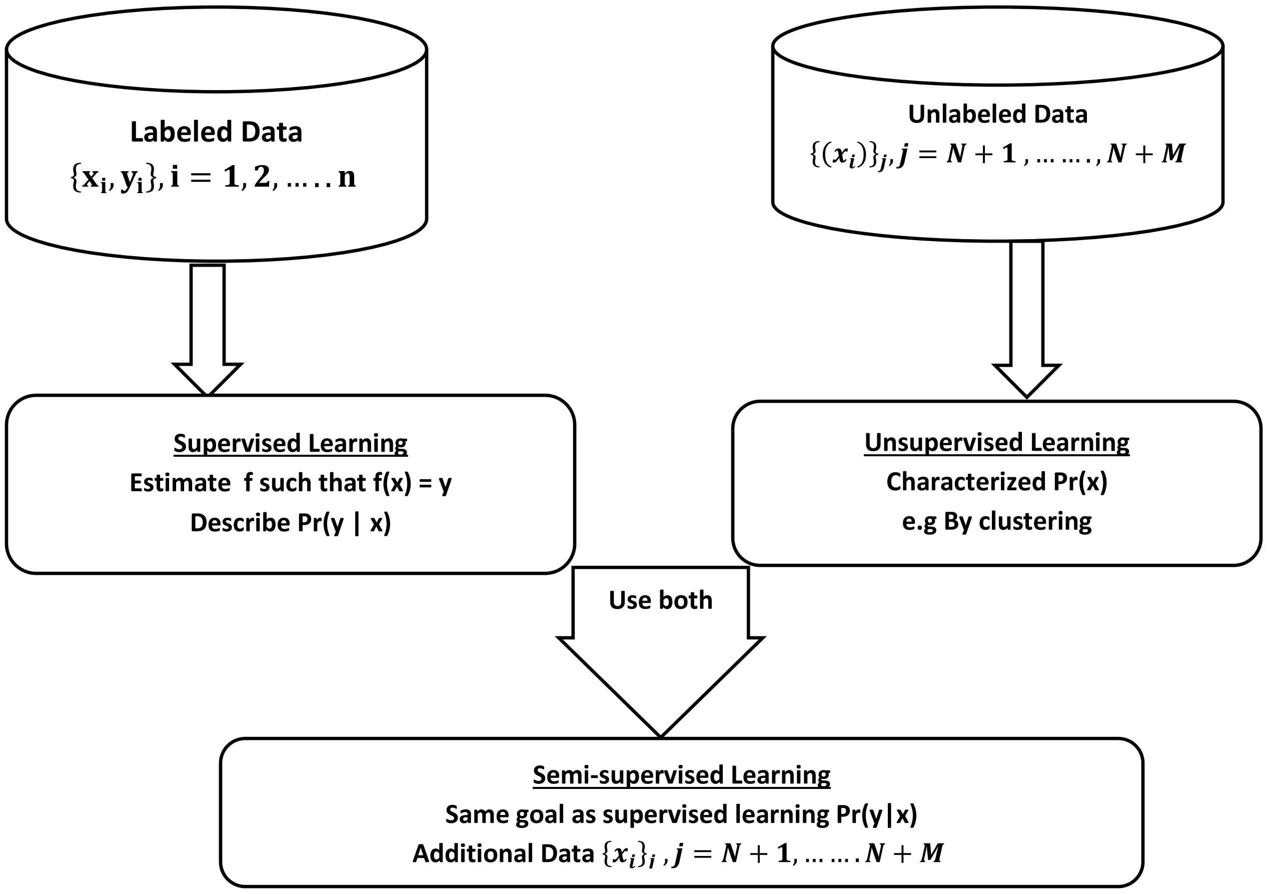

Labeled Data
$\{x_i, y_i\}, i = 1, 2, \ldots.. n$
Unlabeled Data
$\{(x_i)\}_j, j = N + 1, \ldots\ldots, N + M$
Supervised Learning
Estimate f such that f(x) = y
Describe Pr(y | x)
Unsupervised Learning
Characterized Pr(x)
e.g By clustering
Use both
Semi-supervised Learning
Same goal as supervised learning Pr(y|x)
Additional Data $\{x_i\}_i, j = N + 1, \ldots\ldots N + M$

FIG -2 FLOW DIAGRAM OF THE SUPERVISED LEARNING

1.1.3.3 Unsupervised Method

Unsupervised learning is the training of machine using information that is neither classified nor labeled and allowing the algorithm to act on that information without guidance. Here the task of machine is to group unsorted information according to similarities, patterns and differences without any prior training of data. Unlike supervised learning, no teacher is provided that means no training will be given to the machine. Therefore machine is restricted to find the hidden structure in unlabeled data by our-self. For instance, suppose it is given an image having both dogs and cats which have not seen ever. Thus the machine has no idea about the features of dogs and cat so we can't categorize it in dogs and cats. But it can categorize them according to their similarities, patterns, and differences i.e., we can easily categorize the above picture into two parts. First may contain all pics having dogs in it and second part may contain all pics having cats in it. Here you didn't learn anything before, means no training data or examples.

Unsupervised learning classified into two categories of algorithms:

- Clustering: A clustering problem is where you want to discover the inherent groupings in the data, such as grouping customers by purchasing behavior.
- Association: An association rule learning problem is where you want to discover rules that describe large portions of your data, such as people that buy X also tend to buy Y.

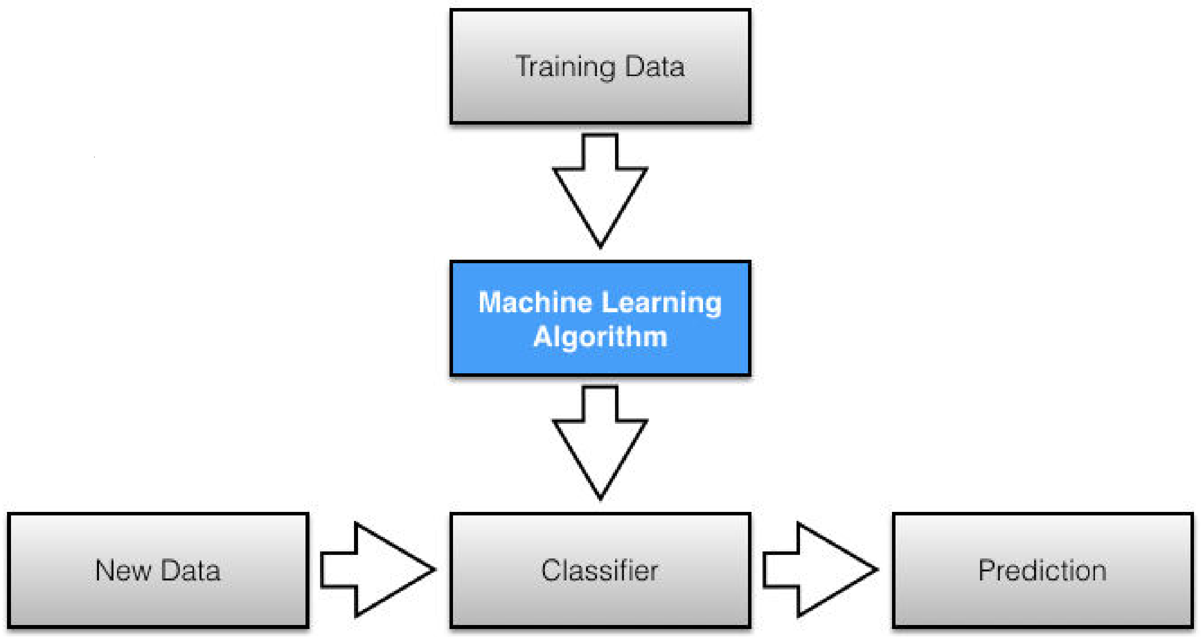
Let $X = (x_1, x_2, \dots \dots x_n)$ be a set of n example where $x_i \epsilon\ X.$ Typically it is assumed that the points are drawn from a common distribution on X . the goal of the unsupervised leanings to find interesting structure in the data X .

Why unsupervised learning is useful?

Deep learning is really about learning representations, which means learning intermediate concepts, features or latent variables that are useful to capture the statistical dependencies that we care about. These dependencies are those from input variables to output variables, in supervised learning, or between any subset of variables, in unsupervised learning. Often, supervised learning is used to teach the computer about intermediate concepts (such as categories) that are important to solve a particular task of interest. However, we see that our supervised deep networks discover meaningful intermediate concepts in their layers.

Unsupervised learning is similar except that we ask the model to capture all of the possible dependencies between all of the observed variables, with no distinction between inputs and outputs. To climb the AI ladder with supervised learning may require "teaching" the computer all the concepts that matter to us by showing tons of examples where these concepts occur. This is not how humans learn: yes, thanks to language we get some examples illustrating new named concepts that are given to us, but the bulk of what we observe does not come labeled, at least initially. Children don't have adults telling them what each pixel represents in every image they see, or what are the objects present in every image, what is the grammatical structure and the fine sense of every word in every sentence they hear. We extract most of the information from simple observation (possibly through an action-perception loop) and that is what unsupervised learning in principle does. The hope is that deep unsupervised learning will be able to discover (possibly with a little bit of help from the few labeled examples we can provide) all of the concepts and underlying causes that matter (some being explicitly labeled, some remaining unnamed) to explain what we see around us. So I believe it is essential for approaching AI to make progress in this direction.

If you think about it, scientists are doing unsupervised learning: observing the world, coming up with explanatory models, testing them by collecting more (targeted, though) observations, and continuously trying to improve our causal model of how the world around us works. We do get a bootstrap from education, though, so ideas like curriculum learning may be needed (as we see on some machine learning tasks).



FLOW DIAGRAM OF UNSUPERISED METHOD

Unsupervised Techniques can be divided into 2 types of techniques: -

I. Hierarchal – here the number of clusters are fixed. this fixed number is not known hierarchal clustering follows one of two approaches: **-**

I. Agglomerative methods start with each observation as a cluster and with each step combine observations to form clusters until there is only one large cluster.

II. Division methods begin with one large cluster and proceed to split into smaller clusters items that are most dissimilar

These are five ways of defining inter-clustering distance

a) Single linkage (based on the shortest distance between objects)
b) Complete linkage (based on the largest distance between object)
c) Average linkage (based on the average distance between object).
d) Ward's method
e) Centroid method

2. Non – Hierarchical :-

A non-hierarchical method generates a classification by partitioning a dataset, giving a set of (generally) non-overlapping groups having no hierarchical relationships between them. A systematic evaluation of all possible partitions is quite infeasible, and many different heuristics have thus been described to allow the identification of good, but possibly sub-optimal, partitions.

Non-hierarchical methods are generally much less demanding of computational resources than the hierarchic methods, typically $O(N)$ to $O(N^2)$ for *N* compounds, since only a single partition of the dataset has to be formed.

Three of the main categories of non-hierarchical method are single-pass, relocation and nearest neighbor:

- single-pass methods (e.g. Leader) produce clusters that are dependent upon the order in which the compounds are processed, and so will not be considered further;
- relocation methods, such as *k*-means, assign compounds to a user-defined number of seed clusters and then iteratively reassign compounds to see if better clusters result. Such methods are prone to reaching local optima rather than a global optimum, and it is generally not possible to determine when or whether the global optimum solution has been reached;
- nearest neighbor methods, such as the Jarvis-Patrick method, assign compounds to the same cluster as some number of their nearest neighbors. User-defined parameters determine how many nearest neighbors need to be considered, and the necessary level of similarity between nearest neighbor lists.

Other non-hierarchical methods are generally inappropriate for use on large, high-dimensional datasets such as those used in chemical applications.

Some important Application Area Of Unsupervised Learning :-

- Bio-Medical Image Processing
- Image Segmentation
- Computational Biology and Bio-Informatics
- QSAR and molecular modeling in Chemo informatics
- Market Research
- Customer Segmentation
- Social Network Analysis
- Sippy map optimization
- Crime Analysis

1.2 Introduction to Soft Computing

Soft computing is an emerging approach to computing which parallel the remarkable ability of the human mind to reason and learn in an environment of uncertainty and imprecision. Soft computing is based on some biological inspired methodologies such as genetics, evolution, ant's behaviors, particles swarming, human nervous systems, etc. Now, soft computing is the only solution when we don't have any mathematical modeling of problem solving (i.e., algorithm), need a solution to a complex problem in real time, easy to adapt with changed scenario and can be implemented with parallel computing. It has enormous applications in many application areas such as medical diagnosis, computer vision, hand written character recondition, pattern recognition, machine intelligence, weather forecasting, network optimization, VLSI design, etc.

Basically Soft computing is not a homogeneous body of concepts and techniques. rather it is a partnership of district methods that in own way or another confirm to its guiding impression and uncertainly to achieve tractability, robustness and low solutions cost . Components of soft Computing incudes: -

1. Artificial Neural Network
2. Fuzzy set
3. Genetic Algorithm

Some important Application Area Of Soft Computing :-

- Handwritten Character Recognition
- Image Processing
- Data Compression
- Architecture
- Decision Support System
- Power System

1.1.2 Artificial Neural Network (ANN)

A neural network is an inter connected assembly of simple processing elements, units or nodes, operating in parallel which can acquire, store, and utilize experiential knowledge. The key element of this paradigm is the novel structure of the information processing system. It is composed of a large number of highly interconnected processing elements (neurons) working in unison to solve specific problems. ANNs, like people, learn by example. An ANN is configured for a specific application, such as pattern recognition or data classification, through a learning process. Learning in biological systems involves adjustments to the synaptic connections that exist between the neurons. This is true of ANNs as well.

Neural networks take a different approach to problem solving than that of conventional computers. Conventional computers use an algorithmic approach i.e. the computer follows a set of instructions in order to solve a problem. Unless the specific steps that the computer needs to follow are known the computer cannot solve the problem. That restricts the problem solving capability of conventional computers to problems that we already understand and know how to solve. But computers would be so much more useful if they could do things that we don't exactly know how to do.

Neural networks process information in a similar way the human brain does. The network is composed of a large number of highly interconnected processing elements(neurons) working in parallel to solve a specific problem. Neural networks learn by example. They cannot be programmed to perform a specific task. The examples must be selected carefully otherwise useful time is wasted or even worse the network might be functioning incorrectly. The disadvantage is that because the network finds out how to solve the problem by itself, its operation can be unpredictable.

On the other hand, conventional computers use a cognitive approach to problem solving; the way the problem is to solved must be known and stated in small unambiguous instructions. These instructions are then converted to a high level language program and then into machine code that the computer can understand. These machines are totally predictable; if anything goes wrong is due to a software or hardware fault.

Neural networks and conventional algorithmic computers are not in competition but complement each other. There are tasks are more suited to an algorithmic approach like arithmetic operations and tasks that are more suited to neural networks. Even more, a large number of tasks, require systems that use a combination of the two approaches (normally a conventional computer is used to supervise the neural network) in order to perform at maximum efficiency.

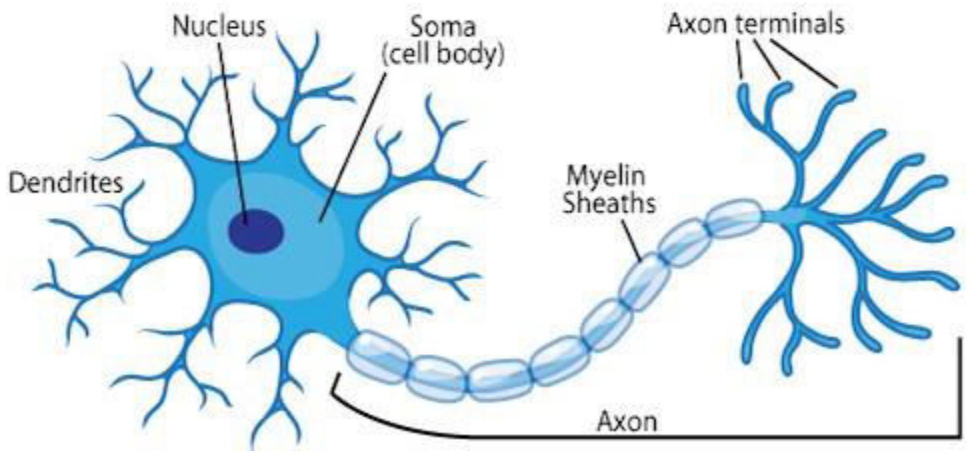


FIG-3 : SCHEMETIC DIAGRAM OF A BIOLOGIAL NEURON

Motivated by architecture and functionalities of biological neuron researchers have modeled the artificial neural network (ANN). we may identify the basic elements of the neuron model. these are stated below.

1. A set of synapses, each of which is characterized by a weight or strength of its own. specially, a signal x_i at the input of synapse j connected to neuron k is multiplied by the synaptic weight w_{kj}. It is important to make a note of the manner in which the subscripts weight w_{kj} are written. the first subscript refers to the neuron in question and the second subscript refers to the input end of the synapse to which the weight refers. the weight w_{kj} is positive if the associated synapse is excitatory. it is negative if the synapse is inhibitory.

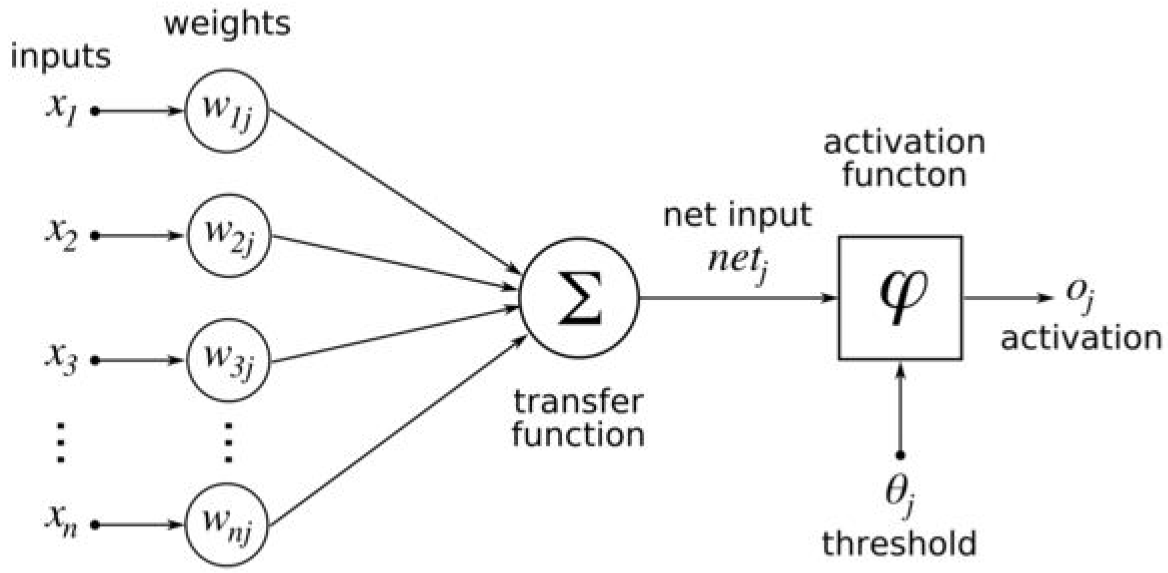
2. An adder for summing the input signals, weighted by the respective synapses of the neuron.

3. An activation function for limiting the amplitude of the output of a neuron. the activation function is also referred to in the literature as a squashing function in that it squashes (limits) the permissible amplitude range of the output signal to some finite value . typically, the normalized amplitude range of the output of a neuron is written as the closed unit interval [0,1] or alternatively [-1,1].

   In mathematical terms, we may describe a neuron k by writing the following pair of equations:

$$net_j = \sum \quad w_{ij}x_j$$

$$O_j = \varphi(net_j) \qquad \begin{array}{l} 1\,, if\ net_j > \theta_j \\ 0\,, elsewhere \end{array}$$



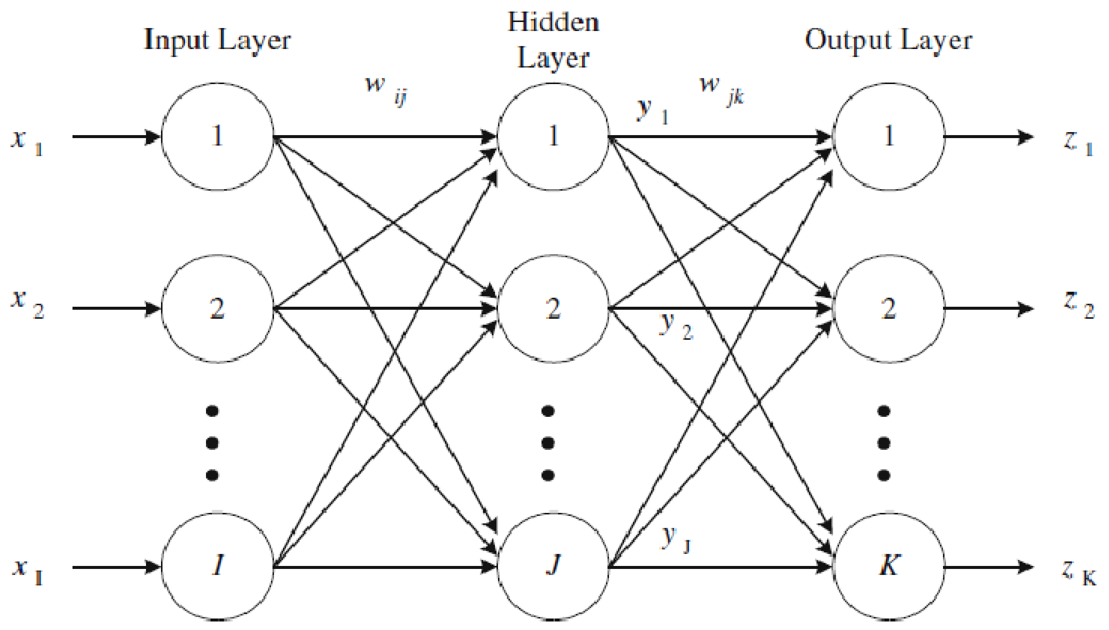
<u>Standard Structure of An Artificial Neural Network Timings</u>

Artificial neural network consists of three groups or layers of units: a layer of “input” units is connected to a layer of “hidden” units , which is connected to a layer of “output” units

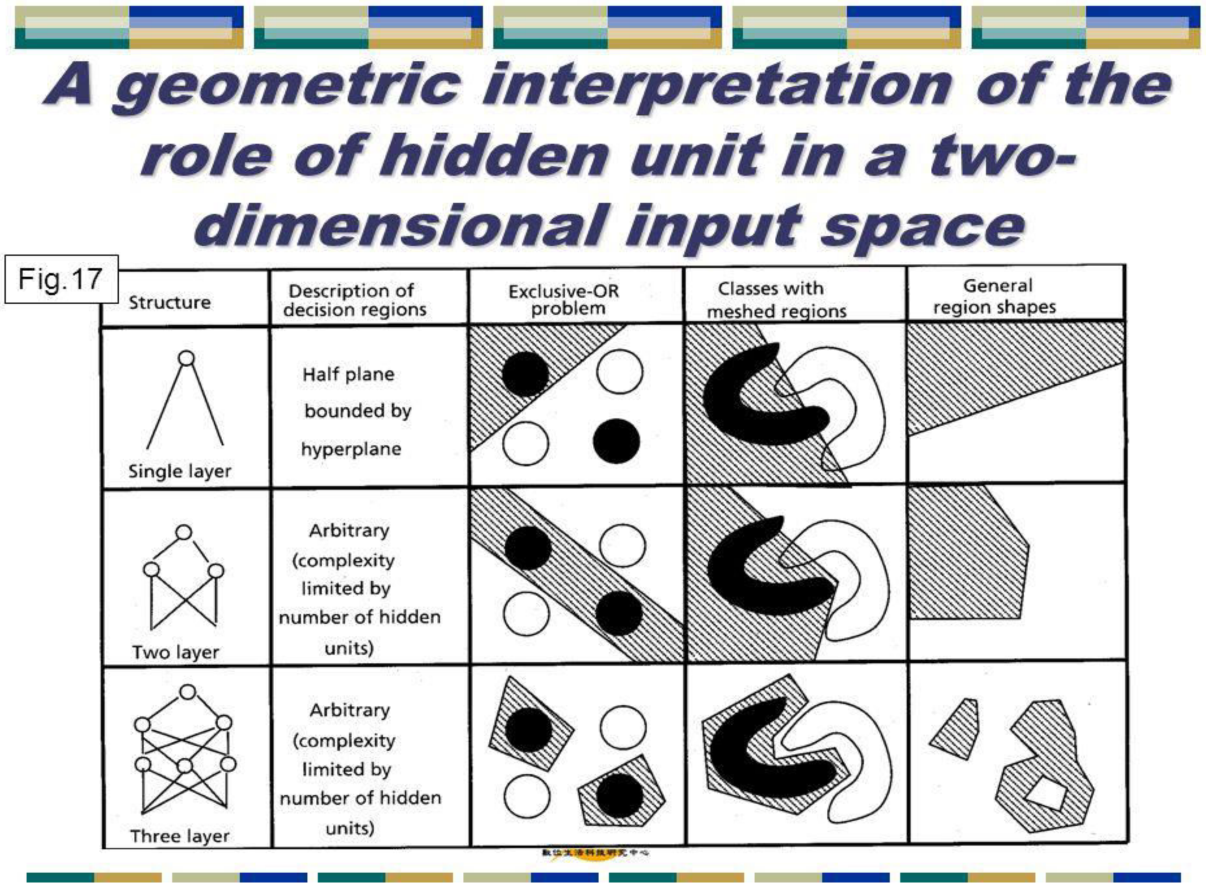
- **Input layer:** One neuron appears in the input layer for each predictor variable. In the case of categorical variables, N-1 neurons are used where N is the number of categories. The input neurons standardizes the value ranges by subtracting the median and dividing by the interquartile range. The input neurons then feed the values to each of the neurons in the hidden layer.

- **Hidden layer:** This layer has a variable number of neurons (determined by the training process). Each neuron consists of a radial basis function centered on a point with as many dimensions as predictor variables. The spread (radius) of the RBF function may be different for each dimension. The centers and spreads are determined by training. When presented with the x vector of input values from the input layer, a hidden neuron computes the Euclidean distance of the test case from the neuron's center point and then applies the RBF kernel function to this distance using the spread values. The resulting value is passed to the summation layer.

- **Output layer:** The value coming out of a neuron in the hidden layer is multiplied by a weight associated with the neuron and adds to the weighted values of other neurons. This sum becomes the output. For classification problems, one output is produced (with a separate set of weights and summation unit) for each target category. The value output for a category is the probability that the case being evaluated has that category



A geometric information (adopted and modified) can help explicate the role of hidden units (with the threshold activation function) . Each unit in the first hidden layer forms a hyperplane in the pattern space; boundaries between pattern classes can be approximated by hyperplane. A unit in the second layer forms a hyper region from on the hyperplane. A unit in the second hidden layer by performing logical OR operations.


A geometric interpretation of the role of hidden unit in a two-dimensional input space
Structure
Description of decision regions
Exclusive-OR problem
Classes with meshed regions
General region shapes
Single layer
Half plane bounded by hyperplane
Two layer
Arbitrary (complexity limited by number of hidden units)
Three layer
Arbitrary (complexity limited by number of hidden units)
Fig.17

ANN can be viewed as weighted directed graphs in which artificial neurons are nodes and directed edges (with weights) are connections between neuron outputs and neuron inputs. Based on the connection pattern (architecture) , ANN can be grouped into two categories :-

- Feed-Forward network, in which graphs have no loops
- Recurrent (or feedback) networks, in which loops occur because of feedback connections .

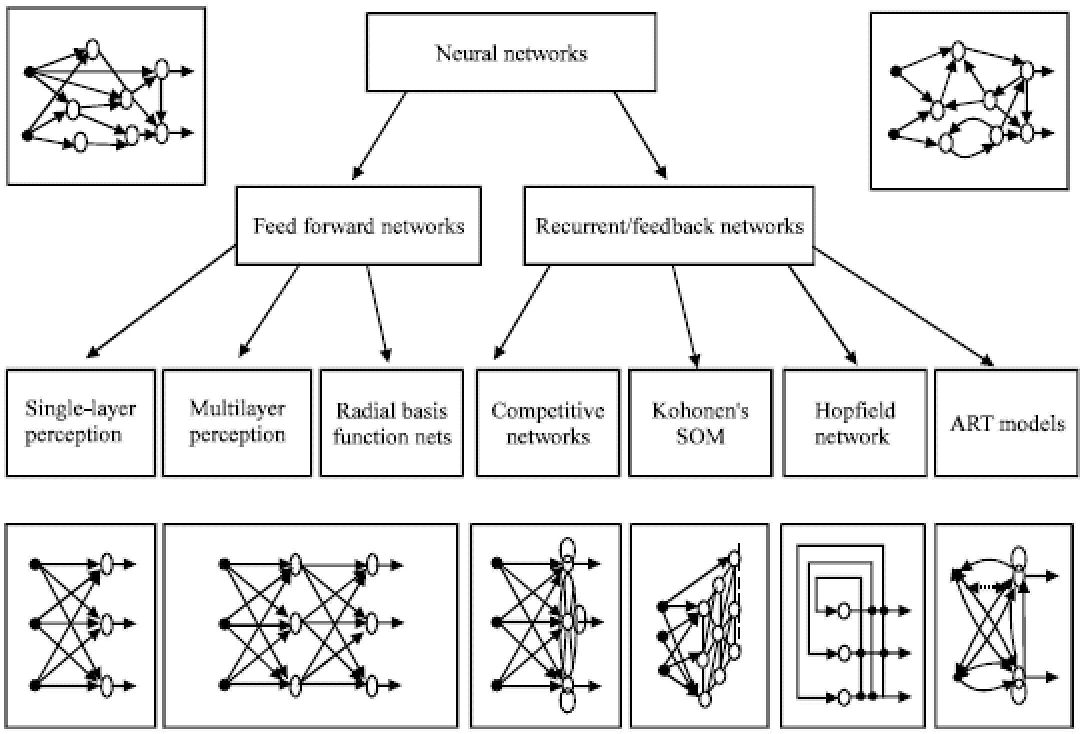


FIG-4 : TAXONOMY OF NEURAL NETS

Applications of ANN:

- Classification:
  - In Marketing: - consumer spending pattern classification
  - In defense: - radar and sonar image classification
  - In agriculture & finishing: - fruit and catch grading
    Medical diagnosis.
- Recognition and identification
  - In general computing and telecommunications: - speech, vision and handwriting recognition.
  - In finance: - signature verification and bank verification
- Assessment
  - In engineering: - product inspection monitoring control
  - In defense: - target tracking
  - In security: - motion detection, surveillance image analysis and fingerprint matching
- Forecasting and prediction
  - In finance: - foreign exchange rate and stock forecasting
  - In agriculture: - crop yield forecasting
  - In marketing: - sales forecasting
  - In meteorology: - weather prediction

1.2.2 Fuzzy logic

The term **fuzzy** refers to things which are not clear or are vague. In the real world many times we encounter a situation when we can't determine whether the state is true or false, their fuzzy logic provides a very valuable flexibility for reasoning. In this way, we can consider the inaccuracies and uncertainties of any situation.
In Boolean system truth value, 1.0 represents absolute truth value and 0.0 represents absolute false value. But in the fuzzy system, there is no logic for absolute truth and absolute false value. But in fuzzy logic, there is intermediate value too present which is partially true and partially false.

- It can be implemented in systems with various sizes and capabilities ranging from small micro-controllers to large, networked, workstation-based control systems.
- It can be implemented in hardware, software, or a combination of both.
- Among various combinations of methodologies in soft computing, the one that has highest visibility at this juncture is that of fuzzy logic and neurocomputing, leading to neuro-fuzzy systems. Within fuzzy logic, such systems play a particularly important role in the induction of rules from observations. An effective method developed by Dr. Roger Jang for this purpose is called ANFIS (Adaptive Neuro-Fuzzy Inference System). This method is an important component of the toolbox.
- Fuzzy logic is all about the relative importance of precision: How important is it to be exactly right when a rough answer will do?
- You can use Fuzzy Logic Toolbox software with MATLAB® technical computing software as a tool for solving problems with fuzzy logic. Fuzzy logic is a fascinating area of research because it does a good job of trading off between significance and precision — something that humans have been managing for a very long time.
- In this sense, fuzzy logic is both old and new because, although the modern and methodical science of fuzzy logic is still young, the concepts of fuzzy logic relies on age-old skills of human reasoning.

Crisp vs Fuzzy Set:-

Among various combinations of methodologies in soft computing, the one that has highest visibility at this juncture is that of fuzzy logic and neurocomputing, leading to neuro-fuzzy systems. Within fuzzy logic, such systems play a particularly important role in the induction of rules from observations. An effective method developed by Dr. Roger Jang for this purpose is called ANFIS (Adaptive Neuro-Fuzzy Inference System). This method is an important component of the toolbox.

Fuzzy logic is all about the relative importance of precision: How important is it to be exactly right when a rough answer will do?

You can use Fuzzy Logic Toolbox software with MATLAB® technical computing software as a tool for solving problems with fuzzy logic. Fuzzy logic is a fascinating area of research because it does a good job of trading off between significance and precision — something that humans have been managing for a very long time.

In this sense, fuzzy logic is both old and new because, although the modern and methodical science of fuzzy logic is still young, the concepts of fuzzy logic relies on age-old skills of human reasoning.in the fuzzy theory , fuzzy set A of universe of discourse X is defined by function

$$\mu_A(x): X \rightarrow [0,1], where\ \mu_A(x) = 1\ if\ x\ is\ totally\ in\ A$$

$$\mu_A(x) = 0 \; if \; x \; is \; not \; in \; A;$$

$$0 < \mu_A(x) < 1 \quad if \; x \; is \; partly \; in \; A$$

Let X be the universe of discourse and its elements be denoted as x. In the classical set theory, crisp set A of X is defined as function $f_A(x)$ called the characteristic function of A .

$$f_A(x) : X \rightarrow \{0,1\} \; , where \; f_A(x) \; = \{1, \; if \; x \; \in A\}$$

$$0, if x \varphi A$$

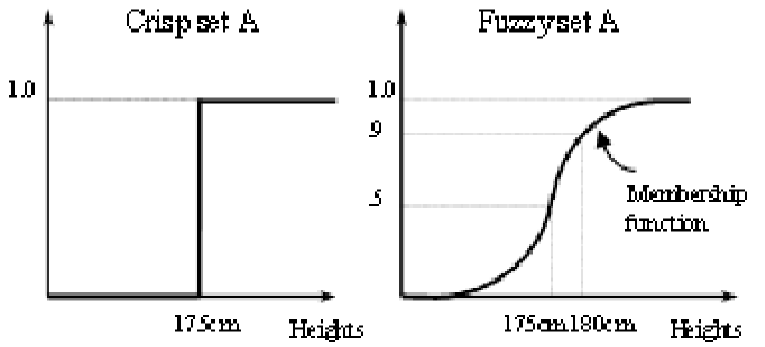


FIG-5 A Comparison of Membership Function Crisp vs Fuzzy set

How does Fuzzy Logic Works?

Fuzzy logic works on the concept on deciding the output on the basis of assumptions. It works on the basis of sets. Each set represents some linguistic variable defining the possible state of the output. Each possible state of the input and the degrees of change of the state are a part of the set, depending upon which the output is predicted. It basically works on the principle of If-else-the, i.e. If A AND B Then Z. Suppose we want to control a system where the output can be anywhere in the set X, with a generic value x, such that x belongs to X. Consider a particular set A which is a subset of X such that all members of A belong to the interval 0 and 1. The set A is known as fuzzy set and the value of $f_A(x)$at x denotes the degree of membership of x in that set. The output is decided based on the degree of membership of x in the set. This assigning of membership depends on the assumption of the outputs depending on the inputs and the rate of change of the inputs.

These fuzzy sets are represented graphically using membership functions and the output is decided based on the degree of membership in each part of the function. The membership of the sets is decided by the IF-Else logic.

Generally the variables of the set are the state of the inputs and the degrees of changes of the input and the membership of the output depends on the logic of AND operation of the state of the input and the rate of change of the input. For a multi input system, the variables can also be the different inputs and the output can be the possible result of the AND operation between the variables.

The above steps are summarized into below Algorithm that has mainly three step process .

I. **Fuzzification**

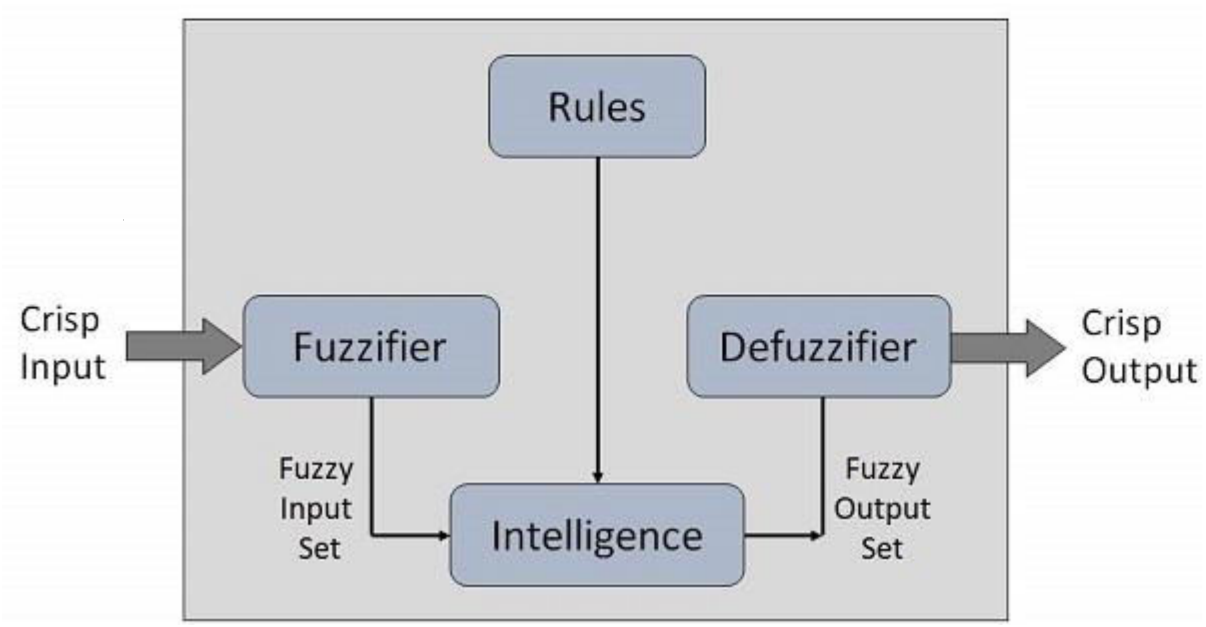
   Firstly, a crisp set of input data are gathered and converted to a fuzzy set using fuzzy linguistic variables, fuzzy linguistic terms and membership functions.

II. **Evaluation of Rules**

   Afterwards, an inference is made based on a set of rule. this is the application of the fuzzy logic rules.

III. **Diffuzzification**

   Lastly , the resulting fuzzy output is mapped to a crisp output using the membership functions , in the diffuzification step**.**



Algorithm : Fuzzy Logic Algorithm

1. Define the linguistic variables and terms(initialization)
2. Construct the membership functions. (initialization)
3. Construct the rule base (initialization)
4. Convert crisp input data to fuzzy values using the membership functions (Fuzzification).
5. Evalute the rules in the rule base(inference)
6. Combine the results of each rule(inference).
7. Convert the output data to non-fuzzy values.

1.2.3 Genetic Algorithm

The genetic algorithm is a method for solving both constrained and unconstrained optimization problems that is based on natural selection, the process that drives biological evolution. The genetic algorithm repeatedly modifies a population of individual solutions.

At each step, the genetic algorithm selects individuals at random from the current population to be parents and uses them to produce the children for the next generation. Over successive generations, the population "evolves" toward an optimal solution. You can apply the genetic algorithm to solve a variety of optimization problems that are not well suited for standard optimization algorithms, including problems in which the objective function is discontinuous, nondifferentiable, stochastic, or highly nonlinear. The genetic algorithm can address problems of mixed integer programming, where some components are restricted to be integer-valued.
The genetic algorithm uses three main types of rules at each step to create the next generation from the current population:

- Selection rules select the individuals, called parents, that contribute to the population at the next generation.
- Crossover rules combine two parents to form children for the next generation.
- Mutation rules apply random changes to individual parents to form children.

How the Genetic Algorithm Works ?

OUTLINE OF THE ALGORITHM

The following outline summarizes how the genetic algorithm works:

1. The algorithm begins by creating a random initial population.
2. The algorithm then creates a sequence of new populations. At each step, the algorithm uses the individuals in the current generation to create the next population. To create the new population, the algorithm performs the following steps:
   a. Scores each member of the current population by computing its fitness value. These values are called the raw fitness scores.
   b. Scales the raw fitness scores to convert them into a more usable range of values. These scaled values are called expectation values.
   c. Selects members, called parents, based on their expectation.
   d. Some of the individuals in the current population that have lower fitness are chosen as *elite*. These elite individuals are passed to the next population.
   e. Produces children from the parents. Children are produced either by making random changes to a single parent—*mutation*—or by combining the vector entries of a pair of parents—*crossover*.
   f. Replaces the current population with the children to form the next generation.
3. The algorithm stops when one of the stopping criteria is met. See Stopping Conditions for the Algorithm.

Application off Genetic Algorithm :-

- Dynamic process control
- Introduction of rule optimization
- Discovering new connectivity topologies
- Pattern recognition
- Scheduling
- Transportation
- Layout and circuit design
- Telecommunication
- Graph-based problems

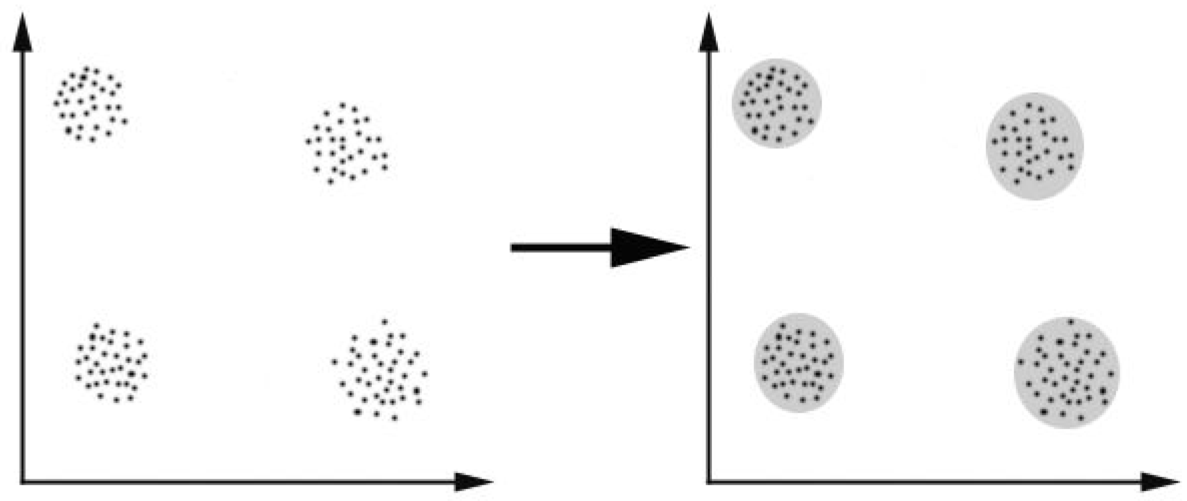
Applications:

- **Image processing, segmentation and analysis**
  Pattern recognition is used to give human recognition intelligence to machine which is required in image processing.
- **Computer vision**
  Pattern recognition is used to extract meaningful features from given image/video samples and is used in computer vision for various applications like biological and biomedical imaging.
- **Seismic analysis**
  Pattern recognition approach is used for the discovery, imaging and interpretation of temporal patterns in seismic array recordings. Statistical pattern recognition is implemented and used in different types of seismic analysis models.
- **Radar signal classification/analysis**
  Pattern recognition and Signal processing methods are used in various applications of radar signal classifications like AP mine detection and identification.
- **Speech recognition**
  The greatest success in speech recognition has been obtained using pattern recognition paradigms. It is used in various algorithms of speech recognition which tries to avoid the problems of using a phoneme level of description and treats larger units such as words as pattern
- **Finger print identification**
  The fingerprint recognition technique is a dominant technology in the biometric market. A number of recognition methods have been used to perform fingerprint matching out of which pattern recognition approaches is widely used.

Chapter 2 : CLUSTERING TECHNIQUES

2.1 Introduction to Clustering

Clustering can be considered the most important *unsupervised learning* problem; so, as every other problem of this kind, it deals with finding a *structure* in a collection of unlabeled data.
A loose definition of clustering could be "the process of organizing objects into groups whose members are similar in some way".
A *cluster* is therefore a collection of objects which are "similar" between them and are "dissimilar" to the objects belonging to other clusters.
We can show this with a simple graphical example:



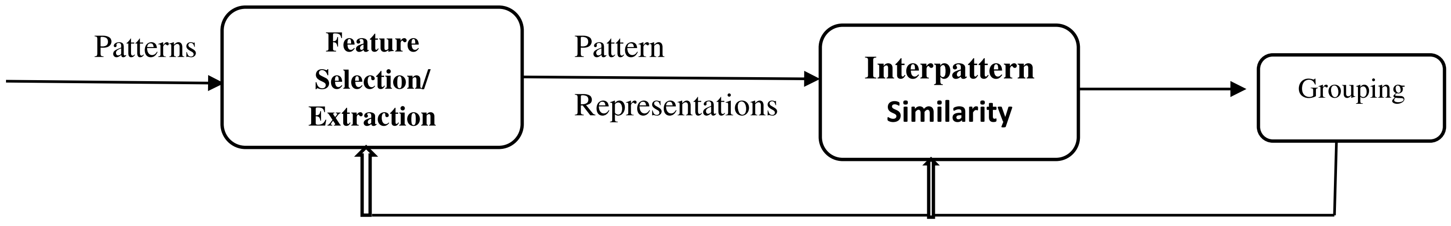
In this case we easily identify the 4 clusters into which the data can be divided; the similarity criterion is *distance*: two or more objects belong to the same cluster if they are "close" according to a given distance (in this case geometrical distance). This is called *distance-based-clustering*.

Another kind of clustering is *conceptual clustering*: two or more objects belong to the same cluster if this one defines a concept *common* to all that objects. In other words, objects are grouped according to their fit to descriptive concepts, not according to simple similarity measures.

Set $S = X_1, X_2, \ldots . X_n$ and K clusters are represented by $C_1, C_2, \ldots . C_k$

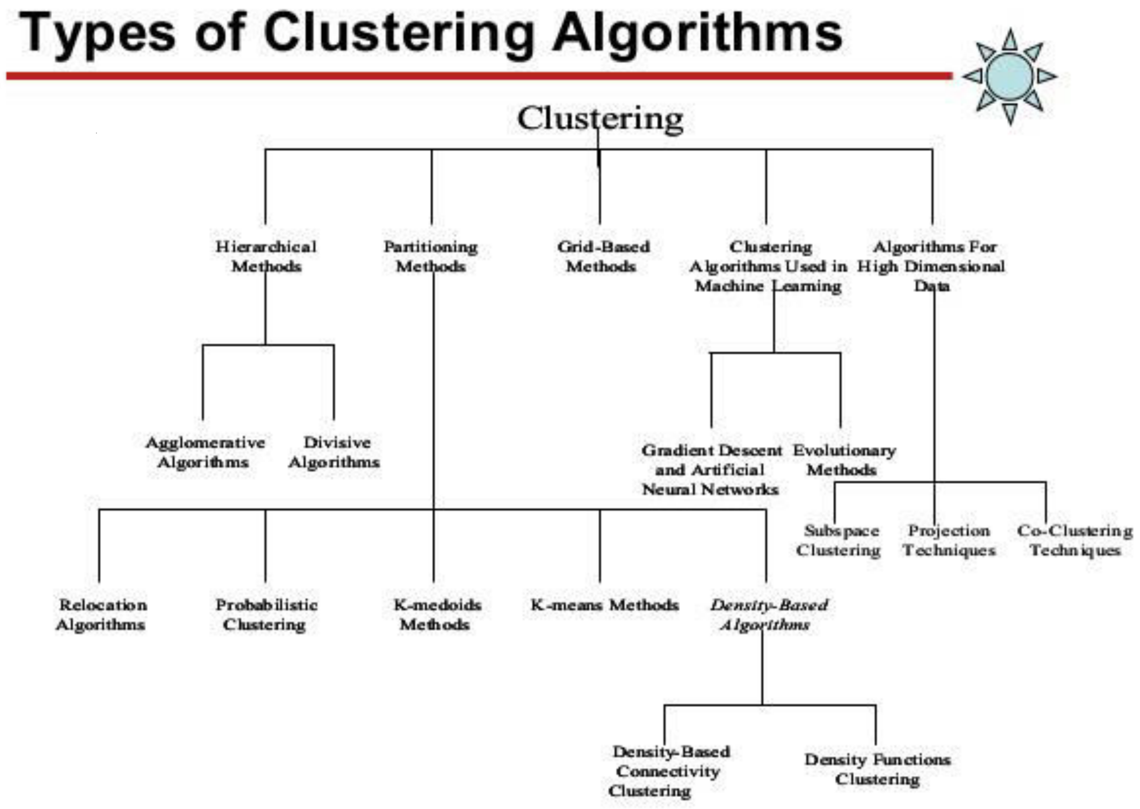
1. $C_i \neq \emptyset$ for $i = 1,2, \ldots . K$
2. $C_i \cap C_j = \emptyset$ for $i \neq j$
3. $\cup_{i=1}^{K} C_i = S$ where $\emptyset$ represents null set .

A clustering algorithm attempts to find natural groups of components (or data) based on some similarity . the output from a clustering algorithm is a basically a statistical description of the cluster centroids with the number of components in each cluster .



Feedback loop

Different stages of Clustering Technique



2.1 <u>Clustering Techniques</u>

2.2.1 <u>Hierarchical Clustering</u>

Hierarchical clustering, also known as hierarchical cluster analysis, is an algorithm that groups similar objects into groups called clusters. The endpoint is a set of clusters, where each cluster is distinct from each other cluster, and the objects within each cluster are broadly similar to each other. This clustering technique is divided into two types:

- Agglomerative
- Divisive

2.2.1.1 Agglomerative Nesting

In this technique, initially each data point is considered as an individual cluster. At each iteration, the similar clusters merge with other clusters until one cluster or K clusters are formed. The basic process of Agglomerative Nesting below --------

- Start by assigning each item to a cluster, so that if you have N items , so that if you have N items, you now have N clusters , each containing just one item. Let the distances between the clusters the same as the distances between the items they certain.

- Find the closest pair of clusters and merge them into a single cluster, so that now you have one cluster less.

- Compute distances (similarities) between the new cluster and each of the old clusters.

- Repeat steps 2 and 3 until all items are clustered into a single cluster of size N.

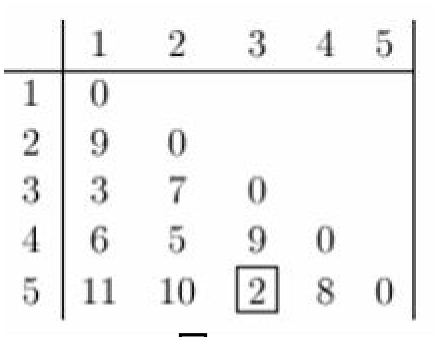
2.2.1.1.1 Single Linkage Method

The dissimilarly between two clusters is the minimum dissimilarity between members of the two clusters. this method produces long claims which from loose, straggly clusters. Step by step process is discussed below:

1) By placing each pattern in its own cluster, a list of inter pattern distances for all distinct unordered pairs of patterns is constructed and sort this list in according order.
2) Step through the sorted list of distances, forming for each distinct similarly value d_k are connected by a graph edge. if all the patterns are members of a connected graph, stop. otherwise, repeat this step.

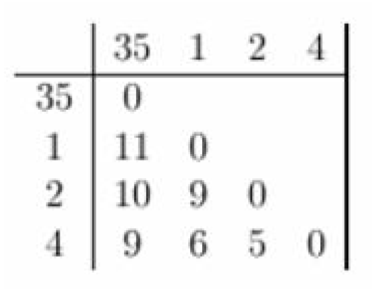
$$d_k = sim\ (c_i, c_j) = \dim(c_i, c_j) = \min_{x \epsilon c_i, y \epsilon c_j} ||x - y||$$

3) The output of the algorithm is a nested hierarchy of graphs which can be cut at a desired dissimilarity level forming a partition (clustering) identified by simply connected components in the corresponding graph .



	1	2	3	4	5
1	0				
2	9	0			
3	3	7	0		
4	6	5	9	0	
5	11	10	$\boxed{2}$	8	0

$$\Downarrow$$



	35	1	2	4
35	0			
1	11	0		
2	10	9	0	
4	9	6	5	0

$$\Downarrow$$

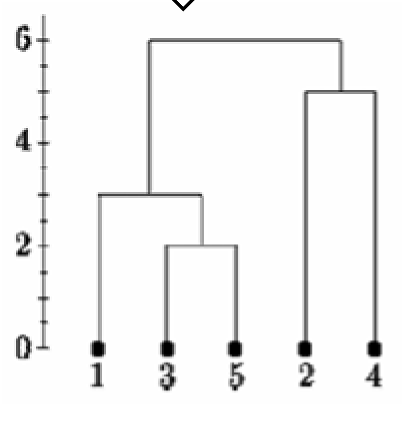
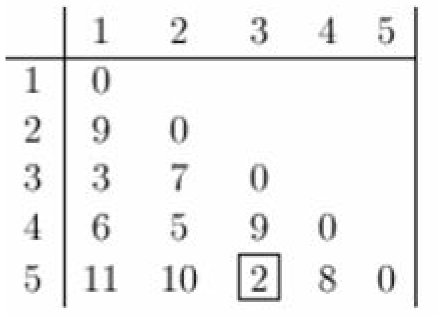


FIG SINGLE LINKAGE

2.2.1.1.2 Complete Linkage Method

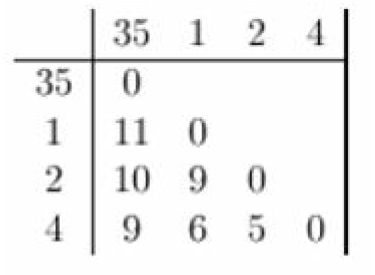
Clustering starts by computing a distance between every pair of units that you want to cluster. A distance matrix will be symmetric (because the distance between x and y is the same as the distance between y and x) and will have zeroes on the diagonal (because every item is distance zero from itself). The table below is an example of a distance matrix. Only the lower triangle is shown, because the upper triangle can be filled in by reflection.



	1	2	3	4	5
1	0				
2	9	0			
3	3	7	0		
4	6	5	9	0	
5	11	10	$\boxed{2}$	8	0

Now lets start clustering. The smallest distance is between three and five and they get linked up or merged first into a the cluster '35'.

To obtain the new distance matrix, we need to remove the 3 and 5 entries, and replace it by an entry "35" . Since we are using complete linkage clustering, the distance between "35" and every other item is the maximum of the distance between this item and 3 and this item and 5. For example, d(1,3)= 3 and d(1,5)=11. So, D(1,"35")=11. This gives us the new distance matrix. The items with the smallest distance get clustered next. This will be 2 and 4.



	35	1	2	4
35	0			
1	11	0		
2	10	9	0	
4	9	6	5	0

Continuing in this way, after 6 steps, everything is clustered. This is summarized below. On this plot, the y-axis shows the distance between the objects at the time they were clustered. This is called the cluster height. Different visualizations use different measures of cluster height.

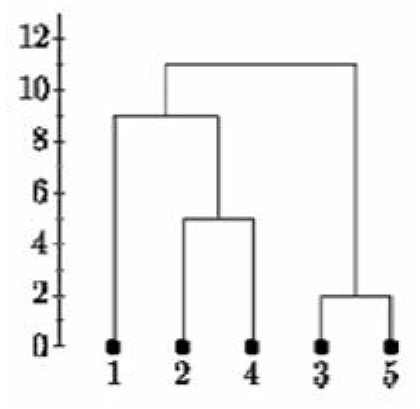


FIG Complete Linkage.

2.2.1.1.3 Average Linkage Method

Average linkage clustering, the distance between two clusters is defined as the average of distances between all pairs of objects, where each pair is made up of one object from each group.

$$d_k = sim(c_i, c_j) = d_{ave}(c_i, c_j) = \left(\frac{1}{n_i n_j}\right) \sum \sum ||x - y||$$

$$x \epsilon c_i \; x \epsilon c_j$$

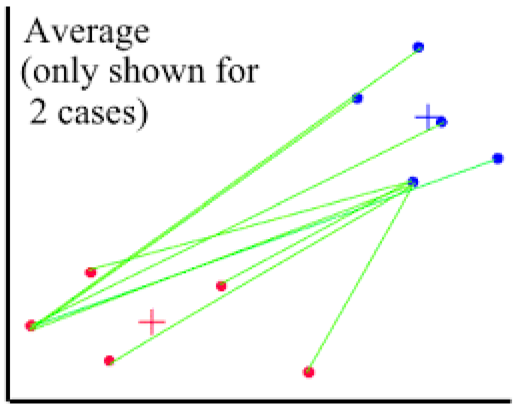


FIG Graphical Example of Average Linkage Method

2.2.1.2 Divisive Analysis (DIANA)

The traffic and transportation facility of any country significantly defines its development. The developing countries like India must have a well defined LOS analysis procedure to develop a good road network, because, it is very essential for the planning, design of transportation system and allocation of limited resources to the competing projects. DIANA is a hierarchical clustering technique, but its main difference with the agglomerative method (AGNES) is that it constructs the hierarchy in the inverse order. Initially (Step 0), there is one large cluster consisting of all n objects. At each subsequent step, the largest available cluster is split into two clusters until finally all clusters, comprise of single objects. Thus, the hierarchy is built in n-1 steps.

In the first step of an agglomerative method, all possible fusions of two objects are considered leading to $n(n-1)/2$ combinations. In the divisive method based on the same principle, there are possibilities to split the data into two clusters. This number is considerably larger than that in the case of an agglomerative method.

To avoid considering all possibilities, the algorithm proceeds as follows.

1. Find the object , which has the highest average dissimilarity to all other objects. This object initiates new cluster- a sort of a splinter group .
2. For each object i outside the splinter group compute .
3. $D_i = \left[average\ d(i,j) \neq R_{splintergroup} \right] - average\ d(i,j) \in R\ splintergroup$
4. Find an object h for which the difference D_h is the largest . if $D_h\ is\ positive$ then h is, on the average close to the $spplintergroup$
5. Repeat Step 2 and 3 until all difference D_h are negative. The data set is then split into two cluster .
6. Select the cluster with the largest diameter .the diameter of a cluster is the largest dissimilarly between any two of its objects. Then divide the cluster, following step 1-4
7. Repeat step 5 until all clusters contain only a single object .

2.2.1.3 Balanced iterative Reducing and Clustering using Hierarchies (BIRCH)

BIRCH, which stands for Balanced Iterative Reducing and Clustering using Hierarchies, was developed in 1996 by Tian Zhang, Raghu Ramakrishnan, and Miron Livny.[1] BIRCH is especially appropriate for very large data sets, or for streaming data, because of its ability to find a good clustering solution with only a single scan of the data. Optionally, the algorithm can make further scans through the data to improve the clustering quality. BIRCH handles large data sets with a time complexity and space efficiency that is superior to other algorithms, according to the authors. The BIRCH algorithm is a updated modified version of agglomerative nesting that overcomes the two difficulties of the previous step . it stores summary information about candidate clusters in a dynamic tree data structure . this tree hierarchically organizes the clusters represented at the leaf nodes . the tree can be result when a threshold specifying cluster size is updated manually or when memory constraints force a change in this threshold. This algorithm has a time complexity linear in the number of instances . it is designed for clustering a large amount of numerical data by integration of hierarchical clustering and other clustering methods such as iterative partitioning . BIRCH introduces two concepts, clustering feature and clustering feature tree (CF tree) , which are used to summarize cluster representations .

A clustering feature (CF) is a three dimensional vector summarizing information about clustering of objects. Given n d-dimensional objects or points in a cluster,$\{x_i\}$, the the CF of the cluster is definded as

$$CF = \{n, LS, SS\}$$

Where n is the number of point in the cluster, LS is the linear sum of the n points (i.e .$\sum_{i=1ton} X_i$) and SS is the square sum of the data points (i.e $\sum_{i=1ton} x_i^2$)

CF Tree:

- A non leaf node represents a cluster made up of all sub clusters represented by its entries.
- A leaf node also represents a cluster made up all sub clusters represented by its entries. the diameter radius of any entry has to be less than the threshold T.
- The tree size is a function of T. the larger T is the smaller the tree is.
- A CF tree will be built dynamically as new data objects are interested.
- B and L are determined by page size P.

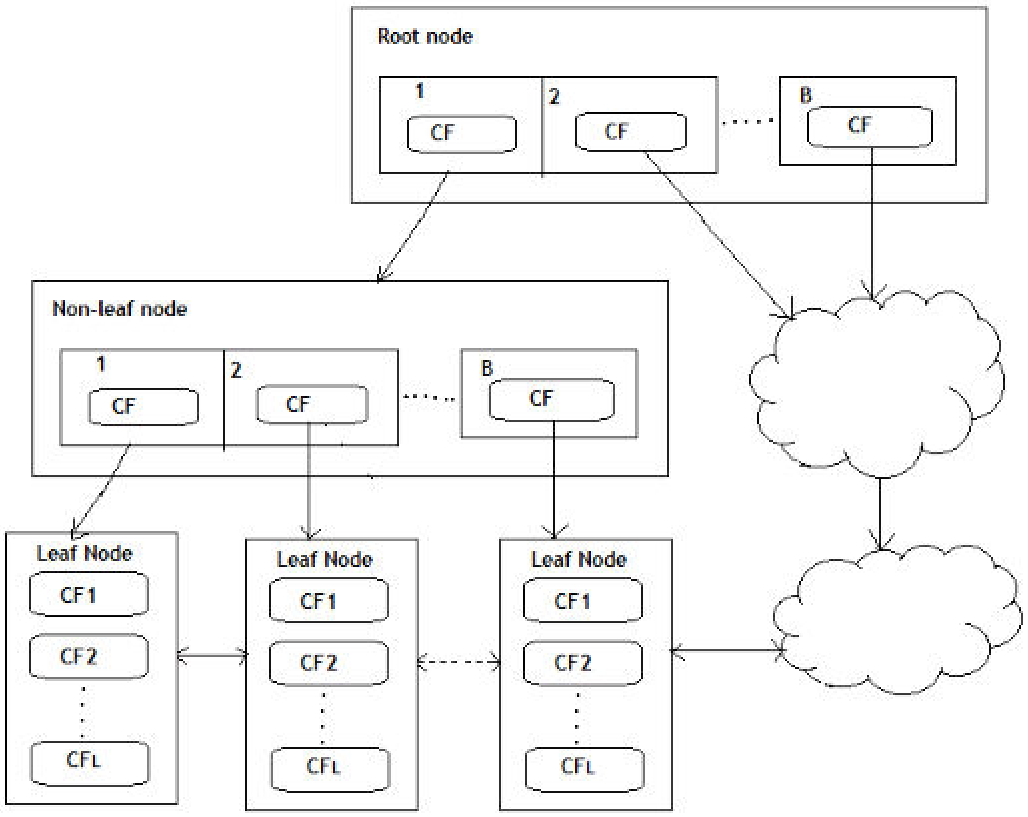


Fig: A CF tree structure

Phases of BIRCH: -

- Phase 1: Load data into memory

Scan DB and load data into memory by building a CF tree. If memory is exhausted rebuild the tree from the leaf node.

- Phase 2: Condense data

Resize the data set by building a smaller CF tree

Remove more outliers

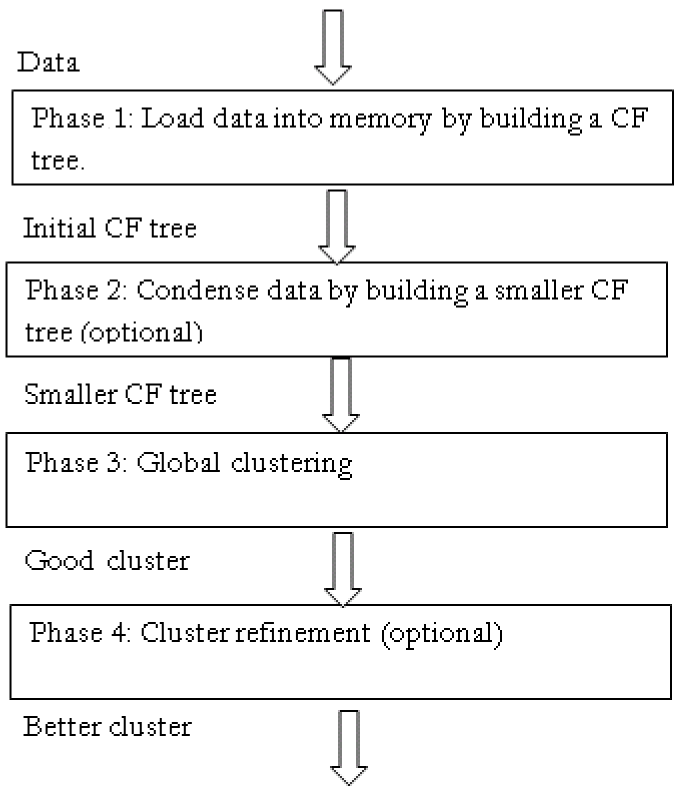
Condensing is optional

- Phase 3: Global clustering

Use existing clustering algorithm (e.g. KMEANS, HC) on CF entries

- Phase 4: Cluster refining

Refining is optional


Data
Phase 1: Load data into memory by building a CF tree.
Initial CF tree
Phase 2: Condense data by building a smaller CF tree (optional)
Smaller CF tree
Phase 3: Global clustering
Good cluster
Phase 4: Cluster refinement (optional)
Better cluster

2.2.1.4 Clustering Using Representatives (CURE)

The CURE (Clustering Using Representatives) Algorithm is large scale clustering algorithm in the point assignment class which assumes Euclidean space. It does not assume anything about the shape of clusters; they need not be normally distributed, and can even have strange bends, S-shapes, or even rings. Instead of representing clusters by their centroid, it uses a collection of representative points, as the name implies.

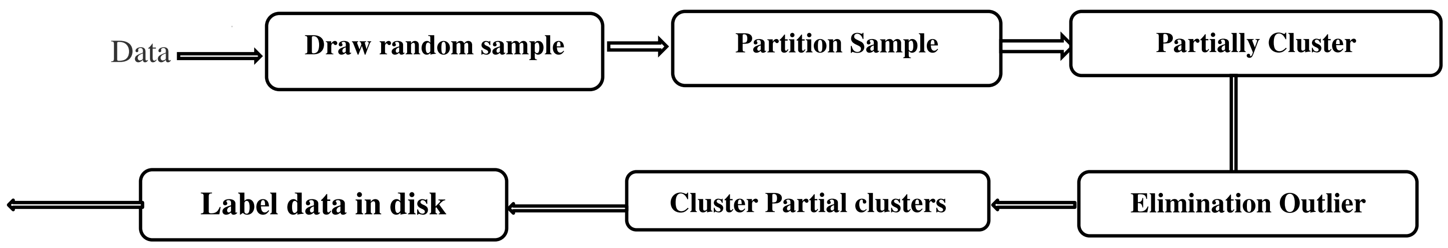
The CURE algorithm is divided into phases:

1. Initialization in CURE
2. Completion of the CURE Algorithm

Initialization in CURE:

1. Take a small sample of the data and cluster it in main memory. In principle, any clustering method could be used, but as CURE is designed to handle oddly shaped clusters, it is often advisable to use a hierarchical method in which clusters are merged when they have a close pair of points.
2. Select a small set of points from each cluster to be representative points. These points should be chosen to be as far from one another as possible, using the K-means method.

3. Move each of the representative points a fixed fraction of the distance between its location and the centroid of its cluster. Perhaps 20% is a good fraction to choose. Note that this step requires a Euclidean space, since otherwise, there might not be any notion of a line between two points.



<u>CURE Clustering Procedure:</u>

1> It is similar to hierarchical clustering approach. but it uses sample point variant as the cluster representative rather than every point in the cluster.

2> First set a target sample number c. than we try to select c well scattered sample point from the cluster.

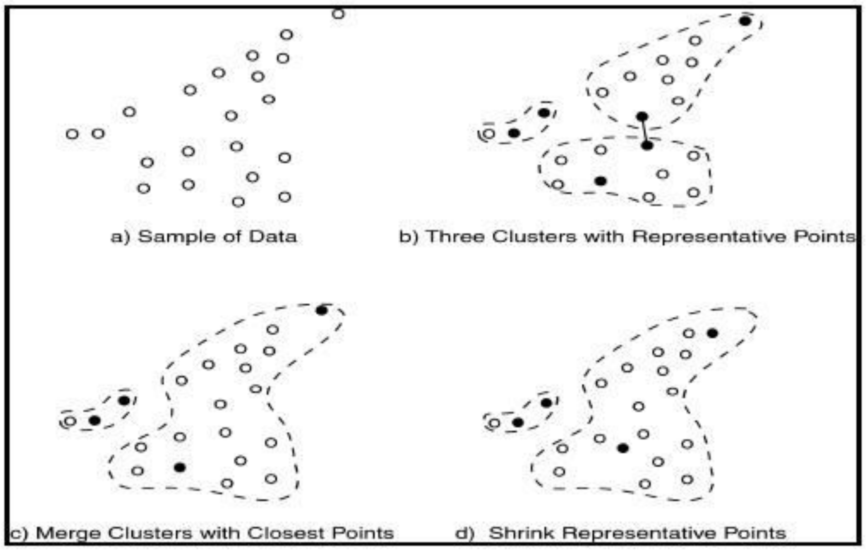
3> The chosen scattered points are shrunk toward the centroid in a fraction of α where $0 \leq \alpha \leq 1$

4> These points are used as representative of clusters answer will be used as the point in $d_{\min}$ cluster merging approach.

5> After each merging, c sample points will be selected from original representative of previous clusters to represent new cluster.

6> Cluster merging will be stopped until target k cluster is found**.**

Example


a) Sample of Data
b) Three Clusters with Representative Points
c) Merge Clusters with Closest Points
d) Shrink Representative Points

Possamai Lino Data Mining - Cure 10

2.2.1.4 Robust Clustering Using Link (ROCK)

Clustering, in data mining, is useful to discover distribution patterns in the underlying data. Clustering algorithms usually employ a distance *metric* based (e.g., Euclidean) similarity measure in order to partition the database such that data points in the same partition are more similar than points in different partitions. In this paper, we study clustering algorithms for data with Boolean and categorical attributes. We show that traditional clustering algorithms that use distances between points for clustering are not appropriate for Boolean and categorical attributes. Instead, we propose a novel concept of *links* to measure the similarity/proximity between a pair of data points. We develop a *robust* hierarchical clustering algorithm ROCK that employs links and not distances when merging clusters. Our methods naturally extend to *non-metric* similarity measures that are relevant in situations where a domain expert/similarity table is the only source of knowledge. In addition to presenting detailed complexity results for ROCK, we also conduct an experimental study with real-life as well as synthetic data sets to demonstrate the effectiveness of our techniques. For data with categorical attributes, our findings indicate that ROCK not only generates better quality clusters than traditional algorithms, but it also exhibits good scalability properties.

<u>CHAMELEON:</u>

Chameleon is a hierarchical clustering algorithm that uses dynamic modeling to determine the similarity between pairs of clusters. edges operated on a sparse graph in which nodes represent data items and weighted represent similarities among the data items. this sparse graph representation of the data set allows CHAMELEON to scale to large data sets. CHAMELEON finds the clustering the data set by using a two-phase algorithm. During the first phase, CHAMELEON uses graph partitioning algorithm to cluster the data items into a large number of relatively small sub-clusters. During the second phase, it uses an agglomerative hierarchical clustering algorithm to find the genuine clusters by repeatedly combing together these sub-clusters.

The graph-partitioning algorithm partitions the k-nearest-neighbor graph such that it minimizes the edge cut. that is , a cluster C is partitioned into sub cluster C_i and C_j So as to minimize the weight of the edges that would be cut should be cut should C is bisected into C_i and C_j . Edge cut is denoted $EC(C_i, C_j)$. CHAMELEON determines the similarity between each pair of clusters C_i and C_j according to their relative interconnectivity $RI(C_i, C_j)$,

$$RI(C_i, C_j) = \frac{|\, EC\{C_i, C_j\}|}{\frac{1}{2}\left(|EC_{c_i}| + |EC_{c_j}|\right)}$$

$EC_{\{C_i, C_j\}}$ is the edge cut, defined as minimum sum of cut edges that would be cut from a cluster C to bisect it into $C_i and\ C_j$. $EC_{c_j} is\ the$ minimum sum of the cut edges that partition C_j into two roughly equal parts.

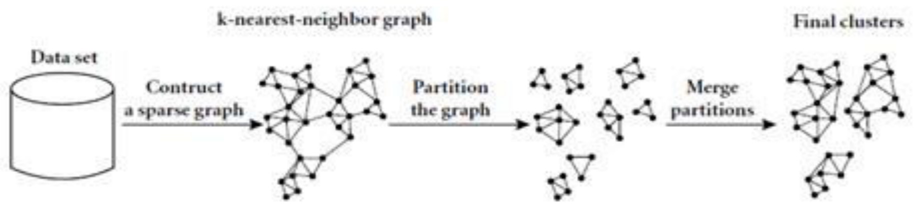


Figure 7.9 Chameleon: Hierarchical clustering based on *k*-nearest neighbors and dynamic modeling. Based on [KHK99].

2.2.2 Partitional Clustering

Partitional clustering decomposes a data set into a set of disjoint clusters. Given a data set of *N* points, a partitioning method constructs *K* ($N \geq K$) partitions of the data, with each partition representing a cluster. That is, it classifies the data into *K* groups by satisfying the following requirements: (1) each group contains at least one point, and (2) each point belongs to exactly one group. Notice that for fuzzy partitioning, a point can belong to more than one group.

Many partitional clustering algorithms try to minimize an objective function. For example, in *K*-means and *K*-medoid. if D is a data set of n objects, and k , the number of clusters to form , a partitioning algorithm organizes the objects onto K partitions , where each partition represents a cluster.

2.2.2.1 K-means

K-means clustering is a type of unsupervised learning, which is used when you have unlabeled data (i.e., data without defined categories or groups). The goal of this algorithm is to find groups in the data, with the number of groups represented by the variable *K*. The algorithm works iteratively to assign each data point to one of *K* groups based on the features that are provided. Data points are clustered based on feature similarity. The results of the *K*-means clustering algorithm are:

1. The centroids of the *K* clusters, which can be used to label new data

2. Labels for the training data (each data point is assigned to a single cluster)

Rather than defining groups before looking at the data, clustering allows you to find and analyze the groups that have formed organically. The "Choosing K" section below describes how the number of groups can be determined.

Each centroid of a cluster is a collection of feature values which define the resulting groups. Examining the centroid feature weights can be used to qualitatively interpret what kind of group each cluster represents.

This introduction to the *K*-means clustering algorithm covers:

- Common business cases where K-means is used
- The steps involved in running the algorithm
- A Python example using delivery fleet data

Algorithm K-means:

The *K*-means clustering algorithm uses iterative refinement to produce a final result. The algorithm inputs are the number of clusters *K* and the data set. The data set is a collection of features for each data point. The algorithms starts with initial estimates for the *K* centroids, which can either be randomly generated or randomly selected from the data set. The algorithm then iterates between two steps:

1. Data assignment step:

   Each centroid defines one of the clusters. In this step, each data point is assigned to its nearest centroid, based on the squared Euclidean distance. More formally, if c_i is the collection of centroids in set C, then each data point x is assigned to a cluster based on

2. Centroid update step:

   In this step, the centroids are recomputed. This is done by taking the mean of all data points assigned to that centroid's cluster.

3. The algorithm iterates between steps one and two until a stopping criterion is met (i.e., no data points change clusters, the sum of the distances is minimized, or some maximum number of iterations is reached).

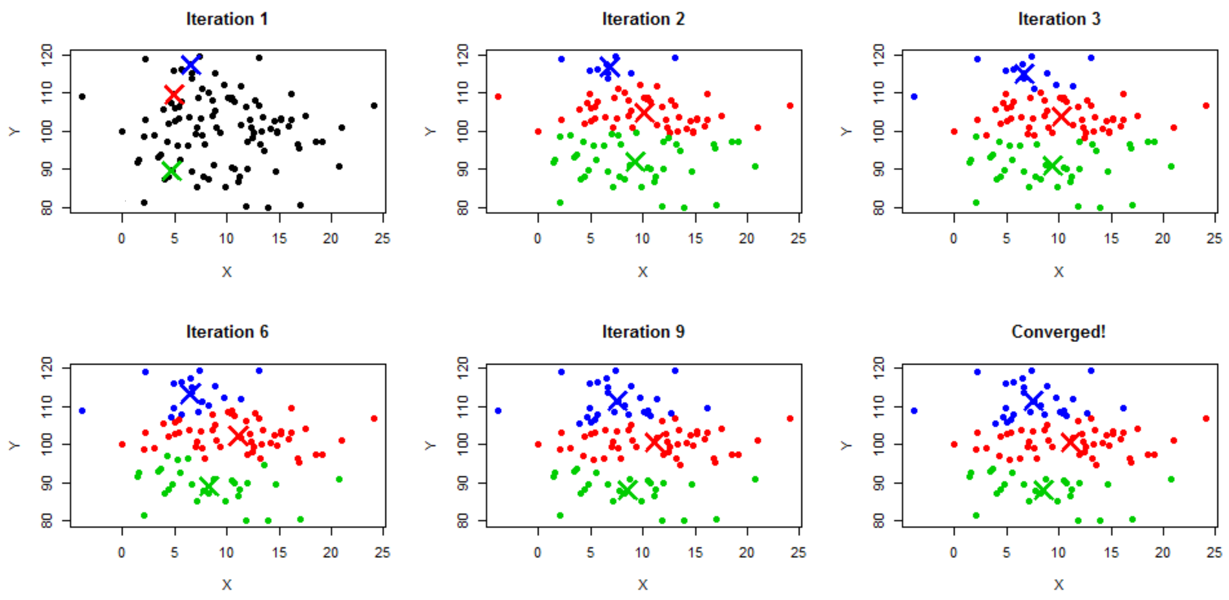
This algorithm is guaranteed to converge to a result. The result may be a local optimum (i.e. not necessarily the best possible outcome), meaning that assessing more than one run of the algorithm with randomized starting centroids may give a better outcome

```
Input: K: the number of clusters.
       D: a data set containing n objects.

Output: A set of k clusters.

Method:

2.2   Choose k data points as the initial centroid (cluster centers)
2.3   Repeat
2.4   For each data point $x \epsilon D$ do
2.5   Compute the distance from x to each centroid.
2.6   Assign x to the closest centroid
2.7   End for
2.8   Re-computer the centroids using the current cluster membership
2.9   Until the stopping criteria is met .
```



2.9.2.2 ISODATA

ISODATA unsupervised classification calculates class means evenly distributed in the data space then iteratively clusters the remaining pixels using minimum distance techniques. Each iteration recalculates means and reclassifies pixels with respect to the new means. Iterative class splitting, merging, and deleting is done based on input threshold parameters. All pixels are classified to the nearest class unless a standard deviation or distance threshold is specified, in which case some pixels may be unclassified if they do not meet the selected criteria. This process continues until the number of pixels in each class changes by less than the selected pixel change threshold or the maximum number of iterations is reached. The ISODATA methods is shown as follow:

2.9.2 Parameters, such as the number of the last clusters, a convergence condition of rearrangement, judgement conditions of a minute cluster, branch condition of division and fusion, and end conditions are determined.

2.9.3 The initial cluster center of gravity is selected.

2.9.4 Based on the convergence condition of rearrangement, an individual is rearranged in the way of the k-means method.

2.9.5 It consider with a minute cluster that it is bellow threshold with the member of individuals of a cluster, and expects from future clustering

2.9.6 When it is more than the threshold that exits within fixed limits which the number of the last clusters, has the minimum of the distance between the cluster center of gravity and is below threshold with the minimum of distribution in a cluster.

2.9.7 If the number of a clusters exceeds the fixed rang, when large, a cluster is divided, and when small, it will unite. it divides, if the number of times of a repetition is odd when there is the number of clusters within fixed limits, and if the number is even, it unites. If division and fusion finish, it ill return to 3 and processing will be repeated.

2.9.8 Division of a cluster . if it is more than threshold with distribution of a cluster, carry out the cluster along with the 1st principle component for 2 minutes, and search for the new cluster center of gravity.

2.9.8.2 Partitioning Around Medoids (PAM)

PAM stands for "partition around medoids". The algorithm is intended to find a sequence of objects called medoids that are centrally located in clusters. Objects that are tentatively defined as medoids are placed into a set S of selected objects. If O is the set of objects that the set U = O − S is the set of unselected objects. The goal of the algorithm is to minimize the average dissimilarity of objects to their closest selected object. Equivalently, we can minimize the sum of the dissimilarities between object and their closest selected object. The algorithm has two phases: (i) In the first phase, BUILD, a collection of k objects are selected for an initial set S. (ii) In the second phase, SWAP, one tries to improve the quality of the clustering by exchanging selected objects with unselected objects. The goal of the algorithm is to minimize the average dissimilarity of objects to their closest selected object. Equivalently, we can minimize the sum of the dissimilarities between object and their closest selected object. For each object p we maintain two numbers:

Dp, the dissimilarity between p and the closest object in S, and
the dissimilarity between p and the second closest object in S.

These numbers must be updated every time when the sets S and U change. Note that $Dj \leq Ej$ and that we have $p \in S$ if and only if $Dp = 0$. The BUILD phase entails the following steps: 1. Initialize S by adding to it an object for which the sum of the distances to all other objects is minimal. 2. Consider an object $i \in U$ as a candidate for inclusion into the set of selected objects. 3. For an object $j \in U - \{i\}$ compute Dj , the dissimilarity between j and the closest object in S.

If $Dj > d(i, j)$ object j will contribute to the decision to select object i (because the quality of the clustering may benefit); let $Cji = \max\{Dj - d(j, i), 0\}$. 5. Compute the total gain obtained by adding i to S as gi = P $j \in U$ Cji. 6. Choose that object i that

maximizes gi ; let S := S ∪ {i} and U = U − {i}. These steps are performed until k objects have been selected. The decisions taken in assessing object i are shown in Figure 1. The second phase, SWAP, attempts to improve the set of selected objects and, therefore, to improve the quality of the clustering. This is done by considering all pairs (i, h) ∈ S ×U and consists of computing the effect Tih on the sum of dissimilarities between objects and the closest selected object caused by swapping i and h, that is, by transferring i from S to U and transferring h to from U to S.
The computation of Tih involves the computation of the contribution Kjih of each object j ∈ U − {h} to the swap of i and h. Note that we have either d(j, i) > Dj or d(j, i) = Dj .

Algorithm: K-Medoids

Input: K: the number of clusters,

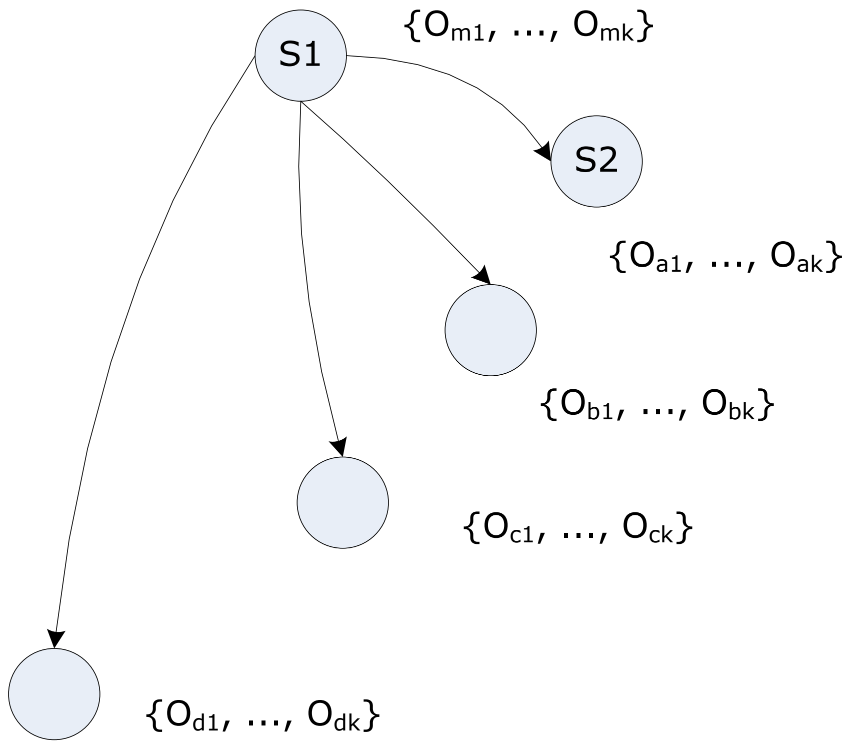
D: a data set containing n objects

Output: A set of k clusters**.**

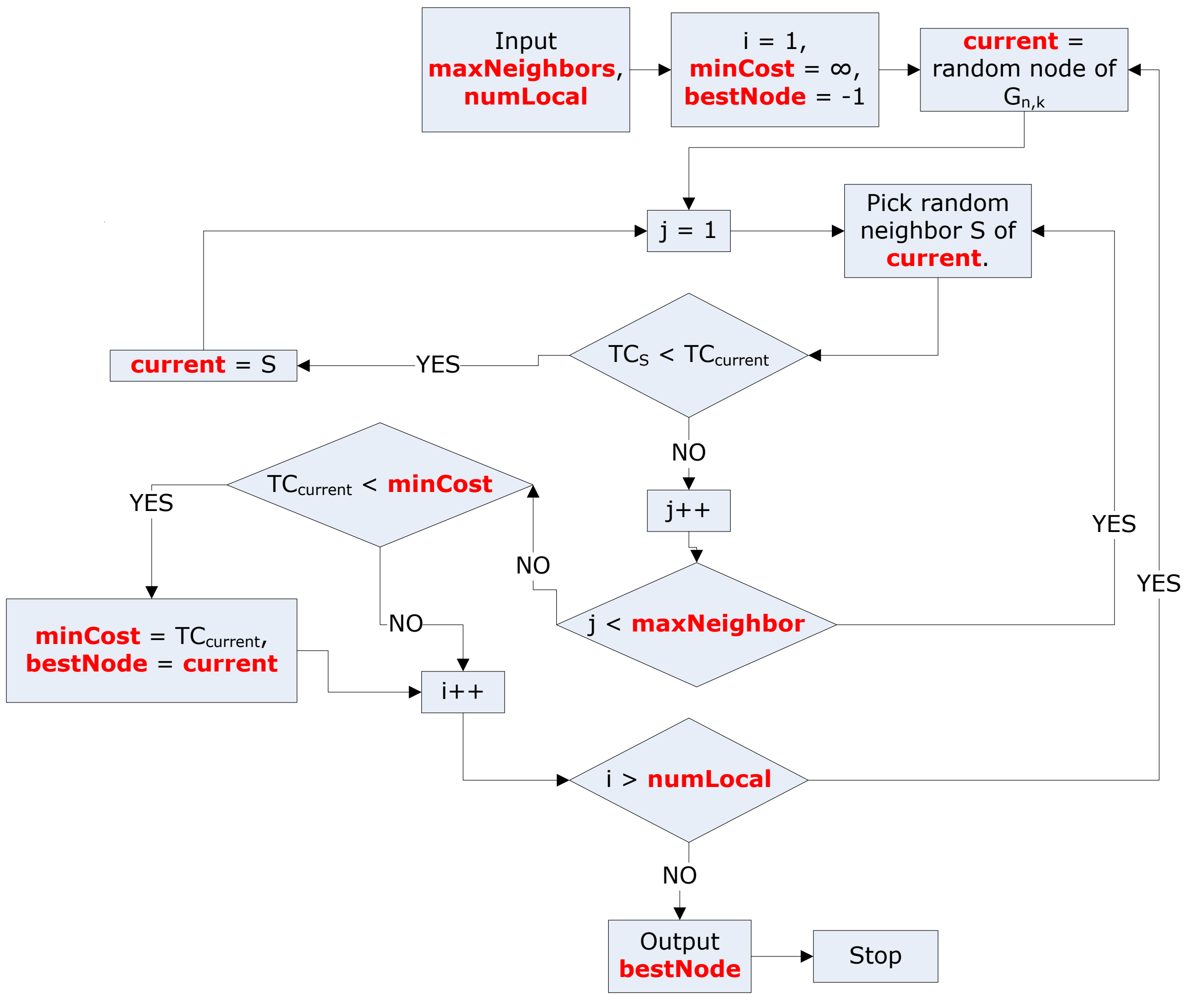
Method:

2.9.8.3 Arbitrarily choose k objects in D as the initial representative objects or seeds;
2.9.8.4 Repeat
2.9.8.5 Assign each remaining object to the cluster with the nearest representative object
2.9.8.6 Randomly select a non-representative object, $O_{random;}$
2.9.8.7 Compute the total cost, S, of swapping representative object O_j with $O_{random\,;}$
2.9.8.8 If S<0 then swap O_j with O_{random} to form new set of k representative objects;
2.9.8.9 Until no change**;**

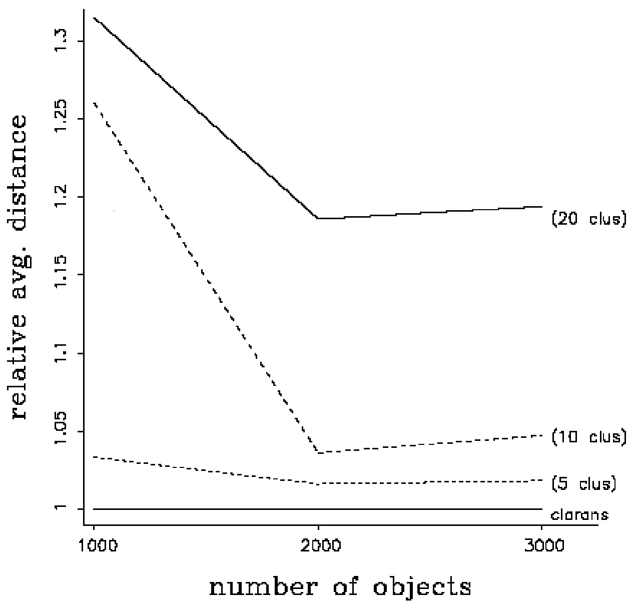
- **2.2.2.4 CLARANS**
  CLARSANS (A Clustering Algorithm based on Randomized Search) is the clustering process can be presented a search a graph where every node is a potential solution that is a of K medoids . Improvement over PAM
- Finds medoids in a sample from the dataset
- [Idea]: If the samples are sufficiently random, the medoids of the sample approximate the medoids of the dataset
- [Heuristics]: 5 samples of size 40+2k gives satisfactory results
- Works well for large datasets (n=1000, k=10)

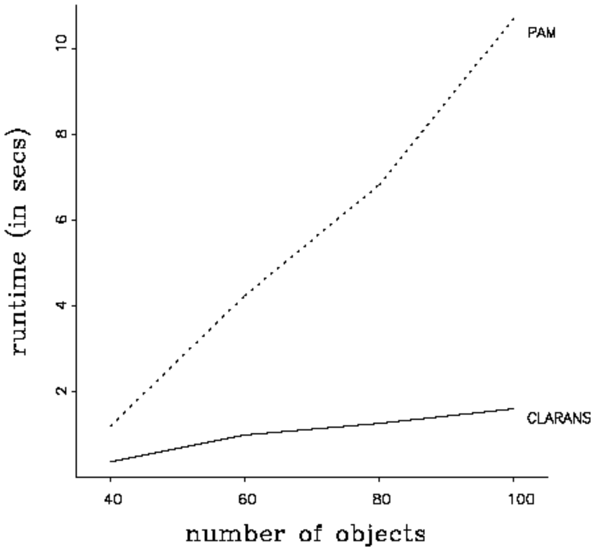

S1
$\{O_{m1}, \ldots, O_{mk}\}$
S2
$\{O_{a1}, \ldots, O_{ak}\}$
$\{O_{b1}, \ldots, O_{bk}\}$
$\{O_{c1}, \ldots, O_{ck}\}$
$\{O_{d1}, \ldots, O_{dk}\}$

- A graph abstraction, $G_{n,k}$
- Each vertex is a collection of k medoids
- $| S1 \quad S2 | = k - 1$
- Each node has k(n-k) neighbors
- Cost of each node is total dissimilarity of objects to their medoids
- PAM searches whole graph
- CLARA searches subgraph


Input maxNeighbors, numLocal
i = 1, minCost = ∞, bestNode = -1
current = random node of G n,k
j = 1
Pick random neighbor S of current.
current = S
YES
TC S < TC current
NO
TC current < minCost
YES
j++
YES
NO
YES
minCost = TC current, bestNode = current
NO
j < maxNeighbor
i++
i > numLocal
NO
Output bestNode
Stop

- Outperforms PAM and CLARA in terms of running time and quality of clustering
- $O(n^2)$ for each iteration





<u>**2.9.9.10K-mode**</u>

The K-modes algorithm extends K-means paradigm to cluster categorical data by removing the limitation imposed by K-means through following modifications:

- Using a simple matching dissimilarity measure or the hamming distance for categorical data objects.
- Replacing means of cluster by their modes.

Algorithm: k-means

a) Select K initial nodes, one of each of the cluster

b) Allocate data object to the cluster whose mode is nearest to it according to below equation.

$$d(X,Y) = \sum_{j=1}^{F} \delta(x_i, y_i)$$

$$where \quad \delta(x_j, y_j) = \{0 \quad (x_{j=}y_j)$$
$$\{1 \quad (x_{j\neq}y_j)$$

c) Compute new modes of all clusters.
d) Repeat step b) to c) until no data object has changed cluster membership .

2.9.9 Distance Based Clustering

Clustering is the data mining technique that groups the data objects into classes or clusters, so that objects within a cluster have high similarity in comparison to one another(intra-class similarity) but are very dissimilar to objects in other clusters(inter-class similarity).There are many clustering methods available, and each of them may give a different grouping of a dataset. These methods are divided into two basic types: hierarchical and partitional clustering [1]. Hierarchical clustering either merges smaller clusters into larger ones (Agglomerative) or splits larger clusters into smaller ones (Divisive). Agglomerative clustering starts with one point in each cluster and at each step selects and merges two most clusters that are most similar in characteristics. The process is repeated till the required number of clusters is formed. Divisive clustering starts by considering all data points in one cluster and splits them until the required number of clusters is formed. The agglomerative and divisive clustering methods further differ in the rule by which it is decided which two small clusters are merged or which large cluster is split. Partitional clustering, attempts to directly decompose the data set into a set of disjoint clusters.

Each object is a member of the cluster with which it is most similar. K-means [2, 3] is one of the most widely used partitional clustering algorithms. The algorithm clusters the n data points into k groups, where k is provided as an input parameter. It defines k centroids, one for each cluster. For this k data points are selected at random from D as initial centroids. The next step is to take each point belonging to the given data set and assign it to the nearest centroid. Euclidean distance is generally considered to determine the distance between data points and the centroids. When all the points are included in some clusters, the mean of each cluster is calculated to find the new centroids for each cluster. It then assigns all the data points to clusters based upon their proximity to the mean of the cluster. The cluster's mean is then recomputed and the process begins again. In due course, a situation is reached where the

clusters centers do not change anymore. This becomes the convergence criterion for clustering.

The common used distance measure is the Euclidean metric which defines the distance between two points p = $(p_1, p_2, \ldots\ldots)$ and $q = (q_1, q_2 \ldots\ldots)$ is given by :

$$D = \sum_{i=1}^{k} (p_i - q_j)\hat{}2$$

For Higher dimensional data, a popular measure is the Murkowski Metric

$$D_p(X_i, X_j) = (\sum_{k=1}^{d} || X_{i,k} - X_{j,k} ||^p)^{1/p}$$

Where d is the dimensionality of the data. the Euclidean distance is a special case where p =2 , while Manhattan metric p=1 , there are no general theoretical guidelines for selecting a measure for any given application .

2.2.3.1 Nearest Neighbor Clustering:

In pattern recognition, the ***k*-nearest neighbors algorithm** (***k*-NN**) is a non-parametric method used for classification and regression.[1] In both cases, the input consists of the *k* closest training examples in the feature space. The output depends on whether *k*-NN is used for classification or regression:

- In *k-NN classification*, the output is a class membership. An object is classified by a plurality vote of its neighbors, with the object being assigned to the class most common among its *k* nearest neighbors (*k* is a positive integer, typically small). If *k* = 1, then the object is simply assigned to the class of that single nearest neighbor.
- In *k-NN regression*, the output is the property value for the object. This value is the average of the values of *k* nearest neighbors.

k-NN is a type of instance-based learning, or lazy learning, where the function is only approximated locally and all computation is deferred until classification. The *k*-NN algorithm is among the simplest of all machine learning algorithms.

Both for classification and regression, a useful technique can be used to assign weight to the contributions of the neighbors, so that the nearer neighbors contribute more to the average than

the more distant ones. For example, a common weighting scheme consists in giving each neighbor a weight of 1/*d*, where *d* is the distance to the neighbor.[2]
The neighbors are taken from a set of objects for which the class (for *k*-NN classification) or the object property value (for *k*-NN regression) is known. This can be thought of as the training set for the algorithm, though no explicit training step is required.
A peculiarity of the *k*-NN algorithm is that it is sensitive to the local structure of the data.

2.9.11 Fuzzy Clustering :

Fuzzy clustering generalizes partition clustering methods (such as k-means and medoid) by allowing an individual to be partially classified into more than one cluster. In regular clustering, each individual is a member of only one cluster. Suppose we have K clusters and we define a set of variables mi1 mi2 mi K , ,Λ, that represent the probability that object i is classified into cluster k. In partition clustering algorithms, one of these values will be one and the rest will be zero. This represents the fact that these algorithms classify an individual into one and only one cluster. In fuzzy clustering, the membership is spread among all clusters. This can now be between zero and one, with the stipulation that the sum of their values is one. We call this a fuzzification of the cluster configuration. It has the advantage that it does not force every object into a specific cluster. It has the disadvantage that there is much more information to be interpreted.

The data have three obvious clusters and two outlier points (6 and 13). A regular clustering algorithm searching for three clusters will force these two points into specific clusters. This may cause distortion in the final solution. Fuzzy clustering, however, will assign a probability of about 0.33 for each cluster. This equal membership probability signals that these two points are outliers. When you only have two variables, you can plot your data and see what the clusters are. Unfortunately, most clustering projects come with more than two variables, so plotting is not possible. Hence, we must use techniques like fuzzy clustering to deal with the anomalies that can occur.

2.2.4.1 Fuzzy c-means (FCM)

This algorithm works by assigning membership to each data point corresponding to each cluster center on the basis of distance between the cluster center and the data point. More the data is near to the cluster center more is its membership towards the particular cluster center. Clearly, summation of membership of each data point should be equal to one. After each iteration membership and cluster centers are updated according to the formula. The FCM algorithm attempts to partition a finite collection of n elements X = $\{x_1 x_{2,.....} x_n\}$ into a collection of c fuzzy clusters with respect to some given criterion. the algorithm returns a list of cluster centers C = $\{c_1 c_2, \ldots. c_n\}$

And a partition matrix W = $W_{ij} \in [0,1], i = 1, \ldots\ldots n, j = 1, \ldots\ldots c$ where each elements W_{ij} tells the degree to which elements x_i belongs to cluster c_j.

$$\boldsymbol{arg_c min \sum_{i=1}^{n} \quad \sum_{j=1}^{c} w_{ij}{}^{m} \,||\, X_i - C_j \,||^2}$$

Where m is real number greater than1 , u_{ij} is the degree od membership of x_i in the cluster j, x_i *is the ith* of dimensional measured data c_j is the d-dimension center of the cluster. Fuzzy partitioning is carried out through an iterative optimization of the objective function shown above, with the update of membership u_{ij} and the cluster centers c_j *by*:

$$\boldsymbol{W_{ij}^{m} = \frac{1}{\sum_{k=1}^{c} \quad \frac{(||x_i - c_j||)}{(||x_i - c_k||)}^{2/m-1} \quad . \quad . \; c_j = \frac{\sum_{i=1}^{N} u_{ij}^{m} x_i}{\sum_{i=1}^{N} u_{ij}^{m}}}}$$

<u>2.9.11.2Gustafson Kessel (GK) :</u>

Clustering is a useful tool for analyzing data by quantitative determination of underlying structures. Various similarity measures such as probability distributions, regression, correlation, hypothesis testing and distance methods have been used for discovering patterns. However, real world data sets often suffer from curse of dimensionality due to inherent sparsity of high dimensional data. Traditional clustering algorithms fail in identifying hidden relationships of underlying structure due to the reason that the nearest neighbor of a pattern may be nearly as close as farthest neighbor, if distance is computed in full dimensional space. To cope with the problem of high dimensional feature spaces, feature reduction and feature selection techniques have been used in the literature. Feature reduction techniques project the whole feature space to a lower dimensional subspace so that cluster structures become apparent. However, feature reduction techniques cannot demonstrate patterns clustered differently in varying subspaces. Traditional feature selection techniques such as principal component analysis (PCA) have been used in multivariate statistics, methods based on singular value decomposition have been used in information retrieval to transform the attributes. Feature selection suffers from usability problem as it becomes hard to interpret the results intuitively. Hence, there is a need for more generalized techniques that can be used to obtain meaningful clusters. Agrawal et al. introduced the concept of projective clustering to generate clusters in distinctive subspaces of multidimensional space so that intra cluster similarity is maximized and inter cluster similarity is minimized. Elke et al. proposed an algorithm to discover cluster in arbitrarily oriented subspaces.

<u>2.2.5 Evolutionary Clustering</u>

Evolutionary clustering is the problem of processing timestamped data to produce a sequence of clustering's; that is, a clustering for each timestep of the system. Each clustering in the sequence should be similar to the clustering at the previous timestep, and should accurately reflect the data arriving during that timestep. The primary setting for this problem is the following. Every day, new data arrives for the day, and must be incorporated into a clustering. If the data does not deviate from historical expectations, the clustering should be "close" to that from the previous day, providing the user with a familiar view of the new data. However, if the structure of the data changes significantly, the clustering must be modified to reflect the new structure. Thus, the clustering algorithm must trade off the benefit of maintaining a consistent clustering over time with the cost of deviating from an accurate representation of the current data. The benefits of evolutionary clustering compared to traditional clustering appear in situations in which the current (say, daily) clustering is being consumed regularly by a user or system. In such a setting, evolutionary clustering is useful for the following reasons: (1) Consistency: A user will find each day's clustering familiar, and so will not be required to learn a completely new way of segmenting data. Similarly, any insights derived from a study of previous clusters are more likely to apply to future clusters. (2) Noise removal: Providing a high-quality and historically consistent clustering provides greater robustness against noise by taking previous data points into effect. As we describe later, our method subsumes standard approaches to windowing and moving averages. (3) Smoothing: If the true clusters shift over time, evolutionary clustering will naturally present the user with a smooth view of the transition. (4) Cluster correspondence: As a side effect of our framework, it is generally possible to place today's clusters in correspondence with yesterday's clusters. Thus, even if the clustering has shifted, the user will still be situated within the historical context.

<u>2.2.5.1 GA Based Clustering:</u>

Representative:

```
Cluster centers encoded in the chromosomes.
        For a d-dimensional space
        Length of chromosome = d * k
Population Initialization:
        Initial cluster centers = c randomly selected points from the data
        For each chromosome i in the population
For each cluster j
        P = randomly chosen points from data set
        Population[i][j] =p
End
End
        Fitness computation:
        Phase 1 – cluster Assignment
                 Each point is assigned to the nearest cluster center
        Phase 2- the cluster centers encoded in the chromosome are replaced by the mean
points of the respective clusters.
        Phase 3- Fitness = (1 / Clustering metric J)
```

$$J = \sum \quad \sum d^2 \; (x_k^j, v_j), j \; = 1, \ldots . c$$

Crossover:

Single point crossover with a fixed crossover probability

Mutation:

A number δ in the range [0,1] is generated with uniform distribution. if the value at a gene position is v , after mutation it becomes

v = v + 2 $* \; \delta \; * \; v$

v = v + 2 * δ

Termination:

GA clustering is run for fixed number of iterations. elitism is incorporated. Best string (Lowest J) is taken as solution of the clustering**.**

2.2.5.2 VGA clustering

Each chromosome encodes different number of clusters.

Representation:

For a d-dimensional space, chromosome length = d * c

$c_i \; =$
$rand() \; mod \; k \; * + 2 \, , k \; * \qquad is \; the \; soft \; estimation \;\; of \; the \; upper \; bound \; of \; number$
cluster

Fitness Computation

Phase 1 – Each point is assigned to the nearest cluster center.

Phase 2 – Computer new centroids and replace them in the chromosome

Phase 3- Use cluster validity index as fitness criterion

Crossover:

Parent chromosome $P_{1,}P_2$ have C_1, C_2 $cluster\ centers\ repectively$

C_1 , $crossover\ point\ in$ P_1 = rand () mod C_1

C_2, crossover point in P_2 = rand () mod C_2

LB ($C2$) = min [2, max (0,2-($C_1 - C_2$))]

UB(C3) = max [C2 – max (0,2-C1)]

C2 = LB(C2) + rand () mod [UB(C2) -LB(C2)]

Mutation:

A number δ in the rang [0,1] is generated with uniform distribution. if the value at a gene position is v, after mutation it becomes

V = V + 2*δ * V if V<>0

V = $\pm$ 2 *δ if V =0

2.9.12Model Based Clustering

Model Based clustering methods attempts to optimize the fit between the given data and some mathematical model Such methods are often on the assumption that the data are generated by a mixture of underlying probability distribution.

2.2.6.1 Expectation Maximization

a) Make an initial guess of the parameter vector: this involves randomly selecting K objects to represent the cluster means or centers (as in k-means partitioning) as a well as making guess for the additional parameter.

b) Iteratively refine the parameters based on the following two steps:

() Expectation step: assign points to clusters

$$P\ (d_i \epsilon\ c_k\ )\ =\ W_k\ Pr\ (\ {}^{d_i}/_{c_k}\ /\ \textstyle\sum_j w_j\ Pr({}^{d_i}/_{c_i})$$

$$w_k = \frac{\sum_i Pr\,(d_i \in c_k)}{N}$$

() Maximation step: estimate model parameters

$$\boldsymbol{\mu_k = \frac{1}{m} \sum_{i-1}^{m} \frac{d_i P(d_i \epsilon\, c_k)}{\sum_k P(d_i \epsilon\, c_j)}}$$

<u>2.2.6.2 Artificial Neural Network Based Clustering</u>

Vector quantization (VQ) is a classical method for approximating a continuous probability density function (PDF) p(x) of the vector variable x ∈ R n by using a finite number of prototypes. A set of feature vectors x is represented by a finite set of prototypes {c1, . . . , cK } ⊂ R n , referred to as the codebook. Codebook design can be performed by using clustering. Once the codebook is specified, approximation of x is to find the reference vector c from the codebook that is closest to x (Kohonen, 1989, 1997). This is the nearest-neighbor paradigm, and the procedure is actually simple competitive learning (SCL). The codebook can be designed by minimizing the expected squared quantization error

E = $\int ||x - C||^2$ p(x) dx

where c is a function of x and c_i . Given the sample x_t , an iterative approximation scheme for finding the codebook is derived by Kohonen (1997)

$$c_i\,(t+1) = c_i(t) + \varphi(t)\,\delta_{wi}\,|\,x_i - c_i(t)|^{\,2}$$

where the subscript w corresponds to the winning prototype, which is the prototype closest to x_t δ_{wi} is the Kronecker delta, with δ_{wi}taking 1 for w = i and 0 otherwise, and η > 0 is a small learning rate that satisfies the classical Robbins–Monro conditions, that is, Pη(t) = ∞ and Pη 2 (t) < ∞. Typically, η is selected to be decreasing monotonically in time. For example, one can select η(t) = η0 1 − t T , where η0 ∈ (0, 1] and T is the iteration bound. This is the SCL based VQ. Voronoi tessellation, also called Voronoi diagram, is useful for demonstrating VQ results. The space is partitioned into a finite number of regions bordered by hyperplanes. Each region is represented by a codebook vector, which is the nearest neighbor to any point within the region. All vectors in each region constitute a Voronoi set. For a smooth underlying probability density p(x) and a large K, all regions in an optimal Voronoi partition have the same within-region variance σ.

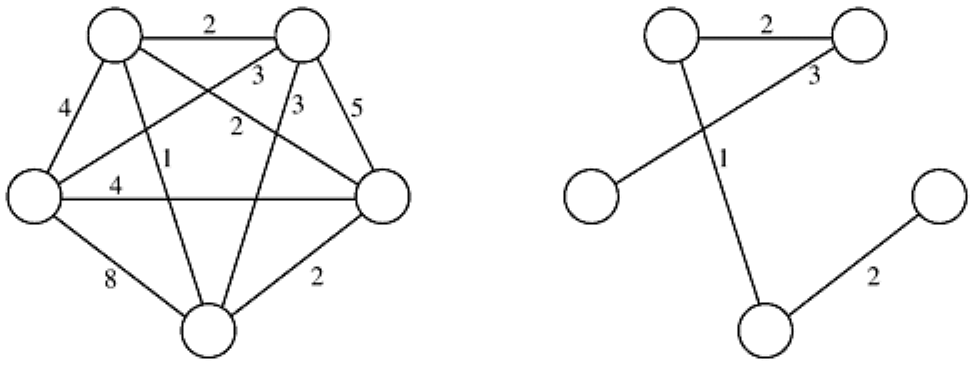
2.2.7 Graph Based Clustering :

Any nonuniform data contains underlying structure due to the heterogeneity of the data. The process of identifying this structure in terms of grouping the data elements is called clustering, also called data classification [152]. The resulting groups are called clusters. The grouping is usually based on some similarity measure defined for the data elements. Clustering is closely related to unsupervised learning in pattern recognition systems [81]. A basic task in unsupervised learning is to classify a data set into two or more classes based on a similarity measure over the data, without resorting to any a priori information on how the classification should be done. Graphs are structures formed by a set of vertices (also called nodes) and a set of edges that are connections between pairs of vertices. Graph clustering is the task of grouping the vertices of the graph into clusters taking into consideration the edge structure of the graph in such a way that there should be many edges within each cluster and relatively few between the clusters. Graph clustering in the sense of grouping the vertices of a given input graph into clusters, which is the topic of this survey, should not be confused with the clustering of sets of graphs based on structural similarity; such clustering of graphs as well as measures of graph although many of the techniques involved are closely related to the task of finding clusters within a given graph. As the field of graph clustering has grown quite popular and the number of published proposals for clustering algorithms as well as reported applications is high, we do not even pretend to be able to give an exhaustive survey of all the methods, but rather an explanation of the methodologies commonly applied and pointers to some of the essential publications related to each research branch.

2.2.7.1 Minimum Spanning Tree:

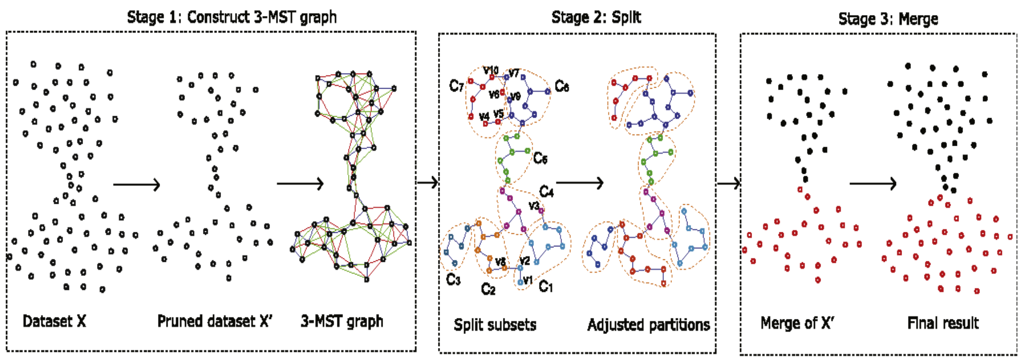
.

An *edge-weighted graph* is a graph where we associate *weights* or *costs* with each edge. A *minimum spanning tree (MST)* of an edge-weighted graph is a spanning tree whose weight (the sum of the weights of its edges) is no larger than the weight of any other spanning tree. Let G = (V,E) be connected , undirected graph . For each edge (u,v) in E, we have a weight w(u,v) specifying the cost (length of edge) to connect u and v . An acyclic subset T to E that connects all of the vertices in V and whose total weight is minimized , such a tree is called Minimum Spanning Tree (MST). MST can be constructed using any of the two algorithm – krushkal’s algorithm or Prim’s Algorithm .



<u>The main procedure of MST based clustering algorithm</u>

a) We have to construct the MST from the graph.
b) After constructing of MST from the graph we have to detect longer inconsistent edges belongs to the MST.
c) After identifying those inconsistent longer edges we have to removes those edges from the MST . after removing n-1 inconsistent edges n numbers will be formed.



[illegible]

<u>2.2.8 Density -Based Clustering</u>

Clusters are dense regions in the data space, separated by regions of lower object density – A cluster is defined as a maximal set of density-connected points – Discovers clusters of arbitrary shape. The most popular type of density-based clustering algorithm includes DBSCAN, DENCLUE and OPTICS.

2.2.8.1 DBSCAN:

ε-Neighborhood – Objects within a radius of ε from an object.

$$N_{\varepsilon}(p) : \{ q \mid d\,(p, q) \;\le\; \varepsilon$$

Density based clustering algorithm has played a vital role in finding non linear shapes structure based on the density. Density-Based Spatial Clustering of Applications with Noise (DBSCAN) is most widely used density based algorithm. It uses the concept of **density reachability** and **density connectivity**.

Density Reachability - A point "p" is said to be density reachable from a point "q" if point "p" is within ε distance from point "q" and "q" has sufficient number of points in its neighbors which are within distance ε.

Density Connectivity - A point "p" and "q" are said to be density connected if there exist a point "r" which has sufficient number of points in its neighbors and both the points "p" and "q" are within the ε distance. This is chaining process. So, if "q" is neighbor of "r", "r" is neighbor of "s", "s" is neighbor of "t" which in turn is neighbor of "p" implies that "q" is neighbor of "p".

Algorithmic steps for DBSCAN clustering

Let X = $\{x_1, x_2, x_3, ..., x_n\}$ be the set of data points. DBSCAN requires two parameters: ε (eps) and the minimum number of points required to form a cluster (minPts).

1) Start with an arbitrary starting point that has not been visited.

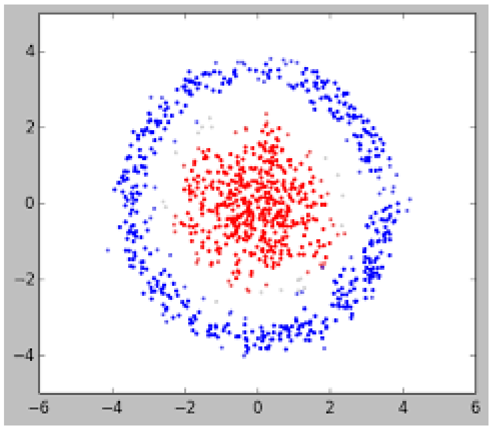
2) Extract the neighborhood of this point using ε (All points which are within the ε distance are neighborhood).

3) If there are sufficient neighborhood around this point then clustering process starts and is marked as visited else this point is labeled as noise (Later this point can become the part of the cluster).

4) If a point is found to be a part of the cluster then its ε neighborhood is also the part of the cluster and the above procedure from step 2 is repeated for all ε neighborhood points. This is repeated until all points in the cluster is determined.

5) A new unvisited point is retrieved and processed, leading to the discovery of a further cluster or noise.

6) This process continues until all points are marked as visited.



2.2.8.2 OPTICS

Clustering is a powerful unsupervised knowledge discovery tool used today, which aims to segment your data points into groups of similar features. However, each algorithm is pretty sensitive to the parameters. Similarity based techniques (K-means, etc) are tasked with designating how many clusters exist, while hierarchical usually require manual intervention to decide when to assign finished clusters. The most common density based approach, DBSCAN, requires only two parameters pertaining to how it defines its “Core Points”, but finding the parameters can often be an extremely difficult task. It also will not be able to find clusters of differing densities.

There is a relative of DBSCAN, called OPTICS (Ordering Points to Identify Cluster Structure), that invokes a different process. It will create a reachability plot that is then used to extract clusters and although there is still an input, maximum epsilon, it is mostly introduced only if you would like to try and speed up computation time. The other parameters don’t have as big an effect as their counterparts in other clustering algorithms, and are much easier to use defaults. First, I will explain a little how this algorithm works, how it can include in-line outlier detection, and then how it was very useful for me in a recent application.

2.2.8.3 DENCLUE

The technological revolution has generated tens of terabytes of heterogeneous data every day. According to an investigation made by the institute IDC1, 1.8 zettabyte data was created in 2011, 2.8 zettabytes in 2012 and it will increase to 40 zettabytes in 2020. This large quantity of complex data, which can be part of Big data, needs a more developed technology to better store, use and analysis. There is several propositions to define Big Data 2. The most widely used definition is that proposed by Gartner 3, it is based on the notion of the 3 Vs: Volume, Velocity and Variety Velocity: The speed and high frequency of data creation represent a real challenge, especially in real time applications. We must note that the thousand of terabytes of data are generated every hour 5. • Variety: The huge amounts of data are generated from the social networks, the smart phones, the sensors and more. As a result, heterogeneous data are produced. In this context, researchers have gone in the direction to add other Vs for big data definition. H.U Buhl et al. 6 suggest that the study of the correctness and accuracy of information is necessary and then include the fourth V for veracity. J. Gantz and D. Reinsel 7 take into account another V which is value. The value of the data extracted after analysis are more important than the data itself. The variability is the sixth V which has been proposed by the NIST8, it refers to the transformation that affect the meaning of the same information, referenced in other contexts. As mentioned above, the big data produce a heterogeneous data (Variety) which makes it difficult to exploit. To overcome these limits, clustering methods can be considered as a promising solution. Generally, The challenge in large data is to provide a clustering method that produce an acceptable quality of clustering in a reasonable runtime. For this end, the density-based clustering methods are widely used thanks to their ability to classify the large databases (Volume), and to their efficiency to omit noisy data (Veracity). In this context, M. Ester et al. 9 proposed a DBSCAN method as a new density-based clustering algorithm for large spacial databases. Based on this work, a lot of variant of DBSCAN are proposed in literature such as OPTICS10, ST-DBSCAN11, MR-DBSCAN12, etc. However DBSCAN and its variants operate efficiently only on spatial data, and have their limitations with respect to large dimensional data. To overcome these shortcoming, the DENCLUE algorithm was proposed by A. Hinneburg and D. A. Keim in13. Nevertheless, the DENCLUE algorithm suffers in term of the execution time. This is due to the hill climbing method which slows down the convergence to the local maximum. That is why in the present contribution we aim to improve this algorithm in order to obtain clusters in a reasonable response time. The paper is organized as follows. We present in the next section, the DENCLUE algorithm

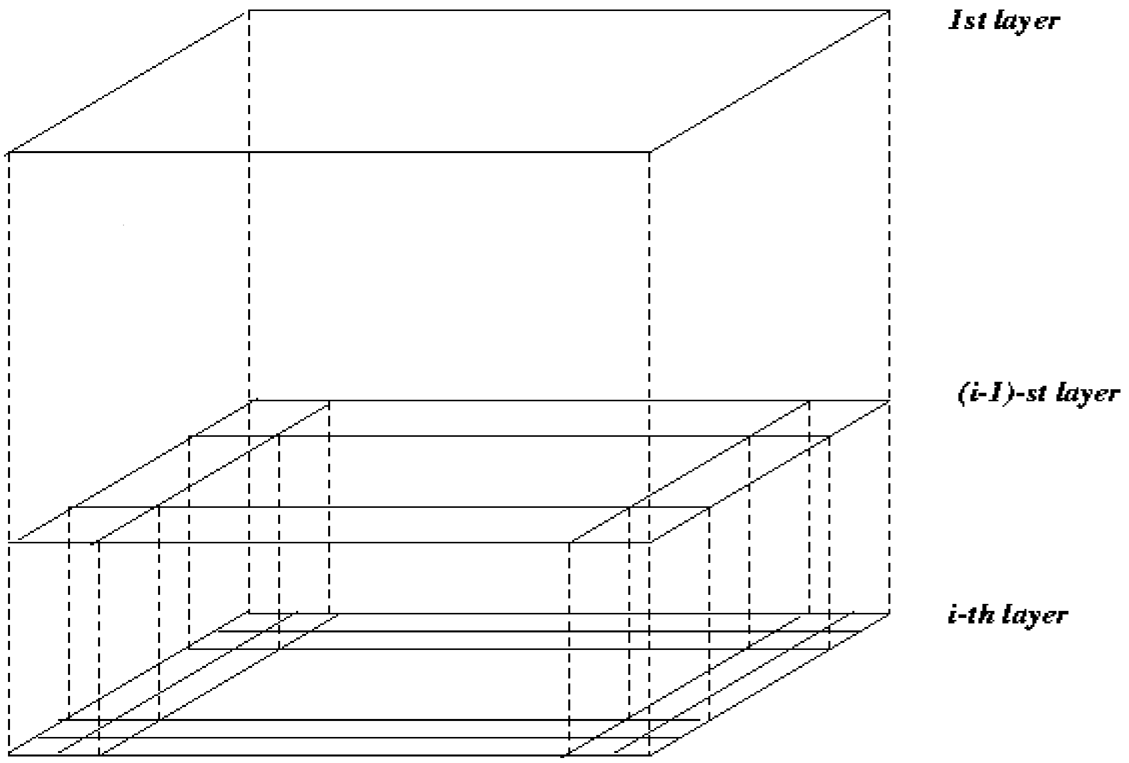
2.2.9 Grid-Based Clustering

Density-based and/or grid-based approaches are popular for mining clusters in a large multidimensional space wherein clusters are regarded as denser regions than their surroundings. In this chapter, we present some grid-based clustering algorithms. The computational complexity of most clustering algorithms is at least linearly proportional to the size of the data set. The great advantage of grid-based clustering is its significant reduction of the computational complexity, especially for clustering very large data sets.The grid-based clustering approach differs from the conventional clustering algorithms in that it is concerned not with the data points but with the value space that surrounds the data points. In general, a typical grid-based clustering algorithm consists of the following five

1. Creating the grid structure, i.e., partitioning the data space into a finite number of cells.
2. Calculating the cell density for each cell.
3. Sorting of the cells according to their densities.
4. Identifying cluster centers.
5. Traversal of neighbor cell

2.2.9.1 STING

- Wang, Yang and Muntz (VLDB'97)
- The spatial area area is divided into rectangular cells
- There are several levels of cells corresponding to different levels of resolution



- Each cell at a high level is partitioned into a number of smaller cells in the next lower level
- Statistical info of each cell is calculated and stored beforehand and is used to answer queries
- Parameters of higher level cells can be easily calculated from parameters of lower level cell
  - *count*, *mean*, *s*, *min*, *max*
  - type of distribution—normal, *uniform*, etc.
- Use a top-down approach to answer spatial data queries
- Advantages:
  - Query-independent, easy to parallelize, incremental update
  - *O(K),* where *K* is the number of grid cells at the lowest level
- Disadvantages:
  - All the cluster boundaries are either horizontal or vertical, and no diagonal boundary is detected

2.2.9.2 WEB CLUSTER:

Web clustering Engines are emerging trend in the field of information retrieval. They organize search results by topic, thus offering a complementary view to the flat ranked list returned by the conventional search engines.

The search results returned by traditional search engines on different subtopics or meanings of a query will be mixed together in the list so that the user may have to sift through a large number of irrelevant items to locate those of interest. The Web clustering engines categorize the search results into different hierarchical groups/clusters and display those cluster labels.

Hence the user can locate the desired document very fast. In this seminar we discuss different phases in the implementation of web clustering engines in detail and also incorporate some of the web clustering algorithms, their advantages and issues. We will familiarize some currently using web clustering engines. Some future research directions are also presented.

Clustering is the act of grouping similar object into sets. The distance between the objects in the same cluster (inter-cluster variations) should be minimum and the distance between objects in different clusters(intra-cluster variations) should be maximum. In the web search context, organizing web pages (search results) into groups, so that different groups correspond to different user needs. In 1979 Van Rijsbergen introduced the concept Cluster Hypothesis in the field of information retrieval. It states that "Closely related documents tend to be relevant to the same requests." Web Clustering Engines are the systems that perform clustering of web search results. This systems group the results returned by a search engine into a hierarchy of labeled clusters (also called categories).

2.2.10 Subspace Clustering:

Cluster analysis seeks to discover groups, or clusters, of similar objects. The objects are usually represented as a vector of measurements, or a point in multidimensional space. The similarity between objects is often determined using distance measures over the various dimensions in the dataset Technology advances have made data collection easier and faster, resulting in larger, more complex datasets with many objects and dimensions. As the datasets become larger and more varied, adaptations to existing algorithms are required to maintain cluster quality and speed. Traditional clustering algorithms consider all of the dimensions of an input dataset in an attempt to learn as much as possible about each object described. In high dimensional data, however, many of the dimensions are often irrelevant. These irrelevant dimensions can confuse clustering algorithms by hiding clusters in noisy data. In very high dimensions it is common for all of the objects in a dataset to be nearly equidistant from each other, completely masking the clusters. Feature selection methods have been employed somewhat successfully to improve cluster quality. These algorithms find a subset of dimensions on which to perform clustering by removing irrelevant and redundant dimensions. Unlike feature selection methods which examine the dataset as a whole, subspace clustering algorithms localize their search and are able to uncover clusters that exist in multiple, possibly overlapping subspaces.

2.2.10.1 CLICK

CLICK seeks Identify highly homogeneous sets of elements - connectivity kernels. • Add elements to kernels via similarity to average kernel fingerprints. •Uses tools from graph theory and probabilistic considerations for similarity evaluation and kernel identification. • Efficient implementation

2.2.10.2CLIQUE

The CLIQUE Algorithm finds clusters by first dividing each dimension into xi equal-width intervals and saving those intervals where the density is greater than tau as clusters. Then each set of two dimensions is examined: If there are two intersecting intervals in these two dimensions and the density in the intersection of these intervals is greater than tau, the intersection is again saved as a cluster. This is repeated for all sets of three, four, five, dimensions. After every step adjacent clusters are replaced by a joint cluster and in the end all of the clusters are output

- Find all the dense areas in the one-dimensional spaces corresponding to each attribute.
  This is the set of dense one-dimensional cells.

- Set k = 2
- Generate all candidate dense k-dimensional cells from dense (k-1) dimensional cell.
- Eliminate cells that have fewer than points.
- Set k= k+ 1

  - Repeat steps until there are no candidate dense K dimensional cells.

  - Find clusters by taking the union of all adjacent, high density cells.

Chapter 3
K-Means Algorithm

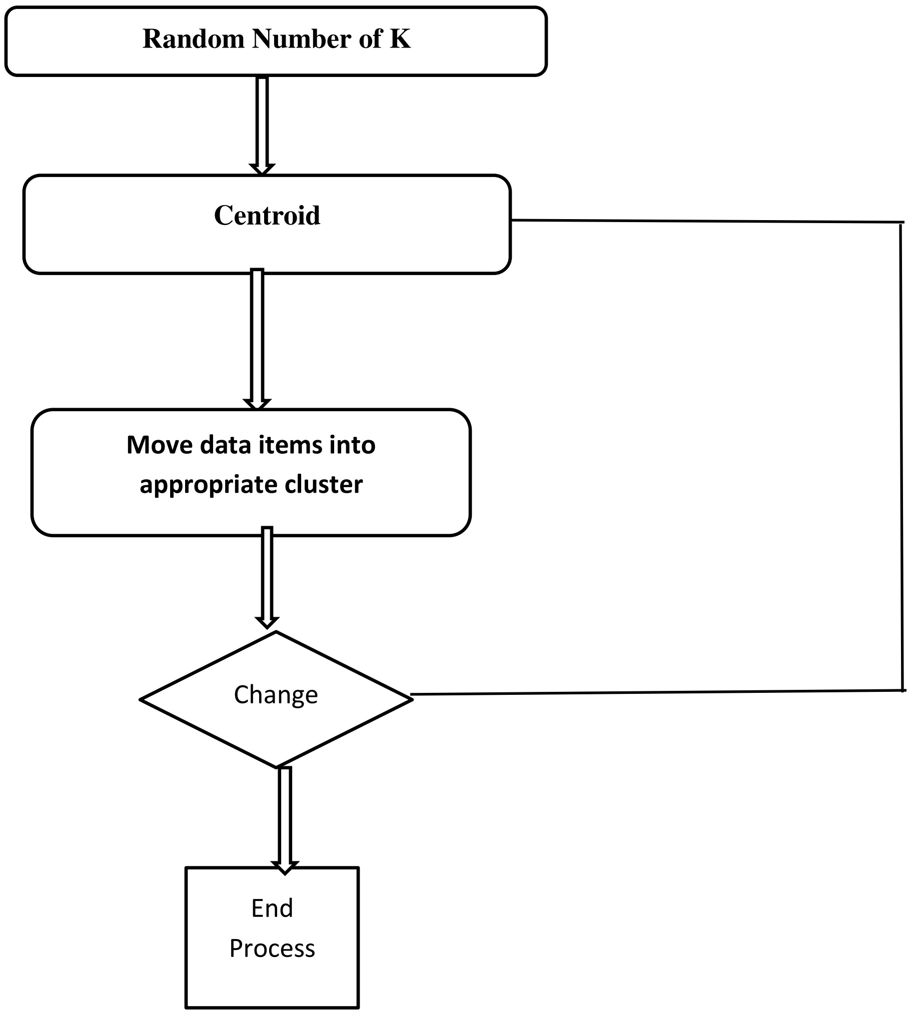
Introduction

CLUSTERING problems arise in many different applications, such as data mining and knowledge discovery data compression and vector quantization and pattern recognition and pattern classification. The notion of what constitutes a good cluster depends on the application and there are many methods for finding clusters subject to various criteria, both ad hoc and systematic. These include approaches based on splitting and merging such as ISODATA randomized approaches such as CLARA CLARANS methods based on neural nets , and methods designed to scale to large databases, including DBSCAN , BIRCH , and ScaleKM . For further information on clustering and clustering algorithms,

Among clustering formulations that are based on minimizing a formal objective function, perhaps the most widely used and studied is k-means clustering. Given a set of n data points in real d-dimensional space, R^d and an integer k, the problem is to determine a set of k points in R^d integer k, from each data point to its nearest center. This measure is

often called the squared-error distortion and this type of clustering falls into the general category of variance-based clustering. Clustering based on k-means is closely related to a number of other clustering and location problems. These include the Euclidean k-medians (or the multisource Weber problem) in which the objective is to minimize the sum of distances to the nearest center and the geometric k-center problem in which the objective is to minimize the maximum distance from every point to its closest center. There are no efficient solutions known to any of these problems and some formulations are NP-hard An asymptotically efficient approximation for the k-means clustering problem has been presented by Matousek but the large constant factors suggest that it is not a good candidate for practical implementation. One of the most popular heuristics for solving the k-means problem is based on a simple iterative scheme for finding a locally minimal solution. This algorithm is often called the k-means algorithm. There are a number of variants to this algorithm, so, to clarify which version we are using, we will refer to it as Lloyd's algorithm. (More accurately, it should be called the generalized Lloyd's algorithm since Lloyd's original result was for scalar data. Lloyd's algorithm is based on the simple observation that the optimal placement of a center is at the centroid of the associated cluster Given any set of k centers Z, for each center $z \epsilon Z$, let V (z) denote its neighborhood, that is, the set of data points for which z is the nearest neighbor. In geometric terminology, V(z) is the set of data points lying in the Voronoi cell of z . Each stage of Lloyd's algorithm moves every center point z to the centroid of V (z) and then updates V (z) by recomputing the distance from each point to its nearest center. These steps are repeated until some convergence condition is met. See Faber for descriptions of other variants of this algorithm. For points in general position (in particular, if no data point is equidistant from two centers), the algorithm will eventually converge to a point that is a local minimum for the distortion. However, the result is not necessarily a global minimum. for further discussion of its statistical and convergence properties. Lloyd's algorithm assumes that the data are memory resident. Bradley et al. have shown how to scale k-means clustering to very large data sets through sampling and pruning. Note that Lloyd's algorithm does not specify the initial placement of centers. See Bradley and Fayyad , for example, for further discussion of this issue.

Because of its simplicity and flexibility, Lloyd's algorithm is very popular in statistical analysis. In particular, given any other clustering algorithm, Lloyd's algorithm can be applied as a postprocessing stage to improve the final distortion. As we shall see in our experiments, this can result in significant improvements. However, a straightforward implementation of Lloyd's algorithm can be quite slow. This is principally due to the cost of computing nearest neighbors. In this paper, we present a simple and efficient implementation of Lloyd's algorithm, which we call the filtering algorithm. This algorithm begins by storing the data points in a kd-tree. Recall that, in each stage of Lloyd's algorithm, the nearest center to each data point is computed and each center is moved to the centroid of the associated neighbors. The idea is to maintain, for each node of the tree, a subset of candidate centers. The candidates for each node are pruned, or ªfiltered, º as they are propagated to the node's children. Since the kd-tree is computed for the data points rather than for the centers, there is no need to update this structure with each stage of Lloyd's algorithm. Also, since there are typically many more data points than centers, there are greater economies of scale to be realized. Note that this is not a new clustering method, but simply an efficient implementation of Lloyd's k-means algorithm.



K-means Algorithm Process

STEPS:

Step 1.Begins with a decision on the value of k= number of clusters.

Step 2. Put any initial partition that classifies the data into k clusters. You may assign the training samples randomly, or systematically as the following: Take the first k training sample as single-element clusters assign each of the remaining (N-k) training sample to the cluster with the nearest centroid. After each assignment, recomputed the centroid of the gaining cluster.

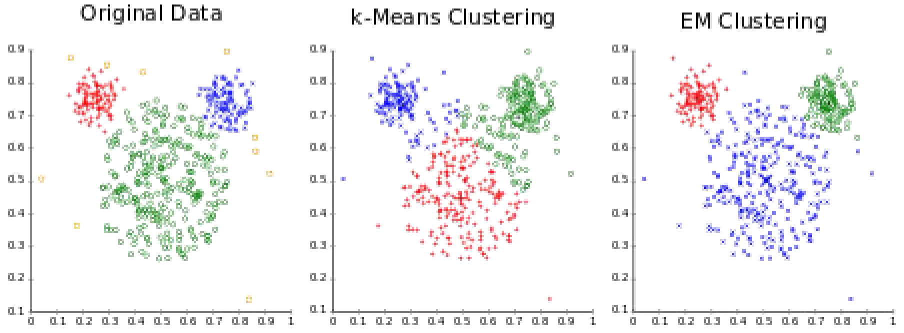
Step 3 .Take each sample in sequence and compute its distance from the centroid of each of the clusters. If a sample is not currently in the cluster with the closest centroid, switch this sample to that cluster and update the centroid of the cluster gaining the new sample and the cluster losing the sample.

Step 4 . Repeat step 3 until convergence is achieved, that is until a pass through the training sample causes no new assignments. If the number of data is less than the number of cluster then we assign each data as the centroid of the cluster. Each centroid will have a cluster number. If the number of data is bigger than the number of cluster, for each data, we calculate the distance to all centroid and get the minimum distance. This data is said belong to the cluster that has minimum distance from this data. Since we are not sure about the location of the centroid, we need to adjust the centroid location based on the current updated data. Then we assign all the data to this new centroid. This process is repeated until no data is moving to another cluster anymore. The algorithm works by using the following equation **.**

$$\boldsymbol{arg_s min = \sum_{i=1}^{k} \sum_{x_j \epsilon s_i} || x_j - u_i || \text{^}2}$$

The formula a given set of observations (X1,X2… Xn) where each observation represent an element of the cluster with a d-dimensional real vector , k-means clustering aims to partition and the n observations into k sets (k <= n) S = { S1, S2, ….. Sk) that's to minimize the cluster where μi is the mean of points in Si .

Different cluster analysis results on "mouse" data set:



Algorithm :

The k -means algorithm for partitioning , where each cluster center is represented by the mean value of the objects in the cluster .

Input:

- K: the number of clusters .
- D: a data set containing n objects

Output: A set of k clusters .

Method:

1> Arbitrarily choose k objects from D as the initial clusters centers ;
2> Repeat
3> Reassign each object to cluster to which is the most similar based on the mean value of the objects in the cluster.
4> Update the cluster means ,…, calculate the mean value of the objects for each cluster;
5> Until no change square error;

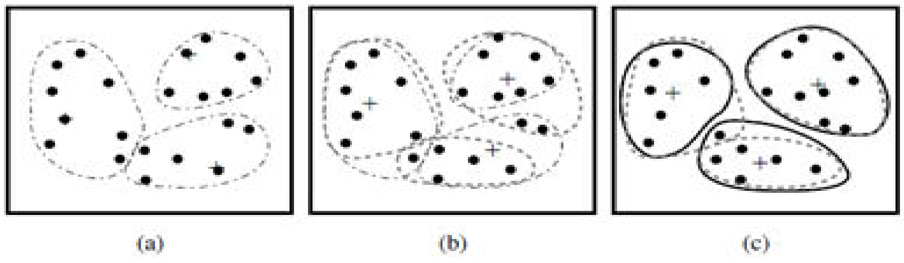


Figure 7.3 Clustering of a set of objects based on the *k*-means method. (The mean of each cluster is marked by a "+".)

Example : A K-Means clustering step by step

As a simple illustration of a k-means algorithm, consider the following data set consisting of the scores of two variables on each of seven individuals:

Steps	A	B
1	1.0	1.0
2	1.5	2.0
3	3.0	4.0
4	5.0	7.0
5	3.5	5.0
6	4.5	5.0
7	3.5	4.5

This data set is to be grouped into two clusters. As a first step in finding a sensible initial partition, let the A & B values of the two individuals furthest apart (using the Euclidean distance measure), define the initial cluster means, giving:

Groups	Individual	Mean Vector Centroid
Group 1	1	{1.0,1.0}
Group 2	4	{5.0,7.0}

The remaining individuals are now examined in sequence and allocated to the cluster to which they are closest, in terms of Euclidean distance to the cluster mean. The mean vector is recalculated each time a new member is added. This leads to the following series of steps:

	CLUSTER 1			CLUSTER 2
STEPS	INDIVIDUALS	MEAN VECTOR CENTROIDS	INDIVIDUALS	MEAN VECTOR CENTROIDS
1	1	(1.0,1.0)	4	(5.0,7.0)
2	1,2	(1.2,1.5)	4	(5.0,7.0)
3	1,2,3	(1.8,2.3)	4	(5.0,7.0)
4	1,2,3	(1.8,2.3)	4,5	(4.2,6.0)
5	1,2,3	(1.8,2.3)	4,5,6	(4.3,5.7)
6	1,2,3	(1.8,2.3)	4,5,6,7	(4.1,5.4)

Now the initial partition has changed, and the two clusters at this stage having the following characteristics:

	INDIVIDUALS	**MEANS VECTOR CENTROIDS**
CLUSTER 1	1,2,3	(1.8,2.3)
CLUSTER 2	4,5,6,7	(4.1,5.4)

But we cannot yet be sure that each individual has been assigned to the right cluster. So, we compare each individual's distance to its own cluster mean and to that of the opposite cluster. And we find:

INDIVIDUAL	**DISTANCE MEAN (CENTROIDOF CLUSTER 1)**	**DISTANCE MEAN (CENTROIDOF CLUSTER 2)**
1	1.5	5.8
2	0.4	4.3
3	2.1	2.8
4	5.7	2.8
5	3.2	0.7
6	3.8	0.6
7	2.8	1.1

Only individual 3 is nearer to the mean of the opposite cluster (Cluster 2) than its own (Cluster 1). In other words, each individual's distance to its own cluster mean should be smaller that the distance to the other cluster's mean (which is not the case with individual 3). Thus, individual 3 is relocated to Cluster 2 resulting in the new partition:

	INDIVIDUALS	**MEAN VECTORS (CENTROID)**
CLUSTER 1	**1,2**	**(1.3,1.5)**
CLUSTER 2	**3,4,5,6,7**	**(3.9,5.1)**

Chapter 4:
Survey of Literature

Introduction

Clustering is defined as dividing input data sets called clusters. As unsupervised, data clustering tasks have been exploited in many fields including image processing, machine learning, data mining, biochemistry and bioinformatics. Depending on the data properties or the purpose of clustering, different types of clustering algorithms have been developed, such as, partitioned, hierarchical, graph-based clustering etc. Most of the clustering task requires iterative procedures to find locally or globally optimal solutions from high dimensional data sets. In addition, very rarely real-life data has unique clustering solution and it is also hard to interpret the cluster representations. Therefore, it requires many experimentations with different algorithms and hence it is computational complexity is a significant issue for the clustering algorithms. Therefore, upgrading the clustering technique to have unique clustering and reducing the dimensionality become emerging trend in current scenario, which gives rise to different approach of algorithms. This chapter focuses a literature review on K-Means clustering algorithm and its modification and hybridization with Fuzzy and Rough Set to reduce the traditional drawbacks. The literatures about clustering algorithms on various . distance, which differs based on area of applications. The remaining of this chapter is as follows: Section 2.2 gives the general information about K-Means and its modifications. Outlines on the Fuzzy C-Means and its modifications are in Section 2.3.Section2.4illustratesthe hybrid approach of K-Means, Fuzzy C-Means and Rough set for performance upgradient.

The corresponding criterion of each property of big data:

Type of Dataset: The collected data in the real world often contain both numeric and categorical attributes. Clustering algorithms work effectively either on purely numeric data or on purely categorical data; most of them perform poorly on mixed categorical and numerical data types.

Size of Dataset: The size of the dataset has a major effect on the clustering quality. Some clustering methods are more efficient clustering methods than others when the data size is small, and vice versa.

Input Parameter: A desirable feature for practical clustering is the one that has fewer parameters, since a large number of parameters may affect cluster quality because they will depend on the values of the parameters.

Handling Noisy Data: A successful algorithm will often be able to handle noisy data. Also, noise makes it difficult for an algorithm to cluster an object into a suitable cluster. Time Complexity: Most of the clustering methods must be used several times to improve the clustering quality. Therefore if the process takes too long, then it can become impractical for applications that handle big data.

Stability: One of the important features for any clustering algorithm is the ability to generate the same partition of the data irrespective of the order in which the patterns are presented to the algorithm.

Types of learning method in data mining In data mining, two learning methods used to mine data i.e. supervised learning and unsupervised learning. Supervised learning: In this learning, data includes together the input and the desired result. It is the fast and perfect learning method. The accurate results are known and are given in inputs to the model during learning procedure. Neural network, Multilayer perception, Decision tree are supervised models. Unsupervised learning: The desired result is not provided to the unsupervised model during learning procedure. This method can be used to cluster the input data in classes on the basis of their statistical properties only. These models are for various types of clustering, k-means, distances and normalization, self-organizing maps.

K-Means clustering K-means clustering algorithm is famous clustering technique. It is used in many areas such as information retrieval, computer vision and pattern recognition. K-means clustering assigns n data points into k clusters so that similar data points can be grouped together. It is an iterative method which assigns each point to the cluster whose centroid is the nearest. Then it again calculates the centroid of these groups by taking its average. Properties of k-means algorithm :

1. Large data set are efficiently processed.
2. It often terminates at a local optimum.
3. It supports numeric values.

Clustering through K-Means:

Clustering has been used in a number of applications such as engineering, biology, medicine and data mining. The most popular clustering algorithm used in several field is K-Means since it is very simple and fast and efficient. K-means is developed by Mac Queen . The K-Means algorithm is effective in producing cluster for many practical applications. But the computational complexity of the original K-Means algorithm is very high, especially for large datasets. The K-Means algorithm is a partition clustering method that separates data into K groups. Main drawback of this algorithm is that of a priori fixation of number of clusters and seeds. To rectify the drawbacks of traditional K-Means clustering algorithm various measures were carried out in recent years by the researcher from multiple fields. Because of the modified approaches, cluster analyses were upgraded in multiple dimensions such as increasing the performance, decreasing the computational complexity, reducing the cluster error and to increase cluster uniqueness.

Chapter 5:
To Study The Efficiency of K-Means Algorithm for Partitional Clustering

Clustering is an unsupervised technique used in discovering inherent structure present in the set of patterns. In fact, clustering techniques aims to extract the groups present in a given data set &amp; each such group is termed as a cluster.

Let the set of patterns be S= $\{x_1 x_2, \ldots\ldots\ldots\ldots x_n\}$ belongs to R m where

x_i is the ith pattern vector.

n is the total number of patterns.

m is the dimensionality of the feature space.

Let the total no. of clusters be k. If the clusters are represented by $c_{1,} c_2, \ldots\ldots\ldots\ldots\ldots\ldots c_k$, then we assume

$$P1\,, C_i\ \neq \emptyset\,, for\ i\ =\ 1,2, \ldots\ldots\ldots K$$
$$P2\,,\ \ C_i \cap C_J\ =\ \emptyset\, for\ i\ \neq\ j\, and$$
$$P3\,, \bigcup_{i=1} k\ \ C_i\ =\ S\ where\ \emptyset\ represents\ the\ null\ set$$

Clustering techniques broadly be divided into two categories: Hierarchical &amp; non-hierarchical. The nonhierarchical of partitional clustering problem deals with obtaining a optimal partition of S into k subsets such that some clustering citation is satisfied. Among the partitional techniques the k-means algorithm has been one of the most widely used algorithms. Here the value of k needs to be known as priori. The principle used for clustering by k-means algorithm is to minimize the sum of intra-class distances to get the optimal clusters. Mathematically the principle has been stated below.

1. $C_{1,} C_2, \ldots\ldots\ldots C_k$ be a set of k clusters of s.
2. Let Z j = (Σ x) / # C_j
3. 3. Let f (C 1 ,C 2 ,…………,C k) = Σ Σ || X – Z j || 2

$f(C_1, C_2, \ldots\ldots\ldots\ldots C_k)$ is referred as the objective function of the clustering $(C_{1,} C_2, \ldots\ldots\ldots\ldots C_k)$

4 . Minimize $f(C_{1,} C_2, \ldots\ldots\ldots\ldots C_k)$ over all such $C_1, C_2, \ldots\ldots\ldots\ldots C_k$ where $(C_1, C_2, \ldots\ldots\ldots C_k)$ satisfy $(P_1, P_2$ started earlier

All possible clustering of S are to be considered to get the optimal $(C_1, C_2, \ldots\ldots\ldots\ldots C_K)$ So obtaining the exact solution of the problem is theoretically possible, yet not feasible in practice due to limitations of computer storage and time one requires the evaluation of S(n,k) partitions if exhaustive enumeration is used to solve the problem, where

$$S_k = {}^{1}/_{k}! \sum_{j=1}^{K} (-1)^{K=j} (K_j)\, J^n$$

This clearly indicates that exhaustive enumeration cannot lead to the required solution for most practical problem in reasonable computation time. Thus, approximate heuristic techniques seeking a compromise or looking for an acceptable solution have usually been adopted. One such method is Fogy’s K-means algorithm.

Demerits of K-means algorithm

- The algorithm fails to provide the natural routine presents in a human data set having non spherical cluster or clusters that develops along 1 and 2 principle exists where the dimensionality of the feature space is 3 or more.
- It has been reported in the literature that the algorithm stuck at local minimum which is not necessarily the global one.
- The behavior of K-means algorithm is influenced by
- The number of cluster center specified.
- The choice of initial cluster centers.
- The order in which the samples are taken, &amp;
- The geometrical properties of data.

Chapter 6

Experimental Result

Hardware and Software/equipment necessary to solve the problem

- OS – Windows 7/8.1 etc.

The necessary hard-wares are:

- Pentium IV/ Celeron Processor
- 128 MB RAM
- Hard Disk (40 GB)
- Keyboard
- Mouse, etc
- For programming purpose, I have used Visual Basic. and the necessary soft wares are
- Turbo C++ compiler

The main objective of the study to show experimentally that the result of the proposed algorithm i.e. K-means algorithm with the execution time taken. The algorithm K-means was run (till the convergence was achieved) on synthetic & real-life data set. Several initial configurations are used for the standard version of the algorithm, to achieve the results. shows the synthetic data distribution of size 300, generated randomly from four clusters. the result of the experiment when the synthetic data of is processed by K-means with 15 different initial configurations of seed points. The partitioning of synthetic data by K-means algorithm for the objective function value .

TABLE 1 : Results for Synthetic data set 1

Initial Configuration	Value of objective function	Percentage (%) of datapoints misclassified
1	**13054.5**	0
2	**13054.5**	0
3	**13054.5**	0
4	**13054.5**	0
5	**13054.5**	0
6	**13054.5**	0
7	**13054.5**	0
8	**13054.5**	0
9	**15342.5**	4.2
10	**13054.5**	0
11	**13054.5**	0
12	1756.9	0
13	**1543.2**	5.9
14	**13054.5**	0
15	**13054.5**	0

TABLE 2 : Result for Synthetic data set 2

Initial configuration	Value of objective function	Percentage (%) of data points misclassified
1	**13054.5**	**51.6**
2	**13054.5**	**51.6**
3	**13054.5**	**51.6**
4	**13054.5**	**51.6**
5	**13054.5**	**51.6**
6	**13054.5**	**51.6**
7	**13054.5**	**51.6**
8	**13054.5**	**51.6**
9	**15472.5**	**72.2**
10	**13054.5**	**51.6**
11	**13054.5**	**51.6**
12	1756.9	**64.5**
13	**1543.2**	**51.6**
14	**13054.5**	**51.6**
15	**13054.5**	**51.6**

Table 3 Result for Synthetic data set 3:

Initial configuration	Value of objective function	Percentage (%) of data points misclassified
1	**13159.1**	**0**
2	**13159.1**	**0**
3	**13159.1**	**0**
4	**13159.1**	**0**
5	**13159.1**	**0**
6	**13159.1**	**0**
7	**13159.1**	**0**
8	**13159.1**	**0**
9	**18822.5**	**7.1**
10	**13159.1**	**0**
11	**13159.1**	**0**
12	**16671.9**	**6.5**
13	**13159.1**	**0**
14	**13159.1**	**0**
15	**13159.1**	**0**

Note that the synthetic data set 1 is having 1000 points generated from 2 clusters, each one having 500 points. The first cluster is having a shape of square and the other one is having the shape of a Circular Disk. It is evident from experimental result shown in **Table 1** that the K-Means algorithm can extract the appropriate clustering for this type of data where the clusters exhibits characteristics pocket that are well separated from each other.

Note that the synthetic data set 2 is having 1000 points generated from 2 clusters each one having 500 points. The first cluster is having a shape of an alphabet 'C' and the other one is having a shape of inverted 'C'. It is evident from the experimental result shown in **Table 2** that the K-Means algorithm cannot extract the appropriate clustering for this type of data where the clusters are overlapping in nature.

Note that the synthetic data set 3 is having 1000 points generated from 2 clusters each one having 500 points. The first cluster is having the shape of a triangle and the other one is having the shape of a rectangle. It is evident from experimental result shown in **Table 3** that the K-means algorithm can extract the appropriate clustering for this type of data where the clusters exhibits characteristics pocket that are well separated from each other.

Chapter 5
Conclusion

CONCLUSION

In this work, basic K-means algorithm has been discussed with an appropriate example. Then we have pointed out the limitation of k-means technique of having more computational complexity, and solution for the same. A drawback is present in the k means algorithms, fixed number of clusters and again empty cluster problem are subsist. K-MEANS algorithm is very simple algorithm to execute. So, it is widely popular. It takes very small amount of time to provide the clustering result in any data set of moderate size. It is found that this algorithm provides desired clustering or expected result for data sets of various size & shape, where the data exhibit characteristics pockets which are relatively far from each other. It has also been found that for data which exhibit characteristic pockets, this algorithm mostly provides global optimum solution. But this algorithm has several disadvantages too. The algorithm fails to provide the natural groups present in a real life data set having non-spherical cluster or clusters that develops along 1 and 2 principle axis where the dimensionality of the feature space is 3 or more. We have performed several experiments with K-means algorithm for three synthetic data sets. And we have observed the following.

For synthetic data set 1 where the data set is having 1000 points generated from 2 clusters, each one having 500 points. The first cluster is having a shape of square and the other one is having the shape of a Circular Disk. It is evident from experimental result shown in **Table 1** that the K-Means algorithm can extract the appropriate clustering for this type of data where the clusters exhibits characteristics pocket that are well separated from each other.

For synthetic data set 2 where the data set is having 1000 points generated from 2 clusters each one having 500 points. The first cluster is having a shape of an alphabet ‘C’ and the other one is having a shape of inverted ‘C’. It is evident from the experimental result shown in **Table 2** that the K-Means algorithm cannot extract the appropriate clustering for this type of data where the clusters are overlapping in nature.

For synthetic data set 3 where the data set is having 1000 points generated from 2 clusters each one having 500 points. The first cluster is having the shape of a triangle and the other one is having the shape of a rectangle. It is evident from experimental result shown in **Table 3** that the K-means algorithm can extract the appropriate clustering for this type of data where the clusters exhibits characteristics pocket that are well separated from each other.

BIBLIOGRAPHY

REFERENCES:

[1] Anderberg, M.R., Cluster Analysis for Application, Academic Press, Inc., NewYork, 1973.

[2] Jain, A.K. and R.C. Dubes., *Algorithms for Clustering Data*. Prentice-Hall, Englewood Cliffs, NJ, 1988.

[3] Devijver, P.A. and J. Kittler., *Pattern Recognition: A statistical Approach*, Prentice-Hall International. Hemel Hemstead, Hertfordshire, UK, 1982

[4] Spath, H., Cluster Analysis Algorithms. Ellis Horwood, Chichester. UK, 1980

[5] Tou, T.J. and C.R. Gonzalez., Pattern Recognition Principles. Addison-Wesley, Reading, MA, 1974.

[6] Jiawei Han, Micheline Kamber and Jian Pei , Data Mining: Concepts and Techniques, 3rd ed

[7] Irani, Jasmine, Nitin Pise, and Madhura Phatak. "Clustering Techniques and the Similarity Measures used in Clustering: A Survey." International Journal of Computer Applications 134, no. 7

[8] Murthy, C.A. and Chowdhury N., In search of optimal clusters using Genetic Algorithms, *Pattern Recognition Letters*, 17(8), 1996, 825-832.

[9] Selim, S.Z. and M.A. Ismail., K-means type algorithms: A generalized convergence theorem and characterization of local optimality. IEEE Trans. Pattern Anal. Mach. Intell. 6(1), 1984, 81-87.

[10] Liu, Hsiang-Chuan, Wen-Pei Sung, and Wenli Yao. Information, Computer and Application Engineering. CRC Press