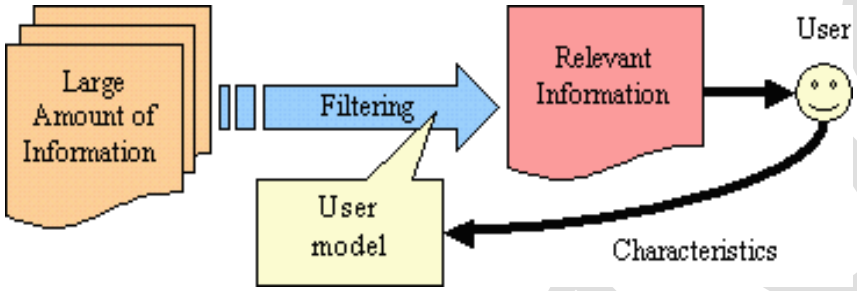
UNIT V DOCUMENT TEXT MINING

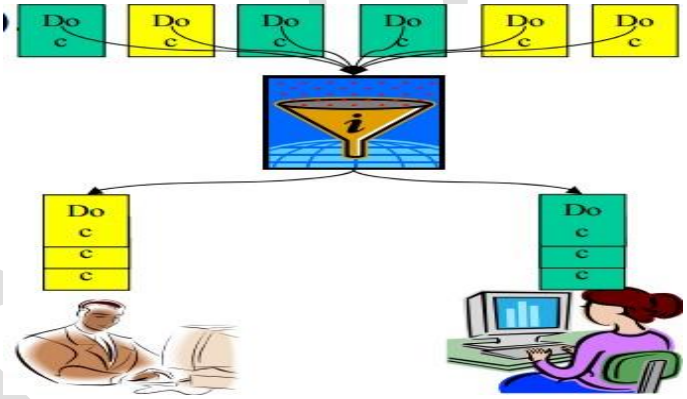
Recommender Systems Functions – Data and Knowledge Sources – Recommendation Techniques – Basics of Content-based Recommender Systems – High Level Architecture – Advantages and Drawbacks of Content-based Filtering – Collaborative Filtering – Matrix factorization models – Neighborhood models.

5.1 Information Filtering

An Information filtering system is a system that removes redundant or unwanted information from an information stream using (semi)automated or computerized methods prior to presentation to a human user. Its main goal is the management of the information overload and increment of the semantic signal-to-noise ratio.



Information Filtering is the process of monitoring large amounts of dynamically generated information and pushing to a user the subset of information likely to be of her/his interest (based on her/his her /his information needs)



An IFS needs an information filter that when applied to an information item, evaluates whether the item is of interest or not to the considered user.

Information filter: it represents the user's interests and as such has the goal of identifying only those pieces of information that a user would find interesting

- a process of selecting things from a larger set of possibilities then presenting them in a prioritized order (Malone et al. 1987).
  Ex: commercial recommendations
- a field of study designed for creating a systematic approach to extracting information that a particular person finds important from a larger stream of information (Canavese 1994, p.2).
  Ex: news stream filterin
- "tools … which try to filter out irrelevant material" (Khan & Card 1997, p.305) Ex: spam filtering

Characteristics

1. It involve large amount of data

2. It deals with textual information & it is applicable for unstructured or semi structured data
3. Filtering is based on descriptions of individual or group information preferences, often called profiles. Such profiles typically represent long-term interests.
4. Filtering is often meant to imply the removal of data from an incoming stream, rather than finding data in that stream.
5. Types are
    a. Content-based filtering - Objects to be filtered: generally texts Filter engine based on content analysis
    b. Collaborative filtering - Objects to be filtered: products/goods Filter engine based on usage analysis
    c. Hybrid Filtering - Combination of the two previous approaches Combination of the two previous approaches

Difference between Information Filtering and Information Retrieval

Information Filtering	**Information Retrieval**
Information Filtering is about processing a stream of information to match your static set of likes, tastes and preferrences.	Information retrieval is about fulfilling immediate queries from a library of information available.
Example : a clipper service which reads all the news articles published today and serves you content that is relevant to you based on your likes and interests.	**Example :** you have a deal store containing 100 deals and a query comes from a user. You show the deals that are relevant to that query.
information filtering is concerned with repeated uses of the system, by a person or persons with long-term goals or interests	IR is typically concerned with single uses of the system, by a person with a one-time goal and one-time query
filtering assumes that profiles can be correct specifications of information interests	IR recognizes inherent problems in the adequacy of queries as representations of information needs
filtering is concerned of with the distribution texts to groups or individuals.	IR is concerned with the collection and organization of texts
filtering is mainly concerned with selection or elimination of texts from a dynamic datastream.	IR is typically concerned with the selection of texts from a relatively static database
filtering is concerned with long-term changes over a series of information- seeking episodes.	IR is concerned with responding to the user‟s interaction with texts within a single information-seeking episode
Models Probabilistic model	Models Boolean IR model Vector space IR model Probabilistic IR model Language Model

5.2 Organization and Relevance feedback Refer unit –ii for relevance feedback

5.3 Text Mining

Text mining is known as Text Data Mining and knowledge discovery in textual database
. A process of identifying novel information from a collection of texts. Text Mining aims to extract useful knowledge from text documents

Text mining can be visualized as consisting of 2 phases. Text refining that transforms free-form text documents into a chosen intermediate form , and knowledge distillation that deduce patterns or knowledge from the intermediate form.

Approaches

1. Keyword based - Relies on IR techniques
2. Tagging - Manual tagging ,Automatic categorization
3. Information extraction - Natural Language Processing (NLP)

Applications

1. Spam filtering.
2. Creating suggestion and recommendations (like amazon)
3. Monitoring public opinions (for example in blogs or review sites)
4. Customer service, email support.
5. Automatic labeling of documents in business libraries.

Difference between Text Mining and Data Mining

Text Mining	**Data Mining**
The discovery of knowledge sources that contain text or unstructured information is called "text mining".	the discovery of knowledge from structured data (data contained in structured databases or data warehouses.)
text mining consists of a **preprocessing step to structure the data**	It process directly

5.4 Text classification and Clustering

Motivation for Text Categorization

- Suppose you want to buy a cappuccino maker as a gift
  - try Google for "cappuccino maker"
  - try "Yahoo! Shopping" for "cappuccino maker"
- Observations
  - Broad indexing & speedy search alone are not enough.
  - Organizational view of data is critical for effective retrieval.
  - Categorized data are easy for user to browse.
  - Category taxonomies become most central in well-known web sites (Yahoo!, Lycos, ...).

Text Categorization Applications

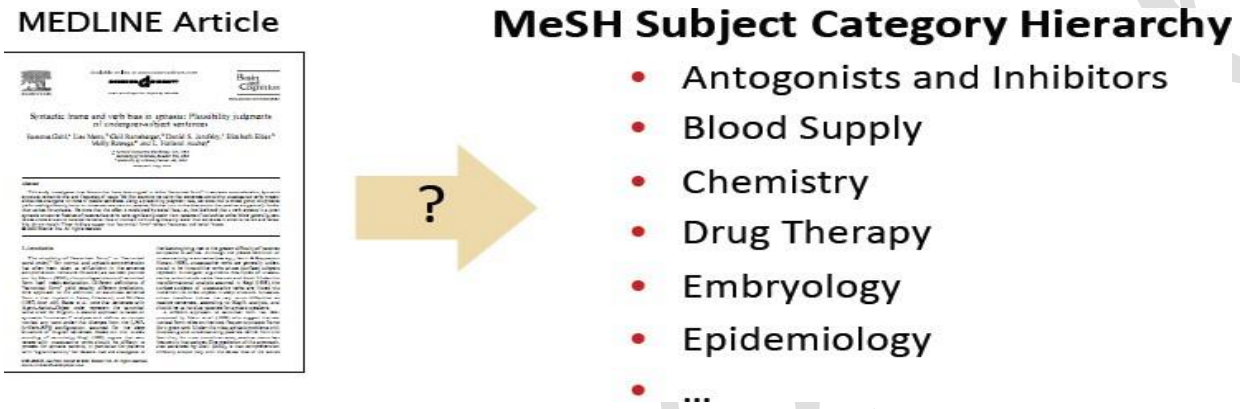
- Web pages organized into category hierarchies
- Journal articles indexed by subject categories (e.g., the Library of Congress, MEDLINE, etc.)
- Responses to Census Bureau occupations
- Patents archived using International Patent Classification
- Patient records coded using international insurance categories
- E-mail message filtering
- News events tracked and filtered by topics

Definition : Text classification

- The goal of text categorization is the classification of documents into a fixed number of predefined categories. Each document can be in multiple, exactly one, or no category at all.
- Text classification is to classify a set of documents as:
  - Assign subject categories, topics etc
  - Spam detection
  - Authorship identification
  - Age /gender identification
  - Language identification
  - Sentiment analysis

    ….

Example: Finding the subject of this article



5.5 Categorization/Classification Algorithms

Classification Methods

1. Manual classification

- Used by the original Yahoo! Directory
- ODP , Looksmart, about.com, PubMed
- Very accurate when job is done by experts
- Consistent when the problem size and team is small
- Difficult and expensive to scale - means we need automatic classification methods for big problems

2. Automatic document classification

- Hand-coded rule-based systems
  - One technique used by spam filters, Reuters, etc.
  - It‟s what Google Alerts is doing –
  - Widely deployed in government and enterprise
  - E.g., assign category if document contains a given boolean combination of words
  - Standing queries: Commercial systems have complex query languages
  - Accuracy is often very high if a rule has been carefully refined over time by a subject expert
  - Building and maintaining these rules is expensive

Standing queries Example : The path from IR to text classification:
You have an information need to monitor, say:" Unrest in the Niger delta region" . You want to rerun an appropriate query periodically to find new news items on this topic You will be sent new documents that are found I.e., it‟s text classification, not ranking.

Such queries are called standing queries. Long used by "information professionals". Standing queries are (hand-written) text classifiers.

3. Supervised Classification

Learning: We are given a collection X, and we want to learn some function γ over X Classification: γ(d) tells you the class of document d
Clustering: γ(d) tells you the cluster in which d belongs

Supervised learning: We have examples of some γ(d) and we want to compute γ for the rest Classification
Unsupervised learning: We only work on raw data. We don"t know the value of any γ(d)
 Clustering

Example: Classification Alg

1. k-Nearest Neighbors (simple, powerful)
2. Naive Bayes (simple, common method)
3. Support-vector machines (newer, more powerful) …..

5.5.1 Naïve Bayesian Classification:

Introduction

Bayesian classifiers are statistical classifiers. They can predict class membership probabilities, such as the probability that a given sample belongs to a particular class.

Bayesian classifier is based on Bayes" theorem. Naive Bayesian classifiers assume that the effect of an attribute value on a given class is independent of the values of the other attributes. This assumption is called class conditional independence. It is made to simplify the computation involved and, in this sense, is considered "naive"(immature / raw).

Bayes' Theorem

According to Bayes" theorem, the probability that we want to compute P (H|X) can beexpressed in terms of probabilities P (H), P (X|H), and P (X) as

$$P(H|\mathbf{X}) = \frac{P(\mathbf{X}|H)\ P(H)}{P(\mathbf{X})},$$

and these probabilities may be estimated from the given data.

Let X = {x1, x2, . . . , xn} be a sample, whose components represent values made on a set of n attributes. In Bayesian terms, X is considered "evidence".

Let H be some hypothesis, such as that the data X belongs to a specific class C. For classification

problems, our goal is to determine

P (H|X), the probability that the hypothesis H holds given the "evidence", (i.e. the observed data sample X). In other words, we are looking for the probability that sample X belongs to Class C, given that we know the attribute description of X.

P (H|X) □ is the a posterior probability of H conditioned on X.

Fox example, suppose our data samples have attributes: age and in-come, and that sample X is a 35-year-old customer with an income of $40,000. Suppose that H is the hypothesis that our customer will buy a computer. Then P (H|X) is the probability that customer X will buy a computer given that we know the customer‟s age and income.

P (H) □is the a priori probability of H. For our example, this is the probability that any given customer will buy a computer, regardless of age, income, or any other information.

P (X|H) □ is the a posteriori probability of X conditioned on H. That is, it is the probability that a customer X, is 35 years old and earns $40,000, given that we know the customer will buy a computer.

P (X) □is the a priori probability of X. In our example, it is the probability that a person from our set of customers is 35 years old and earns $40,000.

Naive Bayesian Classifier

The naive Bayesian classifier works as follows:

1. Let T be a training set of samples, each with their class labels. There are k classes, C1, C2, . . . , Ck. Each sample is represented by an n-dimensional vector, X = {x1, x2, . . . , xn}, depicting n measured values of the n attributes, A1, A2, . . . , An, respectively.

2. Given a sample X, the classifier will predict that X belongs to the class having the highest a posteriori probability, conditioned on X . That is X is predicted to belong to the class Ci if and only if

$$P(C_i|\mathbf{X}) > P(C_j|\mathbf{X}) \qquad \text{for } 1 \leq j \leq m,\ j \neq i.$$

Thus we find the class that maximizes P (Ci|X). The class Ci for which P (Ci|X) is maximized is called the maximum posteriori hypothesis. By Bayes‟ theorem

$$P(C_i|\mathbf{X}) = \frac{P(\mathbf{X}|C_i)\ P(C_i)}{P(\mathbf{X})}.$$

3. As P (X) is the same for all classes, only P (X|Ci)P (Ci) need be maximized. If the class a priori probabilities, P (Ci), are not known, then it is commonly assumed that the classes are equally likely, that is, P (C1) = P (C2) = . . . = P (Ck), and we would therefore maximize P (X|Ci). Otherwise we maximize P (X|Ci)P (Ci). Note that the class a priori probabilities may be estimated by P (Ci) = freq(Ci, T)/|T |.
4. Given data sets with many attributes, it would be computationally expensive to compute P (X|Ci). In order to reduce computation in evaluating P (X|Ci) P (Ci), the naive assumption of class conditional independence is made. This presumes that the values of the attributes are conditionally independent of one another, given the class label of the sample. Mathematically this means that

$$P(\mathbf{X}|C_i) \approx \prod_{k=1}^{n} P(x_k|C_i).$$

The probabilities P (x_1|Ci), P (x_2|Ci), . . . , P (x_n|Ci) can easily be estimated from the training set. Recall that here x_k refers to the value of attribute Ak for sample X.

- If A_k is categorical, then P (x_k|Ci) is the number of samples of class Ci in T having the value x_k for attribute A_k, divided by freq(Ci, T), the number of sample of class Ci in T .
- If A_k is continuous-valued, then we typically assume that the values have a Gaussian distribution with a mean μ and standard deviation σ defined by

$$g(x, \mu, \sigma) = \frac{1}{\sqrt{2\pi\sigma}} \exp -\frac{(x-\mu)^2}{2\sigma^2},$$

so that

$p(x_k|C_i) = g(x_k, \mu_{Ci}, \sigma_{Ci})$.

We need to compute μ_{Ci} and σ_{Ci} , which are the mean and standard deviation of values of attribute A_k for training samples of class C_i.

5. In order to predict the class label of X, P (X|Ci)P (Ci) is evaluated for each class Ci. The classifier predicts that the class label of X is Ci if and only if it is the class that maximizes P (X|Ci)P (Ci).

Example 1 : Using the Naive Bayesian Classifier

We will consider the following training set. The data samples are described by attributes age, income, student, and credit. The class label attribute, buy, tells whether the person buys a computer, has two distinct values, yes (class C_1) and no (class C_2).

RID	Age	Income	student	credit	Ci: buy
1	Youth	High	no	fair	C2: no
2	Youth	High	no	excellent	C2: no
3	middle-aged	High	no	fair	C1: yes
4	Senior	medium	no	fair	C1: yes
5	Senior	Low	yes	fair	C1: yes
6	Senior	Low	yes	excellent	C2: no
7	middle-aged	Low	yes	excellent	C1: yes
8	Youth	medium	no	fair	C2: no
9	Youth	Low	yes	fair	C1: yes
10	Senior	medium	yes	fair	C1: yes
11	Youth	medium	yes	excellent	C1: yes
12	middle-aged	medium	no	excellent	C1: yes
13	middle-aged	High	yes	fair	C1: yes
14	Senior	medium	no	excellent	C2: no

The sample we wish to classify is

X = (age = youth, income = medium, student = yes, credit = fair)

We need to maximize P (X|Ci)P (Ci), for i = 1, 2. P (Ci), the a priori probability of each class, can be estimated based on the training samples:

P(buy =yes) = 9 /14
P(buy =no) = 5 /14

To compute P (X|Ci), for i = 1, 2, we compute the following conditional probabilities:

P(age=youth | buy =yes) = 2/9 P(income =medium | buy =yes) = 4/9 P(student =yes | buy =yes) = 6/9 P(credit =fair | buy =yes) = 6/9

P(age=youth | buy =no) = 3/5 P(income =medium | buy =no) = 2/5 P(student =yes | buy =no) = 1/5 P(credit =fair | buy =no) = 2/5

Using the above probabilities, we obtain

$$\begin{aligned} P(\mathbf{X}|buy = yes) &= P(age = youth|buy = yes) \\ &\quad P(income = medium|buy = yes) \\ &\quad P(student = yes|buy = yes) \\ &\quad P(credit = fair|buy = yes) \\ &= \frac{2}{9}\frac{4}{9}\frac{6}{9}\frac{6}{9} = 0.044. \end{aligned}$$

Similarly

$$P(\mathbf{X}|buy = no) = \frac{3}{5}\frac{2}{5}\frac{1}{5}\frac{2}{5} = 0.019$$

To find the class that maximizes P (X|Ci)P (Ci), we compute

$$P(\mathbf{X}|buy = yes)P(buy = yes) = 0.028$$

$$P(\mathbf{X}|buy = no)P(buy = no) = 0.007$$

Thus the naive Bayesian classifier predicts buy = yes for sample X.

Example 2:

Predicting a class label using naïve Bayesian classification. The training data set is given below. The data tuples are described by the attributes Owns Home?, Married, Gender and Employed. The class label attribute Risk Class has three distinct values. Let C1 corresponds to the class A, and C2 corresponds to the class B and C3 corresponds to the class C.

The tuple is to classify is,

X = (Owns Home = Yes, Married = No, Gender = Female, Employed = Yes)

Owns Home	Married	Gender	Employed	Risk Class
Yes	Yes	Male	Yes	B
No	No	Female	Yes	A
Yes	Yes	Female	Yes	C
Yes	No	Male	No	B
No	Yes	Female	Yes	C
No	No	Female	Yes	A
No	No	Male	No	B
Yes	No	Female	Yes	A

No	Yes	Female	Yes	C
Yes	Yes	Female	Yes	C

Solution

There are 10 samples and three classes.

Risk class A = 3
Risk class B = 3
Risk class C = 4

The prior probabilities are obtained by dividing these frequencies by the total number in the training data,

P(A) = 3/10 = 0.3
P(B) = 3/10 = 0.3
P(C) = 4/10 = 0.4

To compute P(X/Ci) =P {yes, no, female, yes}/Ci) for each of the classes, the conditional probabilities for each:

P(Owns Home = Yes/A) = 1/3 =0.33
P(Married = No/A) = 3/3 =1
P(Gender = Female/A) = 3/3 = 1
P(Employed = Yes/A) = 3/3 = 1

P(Owns Home = Yes/B) = 2/3 =0.67
P(Married = No/B) = 2/3 =0.67
P(Gender = Female/B) = 0/3 = 0
P(Employed = Yes/B) = 1/3 = 0.33

P(Owns Home = Yes/C) = 2/4 =0.5
P(Married = No/C) = 0/4 =0
P(Gender = Female/C) = 4/4 = 1
P(Employed = Yes/C) = 4/4 = 1

Using the above probabilities, we obtain
P(X/A)= P(Owns Home = Yes/A) X P(Married = No/A) x P(Gender = Female/A) X P(Employed = Yes/A)

= 0.33 x 1 x 1 x 1 = 0.33

Similarly, P(X/B)= 0 , P(X/C) =0

To find the class, G, that maximizes,
P(X/Ci)P(Ci), we compute,

P(X/A) P(A) = 0.33 X 0.3 = 0.099
P(X/B) P(B) =0 X 0.3 = 0
P(X/C) P(C) = 0 X 0.4 = 0.0

Therefore x is assigned to class A

Advantages:

- Have the minimum error rate in comparison to all other classifiers.

- Easy to implement
- Good results obtained in most of the cases.
- They provide theoretical justification for other classifiers that do not explicitly use Bayes theorem.

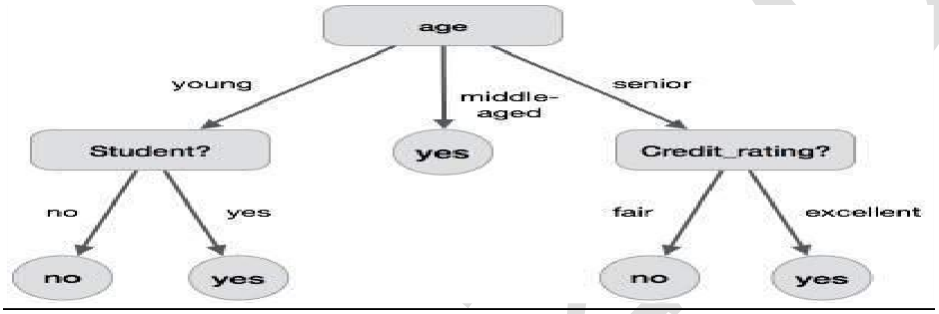
Disadvantages:

- Lack of available probability data.
- Inaccuracies in the assumption.

5.5.2 Decision Trees

A decision tree is a structure that includes a root node, branches, and leaf nodes. Each internal node denotes a test on an attribute, each branch denotes the outcome of a test, and each leaf node holds a class label. The topmost node in the tree is the root node.

The following decision tree is for the concept buy_computer that indicates whether a customer at a company is likely to buy a computer or not. Each internal node represents a test on an attribute. Each leaf node represents a class.



The benefits of having a decision tree are as follows −

- It does not require any domain knowledge.
- It is easy to comprehend.
- The learning and classification steps of a decision tree are simple and fast.

Decision Tree Induction Algorithm

- A machine researcher named J. Ross Quinlan in 1980 developed a decision tree algorithm known as ID3 (Iterative Dichotomiser). Later, he presented C4.5, which was the successor of ID3. ID3 and C4.5 adopt a greedy approach. In this algorithm, there is no backtracking; the trees are constructed in a top-down recursive divide-and-conquer manner.

Generating a decision tree form training tuples of data partition D

Algorithm : Generate_decision_tree Input:

- Data partition, D, which is a set of training tuples and their associated class labels.
- attribute_list, the set of candidate attributes.
- Attribute selection method, a procedure to determine the splitting criterion that best partitions that the data tuples into individual classes. This criterion includes a splitting_attribute and either a splitting point or splitting subset.

Output: A Decision Tree

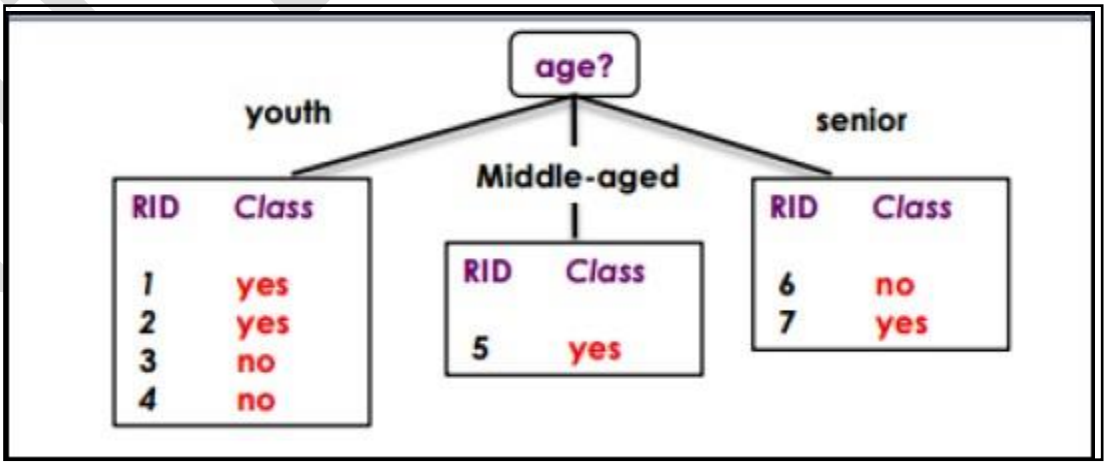
Method

1. create a node N;
2. if tuples in D are all of the same class, C then
3. return N as leaf node labeled with class C;
4. if attribute_list is empty then
5. return N as leaf node with labeled with majority class in D;//majority voting
6. apply attribute_selection_method(D, attribute_list) to find the best splitting_criterion;
7. label node N with splitting_criterion;
8. if splitting_attribute is discrete-valued and multiway splits allowed then // not restricted to binary trees
9. attribute_list = attribute_list - splitting attribute; // remove splitting attribute
10. for each outcome j of splitting criterion
// partition the tuples and grow subtrees for each partition
11. let Dj be the set of data tuples in D satisfying outcome j; // a partition
12. if Dj is empty then
13. attach a leaf labeled with the majority class in D to node N;
14. else attach the node returned by Generate_decision tree(Dj, attribute list) to node N;
end for
15. return N;

Example 1: customer likely to purchase a computer

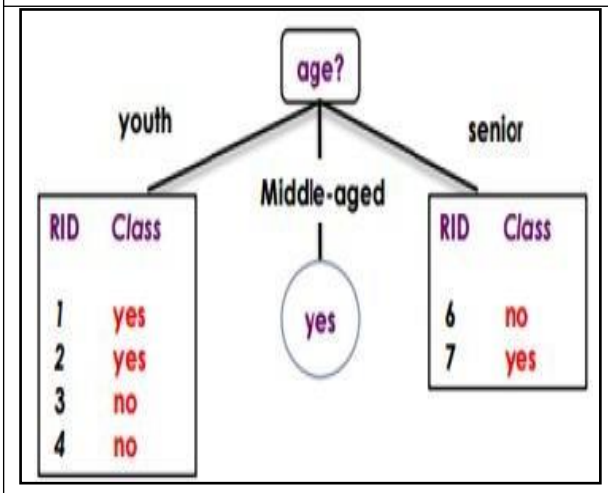
From the training dataset , calculate entropy value, which indicates that splitting attribute is: age (For calculation refer previous example in OHP Notes)

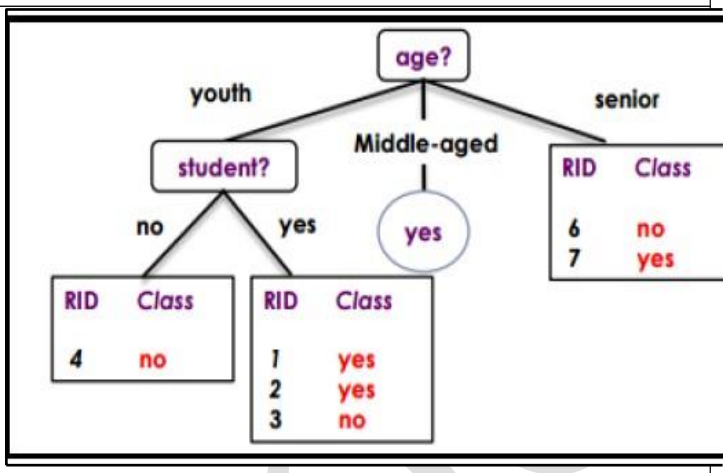
RID	age	student	credit-rating	Class: buys_computer
1	youth	yes	fair	yes
2	youth	yes	fair	yes
3	youth	yes	fair	no
4	youth	no	fair	no
5	middle-aged	no	excellent	yes
6	senior	yes	fair	no
7	senior	yes	excellent	yes



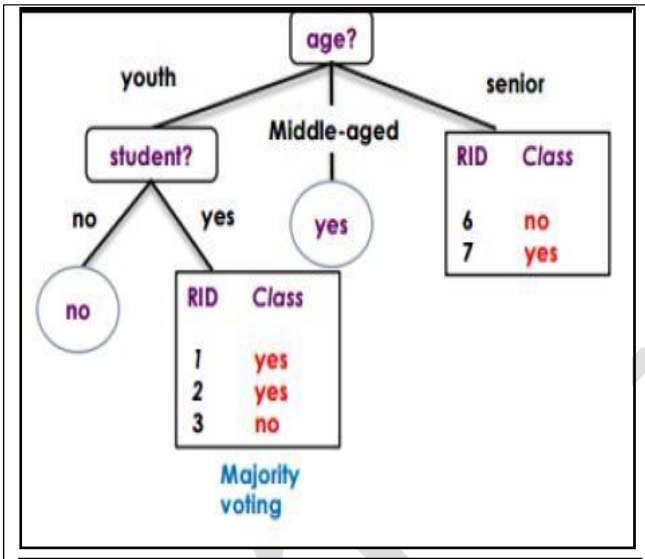
Age has 3 partitions :

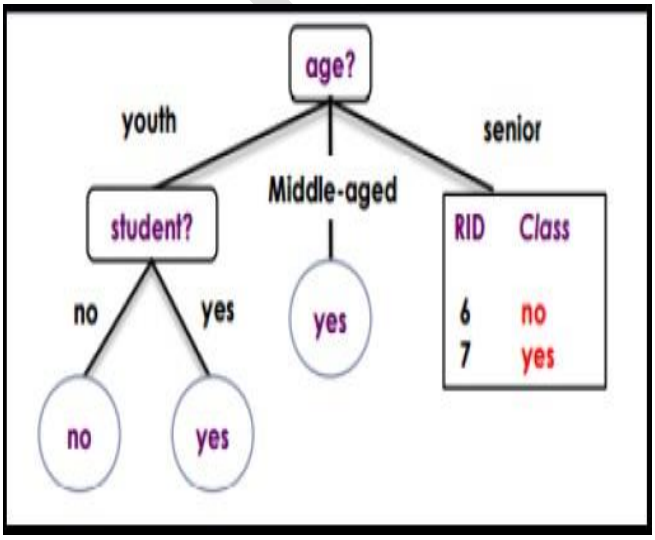
From the training data set , age= youth has 2 classes based on student attribute



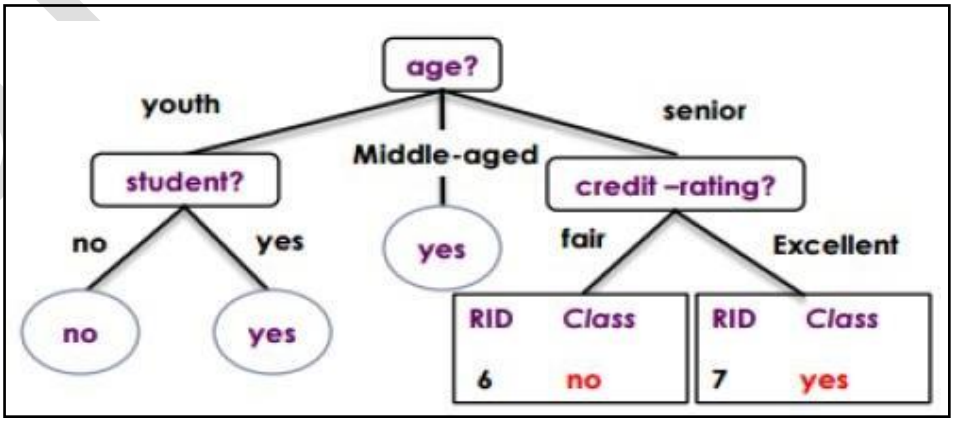


based on majority voting in student attribute , RID=3 is grouped under yes group.

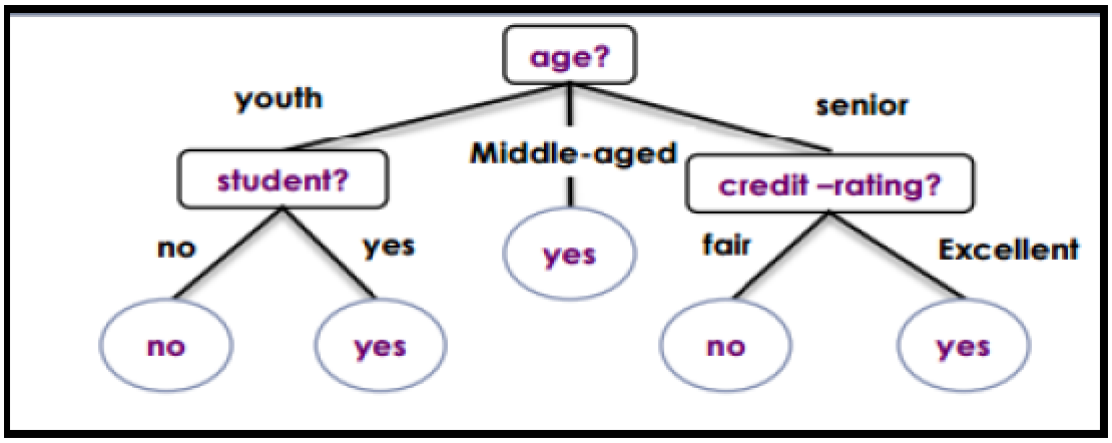




From the training data set , age= senior has 2 classes based on credit rating.

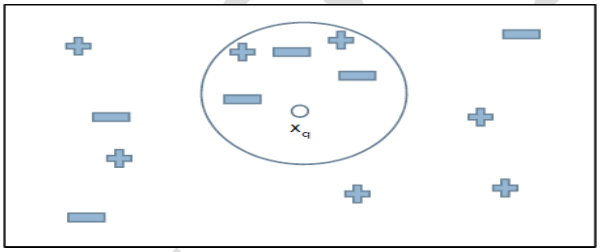


Final Decision Tree



5.5.3 K Nearest Neighbor Classification

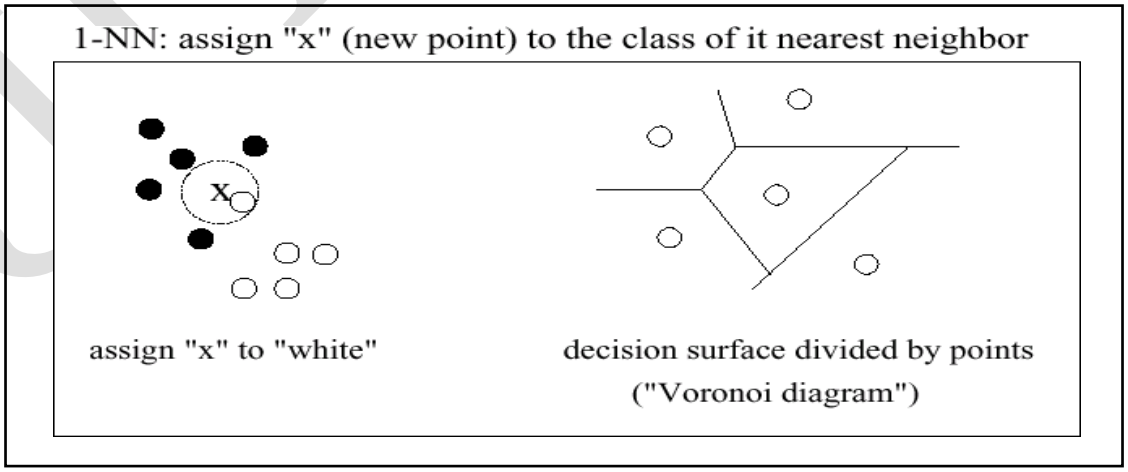
- K nearest neighbors is a simple algorithm that stores all available cases and classifies new cases based on a similarity measure (e.g., distance functions). K represents number of nearest neighbors.
- It classify an unknown example with the most common class among k closest examples
- KNN is based on "tell me who your neighbors are, and I"ll tell you who you are"
- Example:



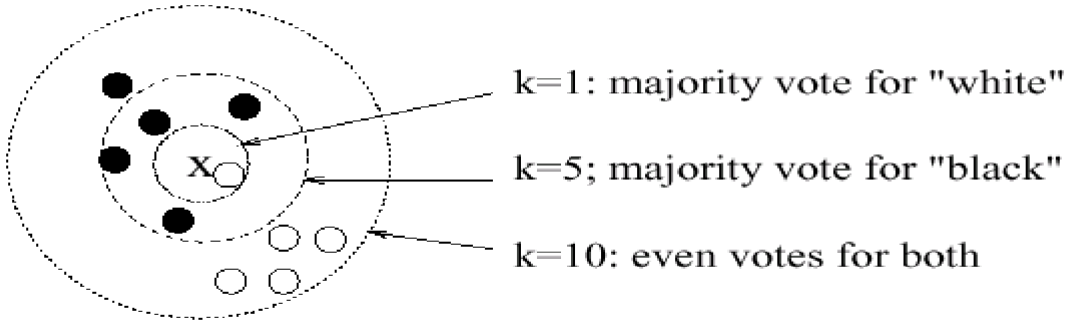
- If K = 5, then in this case query instance xq will be classified as negative since three of its nearest neighbors are classified as negative.

Different Schemes of KNN

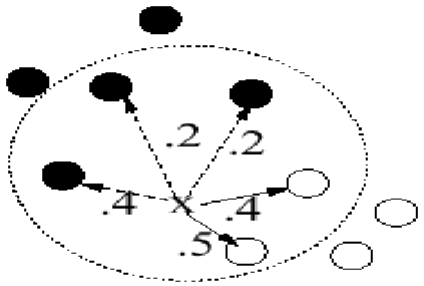
a. 1-Nearest Neighbor
b. K-Nearest Neighbor using a majority voting scheme
c. K-NN using a weighted-sum voting Scheme



K-Nearest Neighbor using a *majority* voting scheme



k-NN using a weighted-sum voting scheme



kNN (k = 5)

Assign "white" to x because the weighted sum of "whites" is larger then the sum of "blacks".

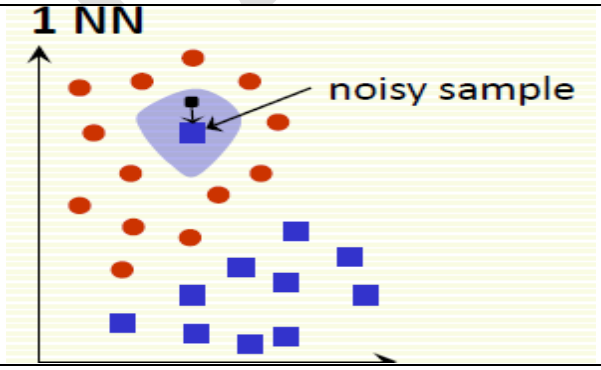
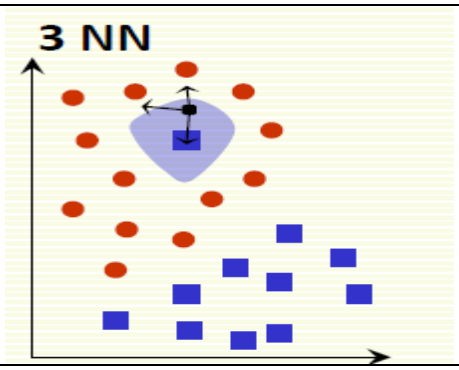
Each neighbor is given a weight according to its nearness.

kNN: How to Choose k?

In theory, if infinite number of samples available, the larger is ***k***, the better is classification
The limitation is that all ***k*** neighbors have to be close

- Possible when infinite no of samples available
- Impossible in practice since no of samples is finite k = 1 is

often used for efficiency, but sensitive to "noise"

	
every example in the blue shaded area will be misclassified as the blue class(rectangle)	every example in the blue shaded area will be classified correctly as the red class(circle)

Larger k gives smoother boundaries, better for generalization But only if locality is preserved. Locality is not preserved if end up looking at samples too far away, not from the same class.
•Interesting theoretical properties if k < sqrt(n), n is # of examples .

Find a heuristically optimal number k of nearest neighbors, based on RMSE(root-mean-square error). This is done using cross validation.

Cross-validation is another way to retrospectively determine a good K value by using an independent dataset to validate the K value. Historically, the optimal K for most datasets has been between 3-10. That produces much better results than 1NN.

Distance Measure in KNN

There are three distance measures are valid for continuous variables.

Distance functions

Euclidean $\sqrt{\sum_{i=1}^{k}(x_i - y_i)^2}$

Manhattan $\sum_{i=1}^{k}|x_i - y_i|$

Minkowski $\left(\sum_{i=1}^{k}(|x_i - y_i|)^q\right)^{1/q}$

It should also be noted that all In the instance of categorical variables the Hamming distance must be used. It also brings up the issue of standardization of the numerical variables between 0 and 1 when there is a mixture of numerical and categorical variables in the dataset.

Hamming Distance

$$D_H = \sum_{i=1}^{k}|x_i - y_i|$$

$$x = y \Rightarrow D = 0$$
$$x \neq y \Rightarrow D = 1$$

X	Y	Distance
Male	Male	0
Male	Female	1

Simple KNN - Algorithm:

For each training example , add the example to the list of training_examples.

Given a query instance xq to be classified,

- Let x1 ,x2….xk denote the k instances from training_examples that are nearest to xq .
- Return the class that represents the maximum of the k instances

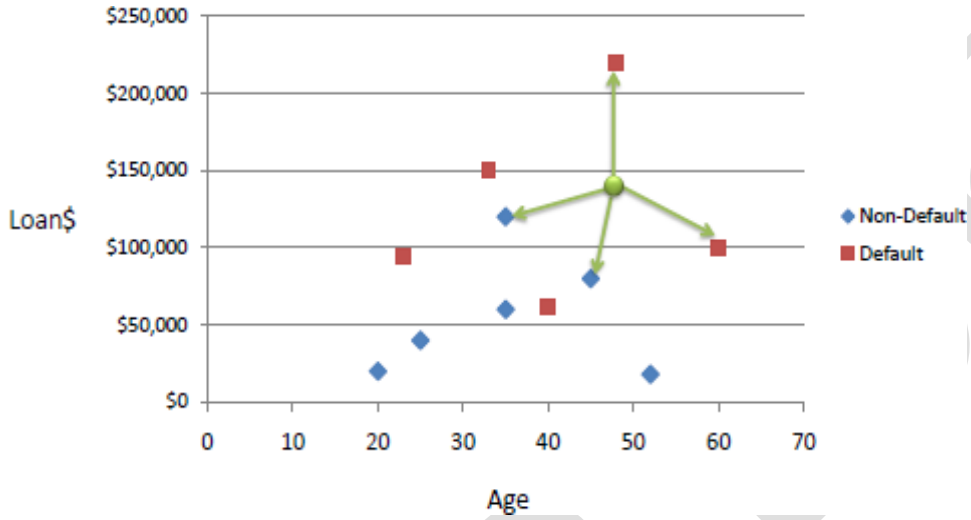
Steps:

1. Determine parameter k= no of nearest neighbor
2. Calculate the distance between the query instance and all the training samples

3. Sort the distance and determine nearest neighbor based on the k –th minimum distance
4. Gather the category of the nearest neighbors
5. Use simple majority of the category of nearest neighbors as the prediction value of the query instance.

Example:

Consider the following data concerning credit default. Age and Loan are two numerical variables (predictors) and Default is the target.



Given Training Data set :

Age	Loan	Default
25	$40,000	N
35	$60,000	N
45	$80,000	N
20	$20,000	N
35	$120,000	N
52	$18,000	N
23	$95,000	Y
40	$62,000	Y
60	$100,000	Y
48	$220,000	Y
33	$150,000	Y

Data to Classify:

to classify an unknown case (Age=48 and Loan=$142,000) using Euclidean distance.

Step1: Determine parameter k

K=3

Step 2: Calculate the distance

D = Sqrt[(48-33)^2 + (142000-150000)^2] = 8000.01 >> Default=Y

Age	Loan	Default	Distance	
25	$40,000	N	102000	
35	$60,000	N	82000	
45	$80,000	N	62000	
20	$20,000	N	122000	
35	$120,000	N	22000	2
52	$18,000	N	124000	
23	$95,000	Y	47000	
40	$62,000	Y	80000	
60	$100,000	Y	42000	3
48	$220,000	Y	78000	
33	$150,000	Y	8000	1
48	**$142,000**	**?**		

Euclidean Distance

$$D = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2}$$

Step 3: Sort the distance (refer above diagram) and mark upto kth rank i.e 1 to 3.

Step 4: Gather the category of the nearest neighbors

Age	Loan	Default	Distance
33	$150000	Y	8000
35	$120000	N	22000
60	$100000	Y	42000

With K=3, there are two Default=Y and one Default=N out of three closest neighbors. The prediction for the unknown case is Default=Y.

Standardized Distance (Feature Normalization)

One major drawback in calculating distance measures directly from the training set is in the case where variables have different measurement scales or there is a mixture of numerical and categorical variables. For example, if one variable is based on annual income in dollars, and the other is based on age in years then income will have a much higher influence on the distance calculated. One solution is to standardize the training set as shown below.

For ex loan , X =$ 40000 ,

Age	Loan	Default	Distance
0.125	0.11	N	0.7652
0.375	0.21	N	0.5200
0.625	0.31	N	0.3160
0	0.01	N	0.9245
0.375	0.50	N	0.3428
0.8	0.00	N	0.6220
0.075	0.38	Y	0.6669
0.5	0.22	Y	0.4437
1	0.41	Y	0.3650
0.7	1.00	Y	0.3861
0.325	0.65	Y	0.3771
0.7	**0.61**	**?**	

Standardized Variable

$$X_s = \frac{X - Min}{Max - Min}$$

Xs = $\frac{40000 - 20000}{220000 - 20000}$ = 0.11

Same way , calculate the standardized values for age and loan attributes, then apply the KNN algorithm.

Advantages

- Can be applied to the data from any distribution
  - for example, data does not have to be separable with a linear boundary
- Very simple and intuitive
- Good classification if the number of samples is large enough

Disadvantages

- Choosing k may be tricky
- Test stage is computationally expensive
  - No training stage, all the work is done during the test stage
  - This is actually the opposite of what we want. Usually we can afford training step to take a long time, but we want fast test step
- Need large number of samples for accuracy

5.6 Clustering Algorithm

- (Document) clustering is the process of grouping a set of documents into clusters of similar documents. Documents within a cluster should be similar.
- Documents from different clusters should be dissimilar. Clustering is the most common form of unsupervised learning.
- Unsupervised = there are no labeled or annotated data.

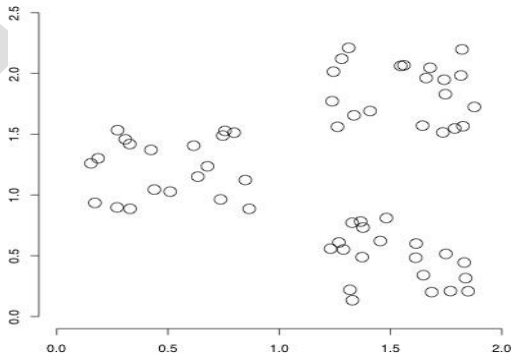
Most common form of *unsupervised learning (* Unsupervised learning = learning from raw data, as opposed to supervised data where a classification of examples is given)

General goal: put related docs in the same cluster, put unrelated docs in different clusters.

Secondary goals in clustering :

- Avoid very small and very large clusters
- Define clusters that are easy to explain to the user

Example: Data set with clear cluster structure

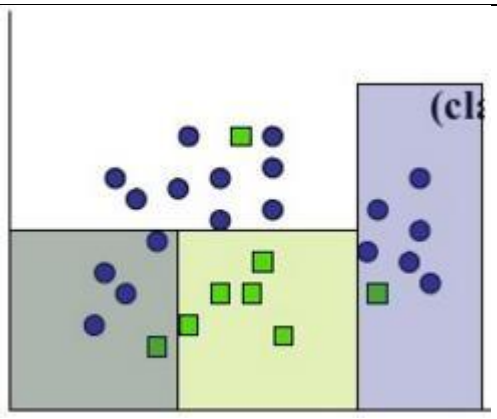
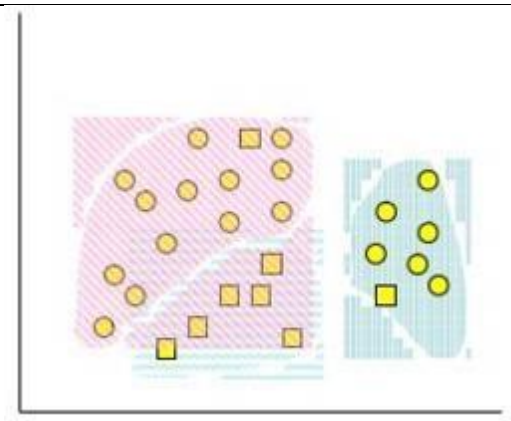


Why cluster documents?

- For improving recall in search applications
- For speeding up vector space retrieval
- Navigation

- Presentation of search results

Classification vs. Clustering

Classification	Clustering
supervised learning	unsupervised learning
Classes are human-defined and part of the input to the learning algorithm.	Clusters are inferred from the data without human input. However, there are many ways of influencing the outcome of clustering: number of clusters, similarity measure, representation of documents, . . .
 Learns a method for predicting the instance class from the prelabeled (classified) instances	 Find "natural" grouping of instances given un-labeled data

Applications of clustering in IR

Application	What is clustered?	Benefit
Search result clustering	search results	more effective information presentation to user
Scatter-Gather	(subsets of) collection	alternative user interface: "search without typing"
Collection clustering	collection	effective information presentation for exploratory browsing
Cluster-based retrieval	collection	higher efficiency: faster search

Clustering Algorithm Types

- Flat algorithms
  - Usually start with a random (partial) partitioning of docs into groups, Refine iteratively
  - Main algorithm: *K*-means
- Hierarchical algorithms
  - Create a hierarchy
  - Bottom-up, agglomerative

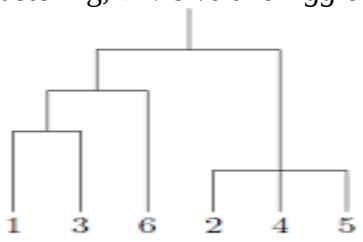
- Top-down, divisive
- Hard vs. Soft clustering
  - **Hard clustering:** Each document belongs to exactly one cluster.More common and easier to do
  - **Soft clustering:** A document can belong to more than one cluster. Ex: You may want to put sneakers in two clusters:
    - sports apparel
    - shoes

5.6.1 Agglomerative Clustering

Hierarchical Clustering

- Hierarchical clustering involves creating clusters that have a predetermined ordering from top to bottom. For example, all files and folders on the hard disk are organized in a hierarchy.
- Build a tree based hierarchical taxonomy from a set of document is called dendrogram.

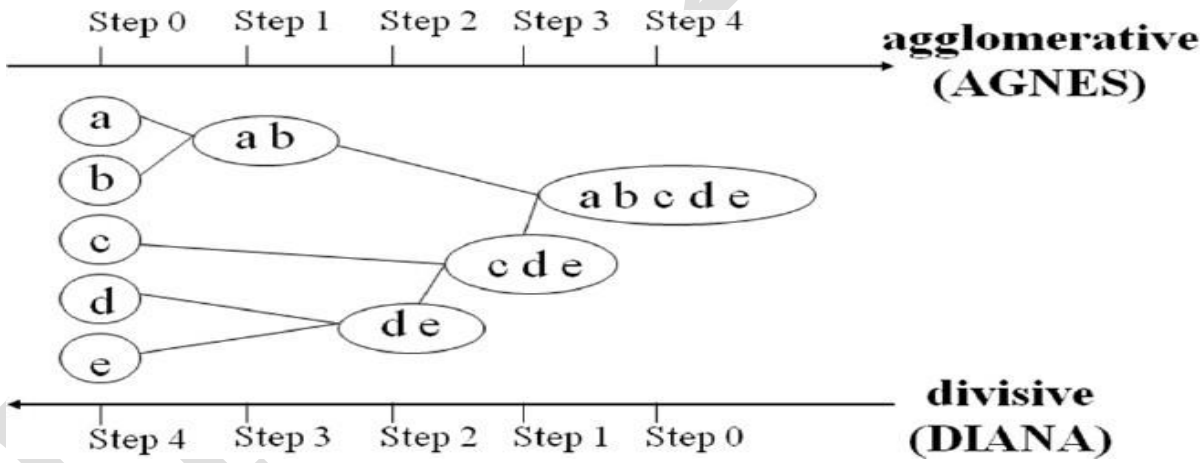
There are two types of hierarchical clustering, *Divisive* and *Agglomerative*.



An example dendrogram

Hierarchical

- Agglomerative
- Divisive



Agglomerative Clustering :

- Start with each document being a single cluster
- Eventually all doc belong to same cluster
- Don"t require no of clusters „k" in advance
- Need a terminating condition

Steps:

1. Start with each doc in a separate cluster
2. Compute the similarity between each of the cluster ,Repeatedly join closest pair of cluster until there is only one cluster

Algorithm :

Given:

A set X of objects $\{x_1,...,x_n\}$

A distance function $dist(c_1,c_2)$

for $i = 1$ to n

$c_i = \{x_i\}$

end for

$C = \{c_1,...,c_n\}$

$l = n+1$

while C.size > 1 **do**

- (c_{min1},c_{min2}) = minimum $dist(c_i,c_j)$ for all c_i,c_j in C
- remove c_{min1} and c_{min2} from C
- add $\{c_{min1},c_{min2}\}$ to C
- $l = l + 1$

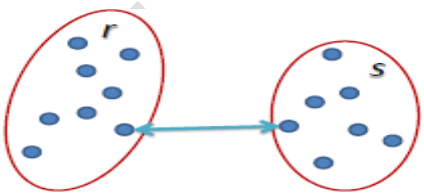
end while

Before any clustering is performed, it is required to determine the proximity matrix containing the distance between each point using a distance function. Then, the matrix is updated to display the distance between each cluster. The following three methods are used to find closest pair of clusters.

Methods to find closest pair of clusters:

Single Linkage

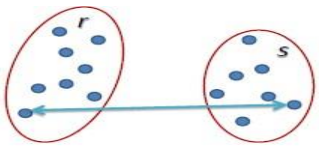
In single linkage hierarchical clustering, the distance between two clusters is defined as the *shortest* distance between two points in each cluster. For example, the distance between clusters "r" and "s" to the left is equal to the length of the arrow between their two closest points.



$$L(r,s) = \min(D(x_{ri}, x_{sj}))$$

Complete Linkage

In complete linkage hierarchical clustering, the distance between two clusters is defined as the *longest* distance between two points in each cluster. For example, the distance between clusters "r" and "s" to the left is equal to the length of the arrow between their two furthest points.

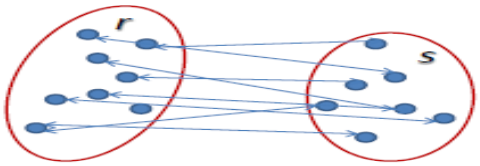


$$L(r,s) = \max(D(x_{ri}, x_{sj}))$$

Average linkage

In average linkage hierarchical clustering, the distance between two clusters is defined as the average distance between each point in one cluster to every point in the other

cluster. For example, the distance between clusters "r" and "s" to the left is equal to the average length each arrow between connecting the points of one cluster to the other.



$$L(r,s) = \frac{1}{n_r n_s} \sum_{i=1}^{n_r} \sum_{j=1}^{n_s} D(x_{ri}, x_{sj})$$

Example :

Let's now see a simple example: a hierarchical clustering of distances in kilometers between some Italian cities. The method used is single-linkage.

Input distance matrix (L = 0 for all the clusters):

	BA	FI	MI	NA	RM	TO
BA	0	662	877	255	412	996
FI	662	0	295	468	268	400
MI	877	295	0	754	564	138
NA	255	468	754	0	219	869
RM	412	268	564	219	0	669
TO	996	400	138	869	669	0

The nearest pair of cities is MI and TO, at distance 138. These are merged into a single cluster called "MI/TO". The level of the new cluster is L(MI/TO) = 138 and the new sequence number is m = 1.

Then we compute the distance from this new compound object to all other objects. In single link clustering the rule is that the distance from the compound object to another object is equal to the shortest distance from any member of the cluster to the outside object. So the distance from "MI/TO" to RM is chosen to be 564, which is the distance from MI to RM, and so on.

After merging MI with TO we obtain:

Dendrogram:

	BA	FI	MI/TO	NA	RM
BA	0	662	877	255	412
FI	662	0	295	468	268
MI/TO	877	295	0	754	564
NA	255	468	754	0	219
RM	412	268	564	219	0

min d(i,j) = d(NA,RM) = 219 => merge NA and RM into a new cluster called NA/RM L(NA/RM) = 219

m = 2

After merging NA and RM we obtain:

Dendrogram:

	BA	FI	MI/TO	NA/RM
BA	0	662	877	255
FI	662	0	295	268
MI/TO	877	295	0	564
NA/RM	255	268	564	0

min d(i,j) = d(BA,NA/RM) = 255 => merge BA and NA/RM into a new cluster called BA/NA/RM
L(BA/NA/RM) = 255
m = 3

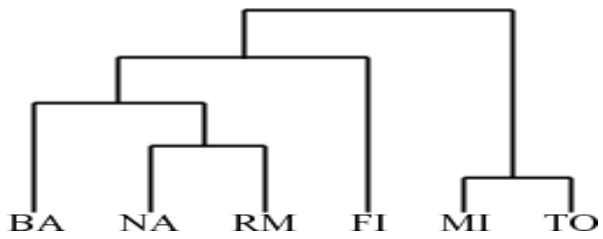
After merging BA and NA/RM:
Dendrogram:

	BA/NA/RM	FI	MI/TO
BA/NA/RM	0	268	564
FI	268	0	295
MI/TO	564	295	0

min d(i,j) = d(BA/NA/RM,FI) = 268 => merge BA/NA/RM and FI into a new cluster called BA/FI/NA/RM
L(BA/FI/NA/RM) = 268
m = 4
After merging BA/NA/RM and FI
Dendrogram:

	BA/FI/NA/RM	MI/TO
BA/FI/NA/RM	0	295
MI/TO	295	0

Finally, we merge the last two clusters at level 295.
The process is summarized by the following dendrogram.



Disadvantages

- they do not scale well: time complexity of at least $O(n^2)$, where n is the number of total objects;
- they can never undo what was done previously.

5.6.2 K – Means Clustering

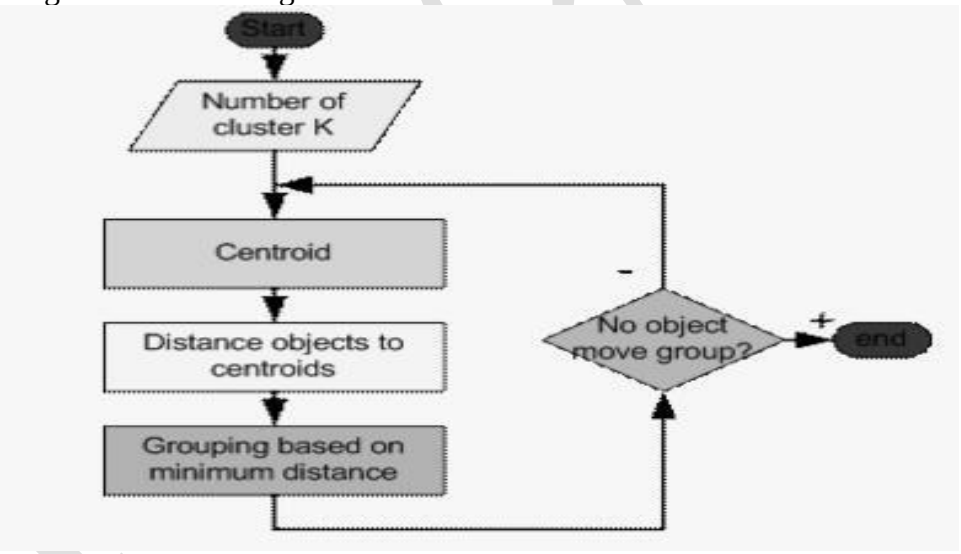
Document representations in clustering

- Vector space model
- As in vector space classification, we measure relatedness between vectors by Euclidean distance .. .which is almost equivalent to cosine similarity.
- Each cluster in *K*-means is defined by a centroid.
- Objective/partitioning criterion: minimize the average squared difference from the centroid
- Recall definition of centroid:

$$\vec{\mu}(\omega) = \frac{1}{|\omega|} \sum_{\vec{x} \in \omega} \vec{x}$$

where we use ω to denote a cluster.

- We try to find the minimum average squared difference by iterating two steps:
  - **reassignment:** assign each vector to its closest centroid
  - **recomputation:** recompute each centroid as the average of the vectors that were assigned to it in reassignment



- K-means can start with selecting as initial clusters centers K randomly chosen objects, namely the seeds. It then moves the cluster centers around in space in order to minimize RSS(A measure of how well the centroids represent the members of their clusters is the Residual Sum of Squares , the squared distance of each vector from its centroid summed over all vectors This is done iteratively by repeating two steps (reassignment , re computation) until a stopping criterion is met.

- We can use one of the following stopping conditions as stopping criterion A fixed number of iterations I has been completed.
  Centroids μi do not change between iterations.
  Terminate when RSS falls below a pre-estabilished threshold.

Algorithm

Input :

K: no of clusters

D: data set containing n objects

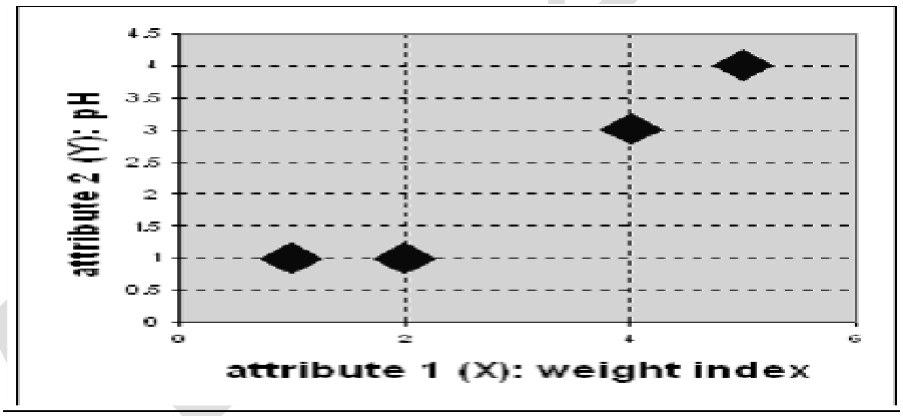
Output : a set of K clusters Steps:

1. Arbitrarily choose k objects from D as the initial cluster centers
2. Repeat
3. Reassign each object to the cluster to which the object is the most similar based on the distance measure
4. Recompute the centroid for newly formed cluster
5. Until no change

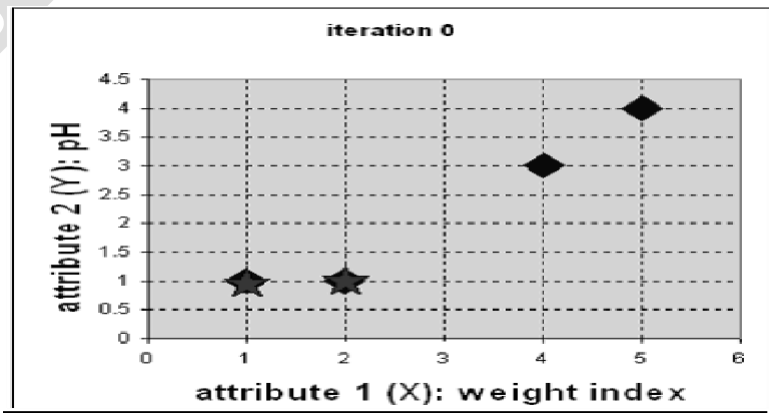
Example :

_Suppose we have several objects (4 types of medicines) and each object have 2 attributes or features as shown below. Our goal is to group these objects into k=2 group of medicine based on the features (pH and Weight Index)

Object	Attribute 1 (X) Weight Index	Attribute 2(Y) pH
Medicine A	1	1
Medicine B	2	1
Medicine C	4	3
Medicine D	5	4



Iteration 0:



K= 2

Initially , Centroid c1 : A(1, 1) ,Centroid c2 : B(2,1) <u>Reassign :</u>

Calculate Distance matrix

$$\begin{matrix} A & B & C & D & \\ \begin{bmatrix} 1 & 2 & 4 & 5 \\ 1 & 1 & 3 & 4 \end{bmatrix} & & & & \begin{matrix} X \\ Y \end{matrix} \end{matrix}$$

Distance from c (4,3) to c1 (1,1) is

$$\sqrt{(4-1)^2+(3-1)^2} = 3.61$$

Same way calculate for all

$$\mathbf{D}^0 = \begin{bmatrix} 0 & 1 & 3.61 & 5 \\ 1 & 0 & 2.83 & 4.24 \end{bmatrix} \quad \begin{matrix} \mathbf{c}_1 = (1,1) & group-1 \\ \mathbf{c}_2 = (2,1) & group-2 \end{matrix}$$

Take the minimum distance (in the distance matrix D^0, take each column , put 1 for min value in the group matrix)

$$\mathbf{G}^0 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 \end{bmatrix} \quad \begin{matrix} group-1 \\ group-2 \end{matrix}$$

Recompute:

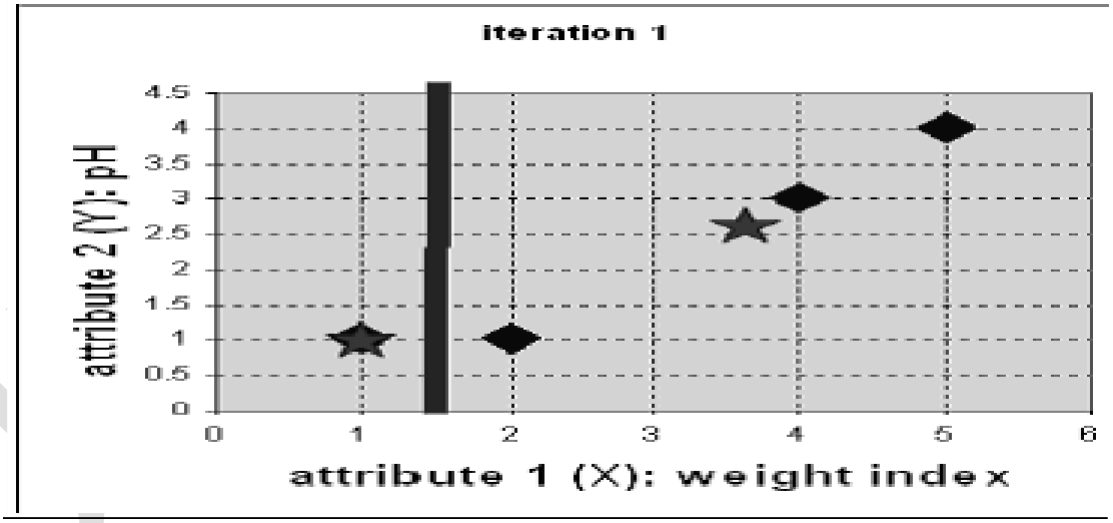
Calculate new c1, new c2 New

c1 = {A}

New c2= centroid of {B,C,D)

$$C2 = (\frac{2+4+5}{3}, \frac{1+3+4}{3}) = (\frac{11}{3}, \frac{8}{3})$$

Iteration 1:



c1(1,1) , c2(11/3, 8/3)

Reassign

Calculate Distance matrix

$$\begin{matrix} A & B & C & D & \\ \begin{bmatrix} 1 & 2 & 4 & 5 \\ 1 & 1 & 3 & 4 \end{bmatrix} & & & & \begin{matrix} X \\ Y \end{matrix} \end{matrix}$$

$$D^1 = \begin{bmatrix} 0 & 1 & 3.61 & 5 \\ 3.14 & 2.36 & 0.47 & 1.89 \end{bmatrix} \quad \begin{matrix} c_1 = (1,1) & group-1 \\ c_2 = (\frac{11}{3}, \frac{8}{3}) & group-2 \end{matrix}$$

Take the minimum distance

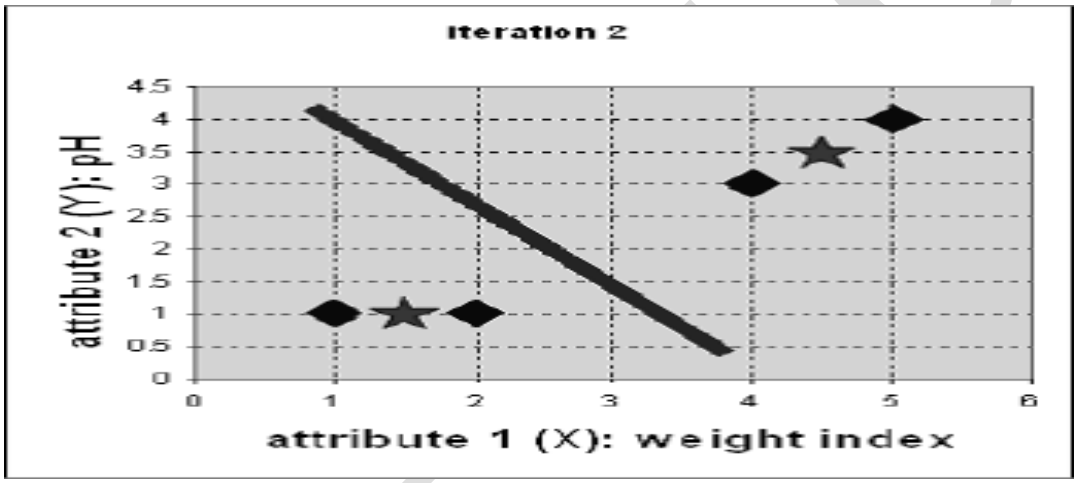
$$G^1 = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix} \quad \begin{matrix} group-1 \\ group-2 \end{matrix}$$
$$A \quad B \quad C \quad D$$

Recompute

$$c_1 = (\frac{1+2}{2}, \frac{1+1}{2}) = (1\tfrac{1}{2}, 1)$$

$$c_2 = (\frac{4+5}{2}, \frac{3+4}{2}) = (4\tfrac{1}{2}, 3\tfrac{1}{2})$$

Iteration 2:



Reassign

$$D^2 = \begin{bmatrix} 0.5 & 0.5 & 3.20 & 4.61 \\ 4.30 & 3.54 & 0.71 & 0.71 \end{bmatrix} \quad \begin{matrix} c_1 = (1\frac{1}{2}, 1) & group-1 \\ c_2 = (4\frac{1}{2}, 3\frac{1}{2}) & group-2 \end{matrix}$$

$$G^2 = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix} \quad \begin{matrix} group-1 \\ group-2 \end{matrix}$$
$$A \quad B \quad C \quad D$$

We obtain result that $G^2 = G^1$. Comparing the groups of last iteration , this iteration does not move group any more. Final Result of k – means clustering is

Object	Attribute 1 (X) Weight Index	Attribute 2(Y) pH	Group (result)
Medicine A	1	1	1
Medicine B	2	1	1
Medicine C	4	3	2
Medicine D	5	4	2

5.6.3 Expectation Maximization (EM).

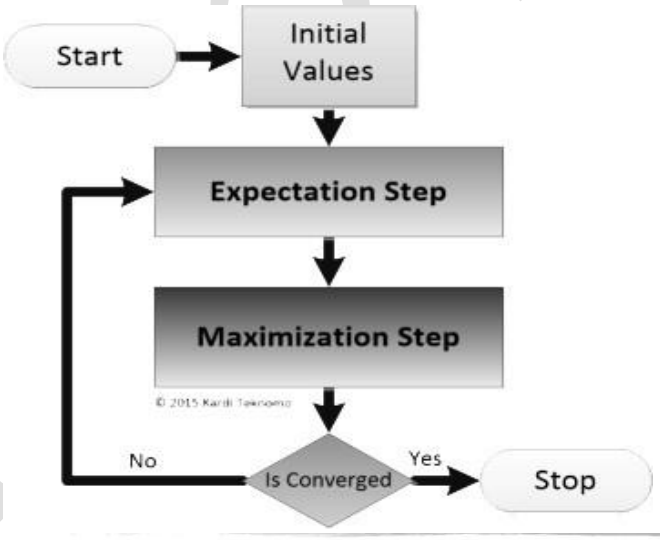
- **EM algorithm** is an iterative estimation algorithm that can derive the maximum likelihood (ML) estimates in the presence of missing/hidden data ("incomplete data")
- It is a type of soft clustering which gives probabilities that an instance belongs to each cluster.
- It is a model based clustering.
  - Assumes that data were generated by a model & tries to recover orginal model from data
  - The Recovered data defines cluster & assignment of doc to clusters
  - To estimate the model parameter use maximum likelihood estimation
- It is a soft version of k means clustering.Direct method that assumes k clusters .
- Consider a set of starting parameters , Use these to "estimate" the missing data , Use "complete" data to update parameters , Repeat until convergence.General algorithm for missing data problems, First introduced in 1977
- Goal : assume the set of data come from a underlying distribution, we need to guess the most likely (maximum likelihood) parameters of that model.

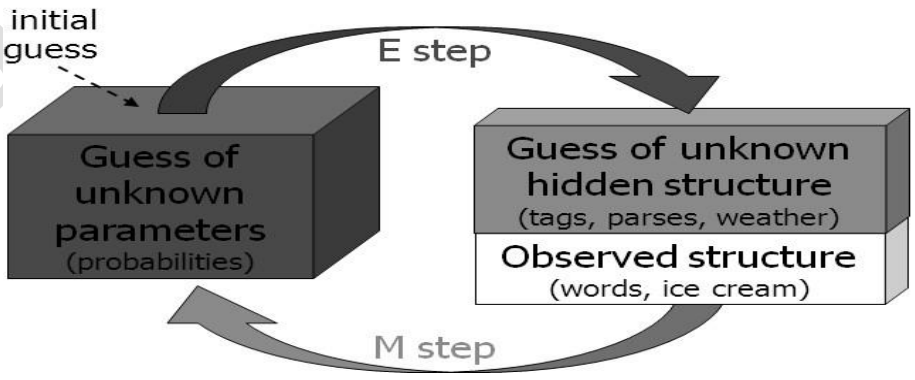
EM algorithm:

2 steps :

1) Expectation (similar to Reassignment of K-means)
2) Maximization (similar to Recomputation of K-means)

Parameters of EM (similar to centroid in K –means)





- The basic functioning of the EM algorithm can be divided into two steps (the parameter to be estimated is θ):
  - Expectation step (E-step)
    - Take the expected value of the complete data given the observation and the current parameter estimate $\hat{\theta}_k$

$$Q(\theta,\hat{\theta}_k) = E\left\{\log f(\mathbf{x}\,|\,\theta)\,|\,\mathbf{y},\hat{\theta}_k\right\}$$

  - Maximization step (M-step)
    - Maximize the Q-function in the E-step (basically, the data of the E-step is used as it were measured observations)

$$\hat{\theta}_{k+1} = \arg\max_{\theta} Q(\theta\,|\,\hat{\theta}_k)$$

- The likelihood of the parameter is increased at every iteration
  - EM converges towards some local maximum of the likelihood function

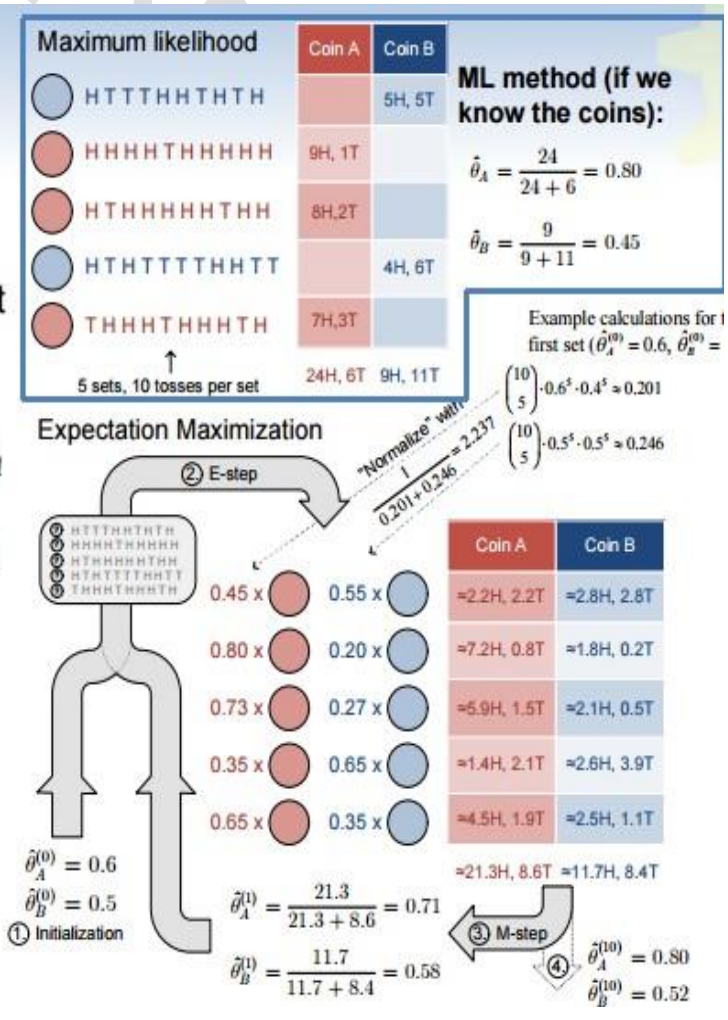
Example 1: Coin Problem

- We have two coins: A and B
- The probabilities for heads are θ_A and θ_B
- We have 5 measurement sets including 10 coin tosses in each set
- If we know which of the coins are tossed in each set, we can calculate the ML probabilities for θ_A and θ_B
- If we don't know which of the coins are tossed in each set, ML estimates cannot be calculated directly

  → EM algorithm

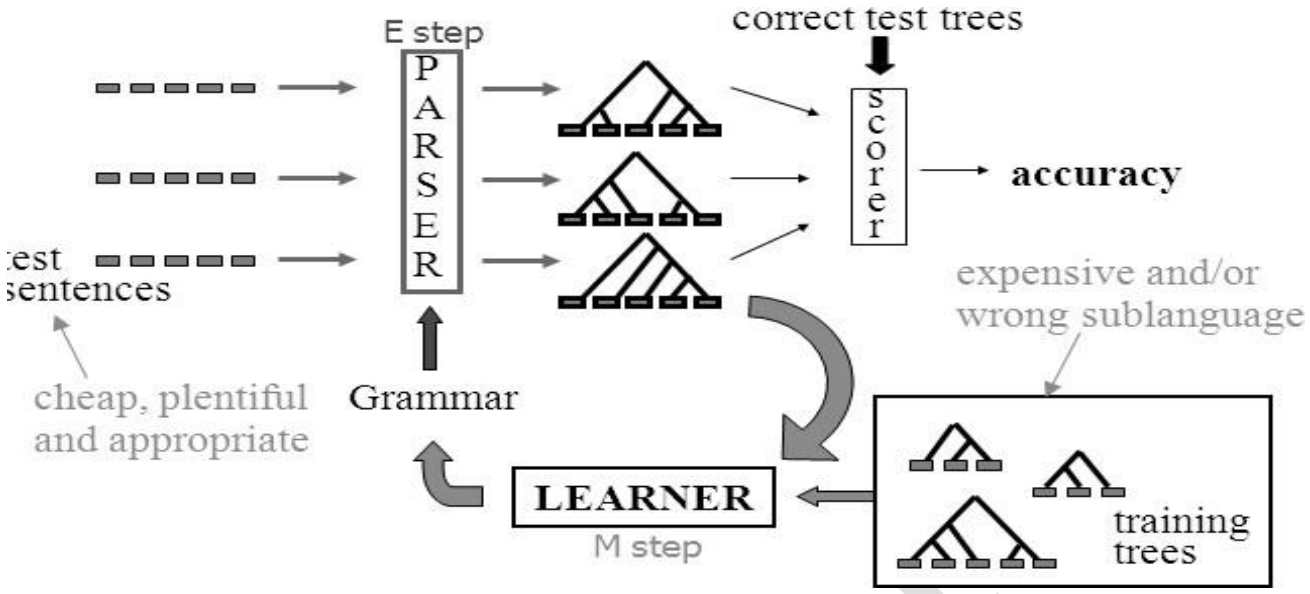
Binomial distribution used to calculate probabilities:

$$\binom{n}{k} p^k (1-p)_H^{n-k}$$



Example 2 :

Test sentences are taken as i/p to parser (E Step) which construct parse tree without the grammer initially , from the parse tree learner (M step) generate the grammer using training trees , this process is repeated till scorer gets the accuracy.



Strengths of EM

- Numerical stability: in every iteration of the EM algorithm, it increases the likelihood of the observed data.
- The EM handles parameter constraints gracefully.

Problems with EM

Convergence can be very slow on some problems

It guarantees to improve the probability of the training corpus