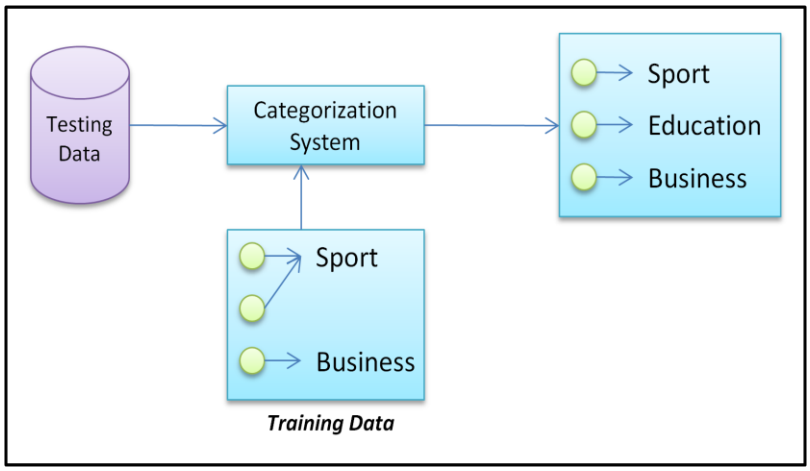
TEXT CATEGORIZATION

- Text categorization is a task of automatically sorting a set of documents into categories (classes) from predefined set.
- Classify new document.
- Supervised learning.



APPLICATIONS

1. News article classification
2. Automatic email filtering
3. Web page classification
4. Word sense disambiguity

CATEGORIZATION ALGORITHM

1. Manually –Rule based
2. Automatic (Learning Algorithm)
   a. Rochhi algorithm
   b. Baye's Theorem
   c. Decision Trees
   d. KNN
   e. SVM (Support Vector Machine)

Given,

- A description of an instance x Є X, where X is the instance language or instance space.
- A fixed set of categories C= $\{c_1, c_2, \ldots\ldots, c_n\}$

Determine: The category of x: c(x) Є C where, c(x) is a categorization function whose domain is X and whose range is C.

ROCCHIO ALGORITHM

- Using relevance feedback, i.e. relevance feedback methods can be adopted for text categorization.
- Use TF/IDF weights vectors to represent text document.
- For each category compute a prototype vector by summing the vectors of the training documents in the category.
- Assign test documents to the category with the closest prototype vector based on cosine similarity.

TRAINING ALGORITHM

- Assume the set of categories is $\{c_1, c_2, \ldots\ldots, c_n\}$
- For i = 1 to n
    - let Pi = <0, 0,...., 0> (initial prototype vectors)
- For each training example x, let d be the normalized TF/IDF term vector for document x.
- For all i, $P_i = P_i + d$

TESTING ALGORITHM

- Given test document x
- Let d be the TF/IDF weighted vector for x
- Let m = −2 (initial minimum cosine)
- For i = 1 to n
    - Let s = cossim(d, P_i) → compute similarity to each prototype
    - if (s>m)

      {

      m = s;

      r = c_i; → update with most closest class

      }
- loop
- return class r

Exercise

- Assume the following training set

  Food: "Turky stuffing"

  Food: "Buffalo wings"

  Beverage: "Cream soda"

  Beverage: "Orange soda"

  Apply the Rocchio algorithm to classify a new name "Turky Soda".

BAYESIAN METHODS

- Learning and classification methods based on probability theory.
- Baye's theorem plays a critical role in probabilistic learning and classification.
- Uses prior probability of each category given no information about an item.
- Categorization produces a posterior probability distribution over the possible categories given a description of an item.

BAYE'S THEOREM

- $P(A|B) = \frac{P(A)*P(B \mid A)}{P(B)}$
- Example:

D =

Size	Color	Shape	Category
Small	Red	Circle	Positive
Large	Red	Circle	Positive
Small	Red	Triangle	Negative
Large	blue	Circle	Negative

AFTER TRAINING

Probability	Positive	Negative
P(Y)	0.5	0.5
P(small|Y)	0.5	0.5
P(medium|Y)	0.0	0.0
P(large|Y)	0.5	0.5
P(red|Y)	1.0	0.5
P(blue|Y)	0.0	0.5
P(green|Y)	0.0	0.0
P(square|Y)	0.0	0.5
P(triangle|Y)	0.0	0.5
P(circle|Y)	1.0	0.5

Size<small, medium, large>

Color<red, blue, green>

Shape<circle, triangle, square>

Category<positive, negative>

- Testing Sample X: <medium, red, circle>

- P(pos | X) $= \frac{P(pos) * P(X|pos)}{P(X)}$

  = P(pos) * P(medium | pos) * P(red | pos) * P(circle | pos)

  = 0.5 * 0.001 * 1.0 * 1.0

  = 0.0005

- P(neg | X) $= \frac{P(neg) * P(X|neg)}{P(X)}$

  = P(neg) * P(medium | neg) * P(red | neg) * P(circle | neg)

  = 0.5 * 0.001 * 0.5 * 0.5

  = 0.000125

TRAINING ALGORITHM

- Let v be the vocabulary of all words in D
- For each category Ci Ɛ C
  - Let D_i be the subset of documents in category C_i
  - $P(C_i) = |D_i|/|D|$
  - Let T_i be the concatenation of all documents in d_i
  - Let n_i be the total number of word occurrences in T_i
  - For each word W_j Ɛ V
    - Let n_{ij} be the number of occurrences of W_j in T_i
    - Let $P(W_j|Ci) = (n_{ij} + 1)/(n_i+|V|)$

DECISION TREES

- Decision tree induction is the learning of decision trees from class labeled training tuples.
- A decision tree is a flowchart like tree structure where each internal node denotes a test on an attribute, each branch represents an outcome of the test and each node holds a class label.

- Example:

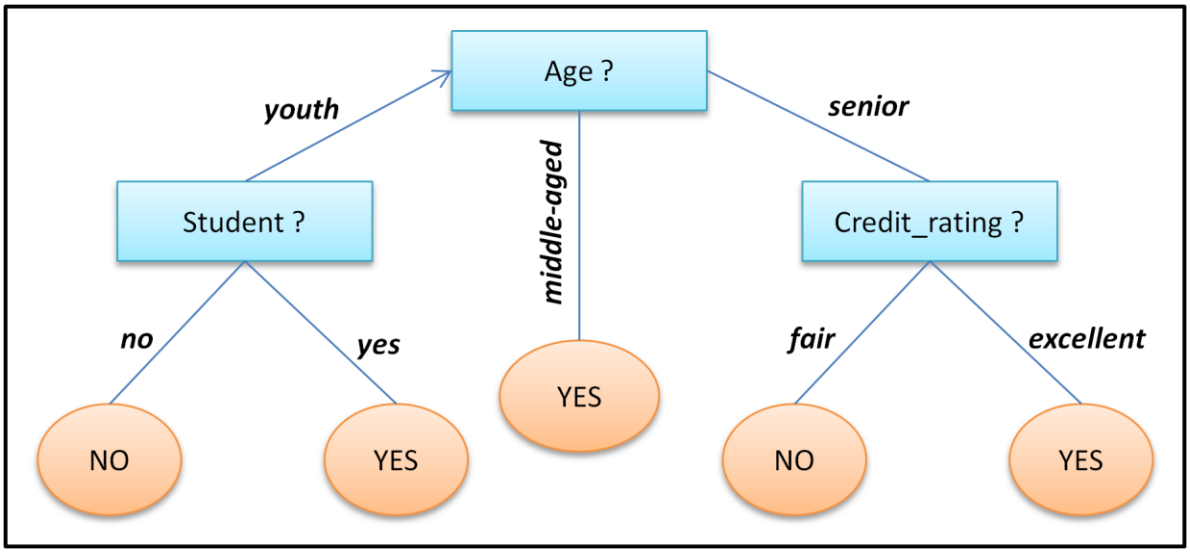


Fig. A decision tree for the concept buys_computer indicating whether a customer is likely to purchase a computer

- Example:

Consider a data with a number of examples for several days with a class "Play Tennis".

Day	Outlook	Temperature	Humidity	Wind	Play Tennis
D1	SUNNY	HOT	HIGH	WEAK	NO
D2	SUNNY	HOT	HIGH	STRONG	NO
D3	OVERCAST	HOT	HIGH	WEAK	YES
D4	RAIN	MILD	HIGH	WEAK	YES
D5	RAIN	COOL	NORMAL	WEAK	YES
D6	RAIN	COOL	NORMAL	STRONG	NO
D7	OVERCAST	COOL	NORMAL	STRONG	YES
D8	SUNNY	MILD	HIGH	WEAK	NO
D9	SUNNY	COOL	NORMAL	WEAK	YES
D10	RAIN	MILD	NORMAL	WEAK	YES

D11	SUNNY	MILD	NORMAL	STRONG	YES
D12	OVERCAST	MILD	HIGH	STRONG	YES
D13	OVERCAST	HOT	NORMAL	WEAK	YES
D14	RAIN	MILD	HIGH	STRONG	NO

Outlook <sunny, overcast, rain>

Temperature <hot, mild, cool>

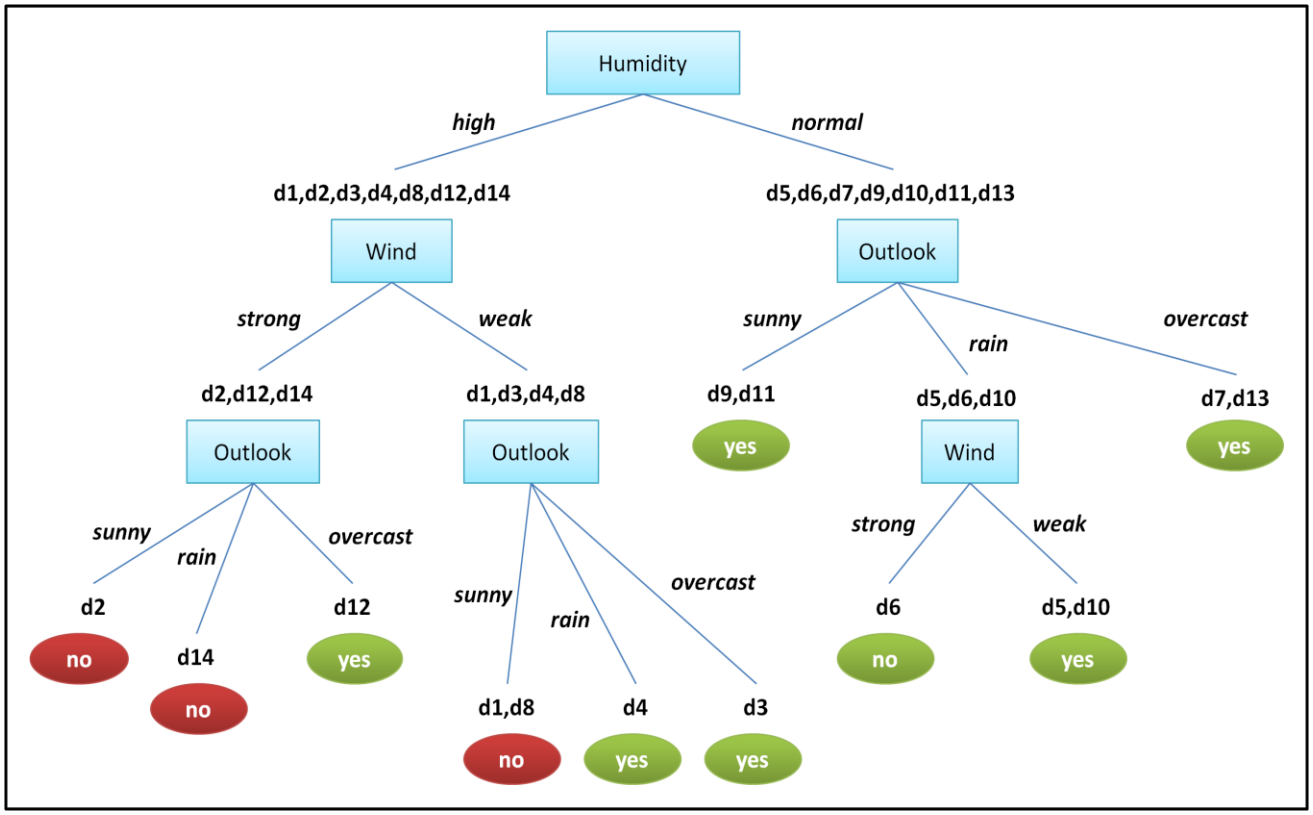
Humidity <high, normal>

Wind <weak, strong>

Play Tennis <no, yes>

Building Decision Tree

- We first need to decide which attribute to make a decision. Let's say we selected "humidity".



Testing : <sunny, hot, normal, weak> → YES

NEAREST NEIGHBOR LEARNING ALGORITHM

- Learning is just storing the representations of the training examples in D.
- Testing instance x.
  - Compute similarity between x and all examples in D.
  - Assign x the category of the most similar example in D.
- Find the K-most similar examples and return the majority category of these K-examples.
- Value of K is typically odd to avoid ties, 3 and 5 are most common.
- Nearest neighbor method depends on similarity (Euclidean distance in m-dimensional instance).
- For text, cosine similarity of TF-IDF weighted vectors is most effective.
- Training Algorithm
  - For each training example <x, c(x)> Ɛ D.

    Compute the corresponding TF-IDF vector dx for document x
- Testing Algorithm
  - For testing instance y
  - Compute TF-IDF vector d for document y
  - For each <x, C(x)>ƐD
    - Let Sx = cossim(d,dx)
  - Sort x, in D by decreasing value of Sx
  - Let N be the first K examples in D
  - Return the majority class of examples in N

Exercise

- Assume the following training set (2 classes)

  Food: “turkey stuffing”

  Food: “buffalo wings”

  Beverage: “cream soda”

  Beverage: “orange soda”

  Apply KNN with K=3 to classify new name “turkey soda”.

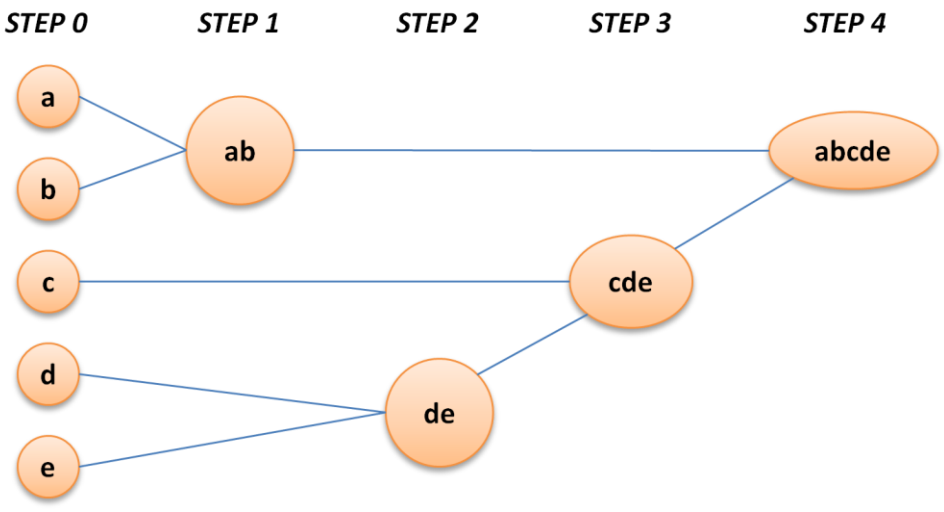
CLUSTERING ALGORITHM

- Clustering algorithm group a set of documents into subsets or clusters.
- To create clusters that is coherent internally but clearly different form each other, i.e. documents within a cluster should be as similar as possible and documents in one cluster should be as dissimilar as possible form documents in other clusters.
- Unsupervised learning i.e. no involvement of human expert who assigned documents into classes.
- Flat clustering creates a set of clusters without any explicit structure.
- Hierarchical clustering creates a hierarchy of clusters.
- Hard clustering assigns each document exactly in one cluster.
- Soft clustering distributes a document over all clusters.

EXPECTATION MAXIMIZATION -SELF STUDY

AGGLOMERATIVE CLUSTERING ALGORITHM

- The algorithm forms clusters in a bottom up manner as follows:
  - Initially put each article in its own cluster.
  - Among all current clusters, pick the two clusters with smallest distance
  - Replace these two clusters with a new cluster formed by merging the two original ones.
  - Repeat the above two steps until there is only one remaining cluster.
- Example:



K-MEANS ALGORITHM

- Each cluster is represented by the centre of the cluster
- Algorithm
  - Choose k number of clusters to be determined.
  - Choose k objects randomly as the initial cluster centers
  - Repeat
    - Assign each object to their closest cluster
    - Compute new clusters, calculate mean points
  - Until
    - No change in cluster entities OR
    - No objects change its clusters.
- Example: Consider the following instances in the table given (2D Form)

Instance	X	Y
1	1.0	1.5
2	1.0	4.5
3	2.0	1.5
4	2.0	3.5
5	3.0	2.5
6	3.0	4.0

1. If the objects are to be partitioned into 2 clusters then k = 2.
2. Next choose two points are random, object 1 and 3 are chosen,
   i.e. C_1 = (1.0, 1.5) and C_2 = (2.0, 1.5)
3. Euclidean distance between i's and j's:

$$D(i - j) = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$$

INFORMATION FILTERING SYSTEM

- Information filtering system is a system that removes redundant or unwanted information from an information stream using automated or computerized methods.
- A filtering system consists of several tools that help people find the most valuable information so in the limited time, you can dedicate to read/listen/view correctly directional and valuable documents.
- It reduces or eliminates the harmful information
- Application:
  (a) Spam filtering, (b) Censorship and (c) Entrance selection