For written notes on this lecture, please read chapter 3 of *The Practical Bioinformatician. Alternatively, please read* "Rule-Based Data Mining Methods for Classification Problems in Biomedical Domains", a tutorial at *PKDD04* by Jinyan Li and Limsoon Wong, September 2004. http://www.comp.nus.edu.sg/~wongls/talks/pkdd04/

Bioinformatics and Biomarker Discovery Part 2: Tools

Limsoon Wong

27 August 2009

Outline

- **Overview of Supervised Learning**
- **Decision Trees Ensembles**
  - Bagging
- **Other Methods**
  - K-Nearest Neighbour
  - Bayesian Approach

Overview of Supervised Learning

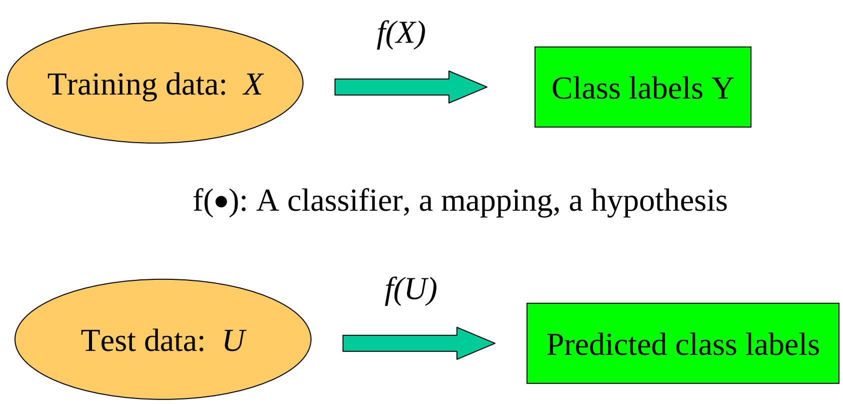
Computational Supervised Learning

- **Also called classification**
- **Learn from past experience, and use the learned knowledge to classify new data**
- **Knowledge learned by intelligent algorithms**
- **Examples:**
  - Clinical diagnosis for patients
  - Cell type classification

Data

- **Classification application involves > 1 class of data. E.g.,**
  - Normal vs disease cells for a diagnosis problem
- **Training data is a set of instances (samples, points) with known class labels**
- **Test data is a set of instances whose class labels are to be predicted**

Process



Relational Representation of Patient Data

n features (order of 1000)

	$gene_1$	$gene_2$	$gene_3$	$gene_4$	…	$gene_n$	class
m samples	x_{11}	x_{12}	x_{13}	x_{14}	…	x_{1n}	P
	x_{21}	x_{22}	x_{23}	x_{24}	…	x_{2n}	N
	x_{31}	x_{32}	x_{33}	x_{34}	…	x_{3n}	P
	………………………………………						
	x_{m1}	x_{m2}	x_{m3}	x_{m4}	…	x_{mn}	N

Requirements of Biomedical Classification

- **High accuracy/sensitivity/specificity/precision**
- **High comprehensibility**

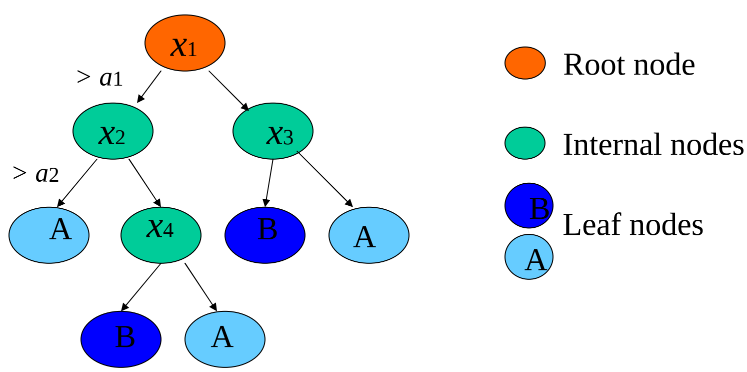
Importance of Rule-Based Methods

- **Systematic selection of a small number of features used for the decision making**

$\Rightarrow$ **Increase the comprehensibility of the knowledge patterns**

- **C4.5 and CART are two commonly used rule induction algorithms---a.k.a. decision tree induction algorithms**

Structure of Decision Trees



- **Every path from root to a leaf forms a decision rule**
  - If $x_1 > a_1$ & $x_2 > a_2$, then it's A class
- **C4.5, CART, two of the most widely used**
- **Easy interpretation, but accuracy generally unattractive**

A Simple Dataset

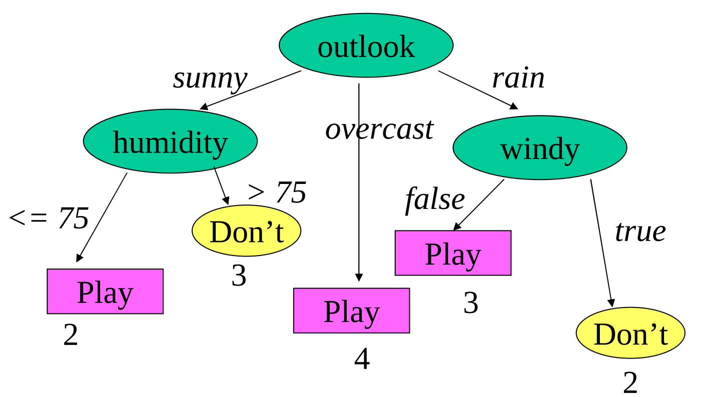
Outlook	Temp	Humidity	Windy	class
Sunny	75	70	true	Play
Sunny	80	90	true	Don't
Sunny	85	85	false	Don't
Sunny	72	95	true	Don't
Sunny	69	70	false	Play
Overcast	72	90	true	Play
Overcast	83	78	false	Play
Overcast	64	65	true	Play
Overcast	81	75	false	Play
Rain	71	80	true	Don't
Rain	65	70	true	Don't
Rain	75	80	false	Play
Rain	68	80	false	Play
Rain	70	96	false	Play

9 Play samples

5 Don't

A total of 14.

A Decision Tree

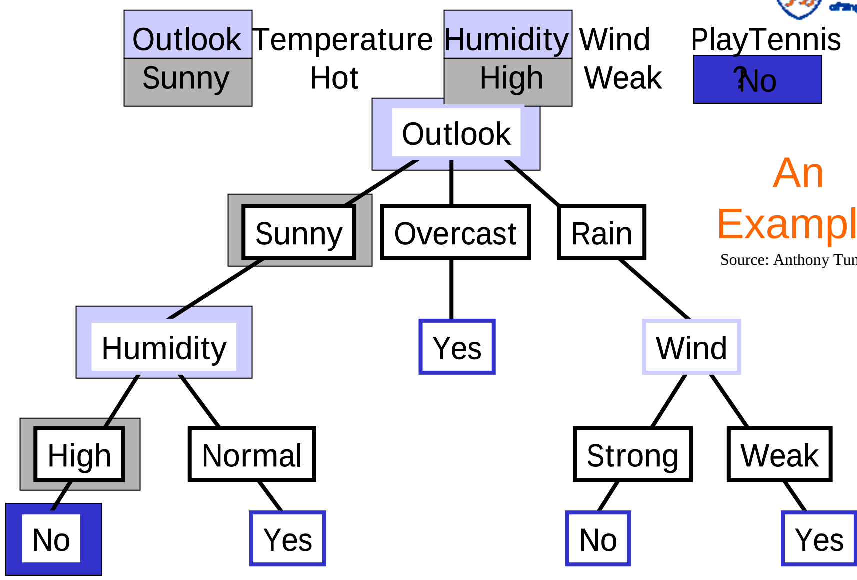


- **Construction of a tree is equivalent to determination of the root node of the tree and the root node of its sub-trees**

Exercise: What is the accuracy of this tree?

An Example

Source: Anthony Tung



Most Discriminatory Feature

- **Every feature can be used to partition the training data**
- **If the partitions contain a pure class of training instances, then this feature is most discriminatory**

Example of Partitions

- **Categorical feature**
  - Number of partitions of the training data is equal to the number of values of this feature
- **Numerical feature**
  - Two partitions

Categorical feature

Numerical feature

Instance #	Outlook	Temp	Humidity	Windy	class
1	Sunny	75	70	true	Play
2	Sunny	80	90	true	Don't
3	Sunny	85	85	false	Don't
4	Sunny	72	95	true	Don't
5	Sunny	69	70	false	Play
6	Overcast	72	90	true	Play
7	Overcast	83	78	false	Play
8	Overcast	64	65	true	Play
9	Overcast	81	75	false	Play
10	Rain	71	80	true	Don't
11	Rain	65	70	true	Don't
12	Rain	75	80	false	Play
13	Rain	68	80	false	Play
14	Rain	70	96	false	Play

Instance #	Outlook	Temp	Humidity	Windy	class
1	Sunny	75	70	true	Play
2	Sunny	80	90	true	Don't
3	Sunny	85	85	false	Don't
4	Sunny	72	95	true	Don't
5	Sunny	69	70	false	Play
6	Overcast	72	90	true	Play
7	Overcast	83	78	false	Play
8	Overcast	64	65	true	Play
9	Overcast	81	75	false	Play
10	Rain	71	80	true	Don't
11	Rain	65	70	true	Don't
12	Rain	75	80	false	Play
13	Rain	68	80	false	Play
14	Rain	70	96	false	Play

Total 14 training instances

Outlook = sunny: 1,2,3,4,5 P,D,D,D,P

Outlook = overcast: 6,7,8,9 P,P,P,P

Outlook = rain: 10,11,12,13,14 D, D, P, P, P

A categorical feature is partitioned based on its number of possible values

Instance #	Outlook	Temp	Humidity	Windy	class
1	Sunny	75	70	true	Play
2	Sunny	80	90	true	Don't
3	Sunny	85	85	false	Don't
4	Sunny	72	95	true	Don't
5	Sunny	69	70	false	Play
6	Overcast	72	90	true	Play
7	Overcast	83	78	false	Play
8	Overcast	64	65	true	Play
9	Overcast	81	75	false	Play
10	Rain	71	80	true	Don't
11	Rain	65	70	true	Don't
12	Rain	75	80	false	Play
13	Rain	68	80	false	Play
14	Rain	70	96	false	Play

Total 14 training instances

Temperature <= 70: 5,8,11,13,14 P,P, D, P, P

Temperature > 70: 1,2,3,4,6,7,9,10,12 P,D,D,D,P,P,P,D,P

A numerical feature is generally partitioned by choosing a “cutting point”

Steps of Decision Tree Construction

- Select the "best" feature as the root node of the whole tree
- Partition the dataset into subsets using this feature so that the subsets are as "pure" as possible
- After partition by this feature, select the best feature (wrt the subset of training data) as the root node of this sub-tree
- Recursively, until the partitions become pure or almost pure

Measures to Evaluate Which Feature is Best

- Gini index
- Information gain
- Information gain ratio
- T-statistics
- $\chi 2$
- …

Gini Index

Let $\mathcal{U} = \{C_1, ..., C_k\}$ be all the classes. Suppose we are currently at a node and D is the set of those samples that have been moved to this node. Let f be a feature and $d[f]$ be the value of the feature f in a sample d. Let S be a range of values that the feature f can take. Then the Gini index for f in D for the range S is defined as

$$gini_f^D(S) = 1 - \sum_{C_i \in \mathcal{U}} \left(\frac{|\{d \in D \mid d \in C_i,\ d[f] \in S\}|}{|D|} \right)^2$$

The purity of a split of the value range S of an attribute f by some split-point into subranges S_1 and S_2 is then defined as

$$gini_f^D(S_1, S_2) = \sum_{S \in \{S_1, S_2\}} \frac{|\{d \in D \mid d[f] \in S\}|}{|D|} * gini_f^D(S)$$

we choose the feature f and the split-point p that minimizes $gini_f^D(S_1, S_2)$ over all possible alternative features and split-points.

Gini index can be thought of as the expected value of the ratio of the diff of two arbitrary specimens to the mean value of all specimens. Thus the closer it is to 1, the closer you are to the expected "background distribution" of that feature. Conversely, the closer it is to 0, the more "unexpected" the feature is.

$$gini(S) = \frac{\text{diff of two arbitrary specimen in } S}{\text{mean specimen in } S}$$

$$= \frac{prob(\text{getting two specimen of diff class in } S)}{prob(\text{getting specimen of same class in } S)}$$

$$= \frac{\sum_{i \neq j} prob(\text{getting specimen of class } i \text{ in } S) * prob(\text{getting specimen of class } j \text{ in } S)}{1}$$

$$= 1 - \sum prob(\text{getting specimen of class } i \text{ in } S)^2$$

$$= 1 - \sum_{C_i \in \mathcal{U}} \left(\frac{|\{d \in D \mid d \in C_i,\ d[f] \in S\}|}{|D|} \right)^2$$

Gini index can be thought of as the expected value of the ratio of the diff of two arbitrary specimens to the mean value of all specimens. Thus the closer it is to 1, the closer you are to the expected "background distribution" of that feature. Conversely, the closer it is to 0, the more "unexpected" the feature is.

Copyright 2009 © Limsoon Wong

Information Gain

the difference between the information needed to identify the class of a sample in $\mathcal{U}$ before and after the value of the feature f is revealed is

$$Gain(f,\mathcal{U},S_1,S_2) = Ent(f,\mathcal{U},S_1 \cup S_2) - E(f,\mathcal{U},\{S_1,S_2\})$$

where

- $Ent(f,\mathcal{U},S)$ is the class entropy of a range S with respect to a feature f and a collection of classes $\mathcal{U}$. It is defined as

$$Ent(f,\mathcal{U},S) = -\sum_{C_i \in \mathcal{U}} \frac{|\{d \in C_i \mid d[f] \in S\}|}{|\{d \in \bigcup\mathcal{U} \mid d[f] \in S\}|} * \log_2\left(\frac{|\{d \in C_i \mid d[f] \in S\}|}{|\{d \in \bigcup\mathcal{U} \mid d[f] \in S\}|}\right)$$

- $E(f,\mathcal{U},\{S_1,S_2\})$ is the class information entropy of the partition (S_1,S_2). It is defined as

$$E(f,\mathcal{U},S) = \sum_{S_i \in S} \frac{|\{d \in \mathcal{U} \mid d[f] \in S_i\}|}{|\{d \in \mathcal{U} \mid d[f] \in \bigcup S\}|} * Ent(f,\mathcal{U},S_i)$$

The more partitions S has, the bigger this sum is

Then the information gain is the amount of information that is gained by looking at the value of the feature f, and is defined as

$$InfoGain(f,\mathcal{U}) = \max\{Gain(f,\mathcal{U},S_1,S_2) \mid (S_1,S_2) \text{ is a partitioning of the values of } f \text{ in } \bigcup\mathcal{U} \text{ by some point } T\}$$

Information Gain Ratio

$$GainRatio(f,\mathcal{U},S_1,S_2) = \frac{Gain(f,\mathcal{U},S_1,S_2)}{SplitInfo(f,\mathcal{U},S_1,S_2)}$$

where $SplitInfo(f,\mathcal{U},S_1,S_2) = Ent(f,\{\mathcal{U}_f^{S_1},\mathcal{U}_f^{S_2}\},S_1 \cup S_2)$, and $\mathcal{U}_f^S = \bigcup_{C_i \in \mathcal{U}}\{d \in C_i \mid d[f] \in S\}$. Then the information gain ratio is defined as

$$InfoGainRatio(f,\mathcal{U}) = \max\{GainRatio(f,\mathcal{U},S_1,S_2) \mid (S_1,S_2) \text{ is a partitioning of the values of } f \text{ in } \bigcup\mathcal{U} \text{ by some point } T\}$$

Example Use of Decision Tree Methods: Proteomics Approaches to Biomarker Discovery

- **In prostate and bladder cancers (Adam et al. *Proteomics*, 2001)**
- **In serum samples to detect breast cancer (Zhang et al. *Clinical Chemistry*, 2002)**
- **In serum samples to detect ovarian cancer (Petricoin et al. *Lancet*; Li & Rao, *PAKDD* 2004)**

Decision Tree Ensembles

Motivating Example

- h_1, h_2, h_3 are indep classifiers w/ accuracy = 60%
- C_1, C_2 are the only classes
- t is a test instance in C_1
- $h(t) = \text{argmax}_{C \in \{C1,C2\}} |\{h_j \in \{h_1, h_2, h_3\} \mid h_j(t) = C\}|$
- Then prob(h(t) = C_1)

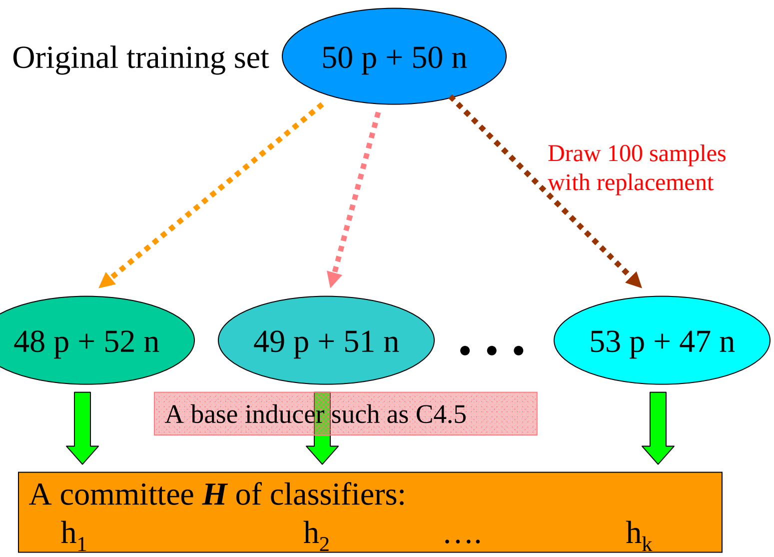
$$= \text{prob}(h_1(t)=C_1 \ \& \ h_2(t)=C_1 \ \& \ h_3(t)=C_1) + \text{prob}(h_1(t)=C_1 \ \& \ h_2(t)=C_1 \ \& \ h_3(t)=C_2) + \text{prob}(h_1(t)=C_1 \ \& \ h_2(t)=C_2 \ \& \ h_3(t)=C_1) + \text{prob}(h_1(t)=C_2 \ \& \ h_2(t)=C_1 \ \& \ h_3(t)=C_1)$$

$$= 60\% * 60\% * 60\% + 60\% * 60\% * 40\% + 60\% * 40\% * 60\% + 40\% * 60\% * 60\% = 64.8\%$$

Bagging

- Proposed by Breiman (1996)
- Also called **B**ootstrap **agg**regating
- Make use of randomness injected to training data

Main Ideas



Decision Making by Bagging

Given a new test sample T

$$bagged(T) = \text{argmax}_{C_j \in \mathcal{U}} |\{h_i \in \mathcal{H} \mid h_i(T) = C_j\}|$$

$$\text{where } \mathcal{U} = \{C_1, \ldots, C_r\}$$

Exercise: What does the above formula mean?

Summary of Ensemble Classifiers

Bagging

Random Forest

AdaBoost.M1

Rules may not be correct when applied to training data

Randomization Trees

CS4

Rules correct

Exercise: Describe the 3 decision tree ensemble classifiers not explained in this ppt

Other Machine Learning Approaches

Outline

- K-Nearest Neighbour
- Bayesian Approach

Exercise: Name and describe one other commonly used machine learning method

K-Nearest Neighbours

How kNN Works

- Given a new case
- Find k “nearest” neighbours, i.e., k most similar points in the training data set
- Assign new case to the same class to which most of these neighbours belong

- A common “distance” measure betw samples x and y is

$$\sqrt{\sum_f (x[f] - y[f])^2}$$

where f ranges over features of the samples

Exercise: What does the formula above mean?

Illustration of kNN (k=8)

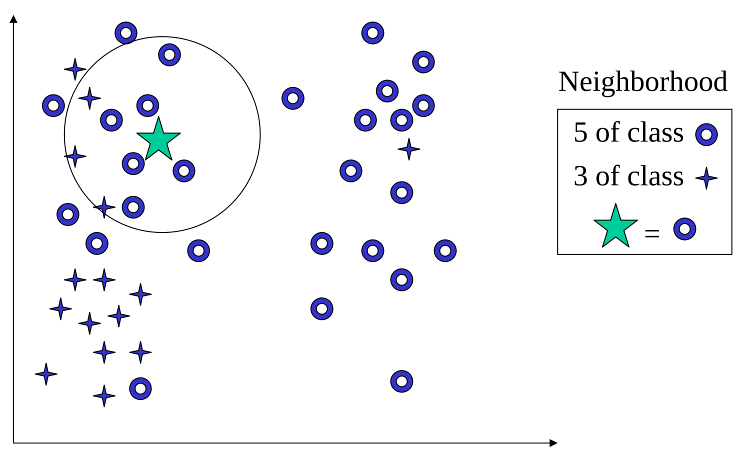


Image credit: Zaki

Some Issues

- Simple to implement
- But need to compare new case against all training cases

⇒ May be slow during prediction

- No need to train
- But need to design distance measure properly

⇒ may need expert for this

- Can't explain prediction outcome

⇒ Can't provide a model of the data

Bayesian Approach

Bayes Theorem

$$P(h|d) = \frac{P(d|h) * P(h)}{P(d)}$$

- *P(h)* = prior prob that hypothesis *h* holds
- *P(d|h)* = prob of observing data *d* given *h* holds
- *P(h|d)* = posterior prob that *h* holds given observed data *d*

Bayesian Approach

- Let *H* be all possible classes. Given a test instance w/ feature vector $\{f_1 = v_1, \ldots, f_n = v_n\}$, the most probable classification is given by

$$\mathrm{argmax}_{h_j \in H} P(h_j | f_1 = v_1, \ldots, f_n = v_n)$$

- Using Bayes Theorem, rewrites to

$$\mathrm{argmax}_{h_j \in H} \frac{P(f_1 = v_1, \ldots, f_n = v_n | h_j) * P(h_j)}{P(f_1 = v_1, \ldots, f_n = v_n)}$$

- Since denominator is independent of h_j, this simplifies to

$$\mathrm{argmax}_{h_j \in H} P(f_1 = v_1, \ldots, f_n = v_n | h_j) * P(h_j)$$

Naïve Bayes

- But estimating $P(f_1=v_1, \ldots, f_n=v_n|h_j)$ accurately may not be feasible unless training data set is sufficiently large
- "Solved" by assuming $f_1, \ldots, f_n$ are conditionally independent of each other
- Then $\text{argmax}_{h_j \in H} P(f_1 = v_1, \ldots, f_n = v_n | h_j) * P(h_j)$

$$= \text{argmax}_{h_j \in H} \prod_i P(f_i = v_i | h_j) * P(h_j)$$

- where $P(h_j)$ and $P(f_i=v_i|h_j)$ can often be estimated reliably from typical training data set

Exercise: How do you estimate $P(h_j)$ and $P(f_j=v_j|h_j)$?

Independence vs Conditional Independence

- **Independence: P(A,B) = P(A) * P(B)**
- **Conditional Independence: P(A,B|C) = P(A|C) * P(B|C)**
- **Indep does not imply conditional indep**
  - Consider tossing a fair coin twice
    - **A is event of getting head in 1st toss**
    - **B is event of getting head in 2nd toss**
    - **C is event of getting exactly one head**
  - Then A={HT, HH}, B={HH, TH} and C={HT, TH}
  - P(A,B|C) =P({HH}|C)=0
  - P(A|C) = P(A,C)/P(C) =P({HT})/P(C)=(1/4)/(1/2) =1/2
  - Similarly, P(B|C) =1/2

Concluding Remarks…

What have we learned?

- **Decision Trees**
- **Decision Trees Ensembles**
  - Bagging
- **Other Methods**
  - K-Nearest Neighbour
  - Bayesian Approach

Any Question?

Acknowledgements

- **The “indep vs conditional indep” example came from Kwok Pui Choi**

References

- L. Breiman, et al. Classification and Regression Trees. Wadsworth and Brooks, 1984
- L. Breiman, Bagging predictors, Machine Learning, 24:123--140, 1996
- L. Breiman, Random forests, Machine Learning, 45:5-32, 2001
- J. R. Quinlan, Induction of decision trees, Machine Learning, 1:81--106, 1986
- J. R. Quinlan, C4.5: Program for Machine Learning. Morgan Kaufmann, 1993
- C. Gini, Measurement of inequality of incomes, The Economic Journal, 31:124--126, 1921
- Jinyan Li et al., Data Mining Techniques for the Practical Bioinformatician, *The Practical Bioinformatician*, Chapter 3, pages 35—70, WSPC, 2004

References

- Y. Freund, et al. Experiments with a new boosting algorithm, ICML 1996, pages 148--156
- T. G. Dietterich, An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization, Machine Learning, 40:139--157, 2000
- J. Li, et al. Ensembles of cascading trees, ICDM 2003, pages 585—588
- Naïve Bayesian Classification, *Wikipedia*, http://en.wikipedia.org/wiki/Naive_Bayesian_classification
- Hidden Markov Model, Wikipedia, http://en.wikipedia.org/wiki/Hidden_Markov_model