Pattern Recognition:
Readings: Ch 4: 4.1-4.6, 4.8-4.10, 4.13

- statistical vs. structural
- terminology
- nearest mean & nearest neighbor
- naive Bayes classifier (from Mitchell)
- decision trees, neural nets, SVMs (quick)

Pattern Recognition

Pattern recognition is:

1. The name of the journal of the Pattern Recognition Society.

2. A research area in which patterns in data are found, recognized, discovered, …whatever.

3. A catchall phrase that includes classification, clustering, and data mining.

4. Also called “machine learning,” especially in CS.

Two Schools of Thought

1. Statistical Pattern Recognition

   The data is reduced to vectors of numbers and statistical techniques are used for the tasks to be performed.

2. Structural Pattern Recognition

   The data is converted to a discrete structure (such as a grammar or a graph) and the techniques are related to computer science subjects (such as parsing and graph matching).

In this course

1. How should objects to be classified be represented?

2. What algorithms can be used for recognition (or matching)?

3. How should learning (training) be done?

Classification in Statistical PR




- A class is a set of objects having some important properties in common.

- A feature extractor is a program that inputs the data (image) and extracts features that can be used in classification.

- A classifier is a program that inputs the feature vector and assigns it to one of a set of designated classes or to the "reject" class.

Feature Vector Representation

- $X=[x_1, x_2, \ldots , x_n]$, each x_j a real number
- x_j may be an object measurement
- x_j may be a count of object parts

```
000000000010000000000
000000000110000000000
000000000101000000000
000000001000010000000
000000010000010000000
000001000000001000000
000010000000000100000
000011001111111110000
000011111110000010000
000110000000000011000
000100000000000001100
001100000000000000100
001100000000000000110
001000000000000000010
001000000000000000010
011000000000000000010
010000000000000000000
000000000000000000000
```

```
000000000011110000000
000000011000011000000
000000110000000110000
000011000000000011000
000010000000000001000
000011000000000010000
000001110000000100000
000000111001111100000
000000000111100000000
000000110001110000000
000011000000000110000
000110000000000011000
001100000000000001000
001000000000000001100
000100000000000011000
000110000000000010000
000010000000000110000
000000111111110000000
```

Example: [area, height, width, #holes, #strokes, cx, cy]

Possible Features for Character Recognition

(class) character	area	height	width	number #holes	number #strokes	(cx,cy) center	best axis	least inertia
'A'	medium	high	3/4	1	3	1/2,2/3	90	medium
'B'	medium	high	3/4	2	1	1/3,1/2	90	large
'8'	medium	high	2/3	2	0	1/2,1/2	90	medium
'0'	medium	high	2/3	1	0	1/2,1/2	90	large
'1'	low	high	1/4	0	1	1/2,1/2	90	low
'W'	high	high	1	0	4	1/2,2/3	90	large
'X'	high	high	3/4	0	2	1/2,1/2	?	large
'*'	medium	low	1/2	0	0	1/2,1/2	?	large
'-'	low	low	2/3	0	1	1/2,1/2	0	low
'/'	low	high	2/3	0	1	1/2,1/2	60	low

Feature values can be numbers, vectors of numbers, strings: any datatype.

Some Terminology

- Classes: set of m known categories of objects

  (a) might have a known description for each

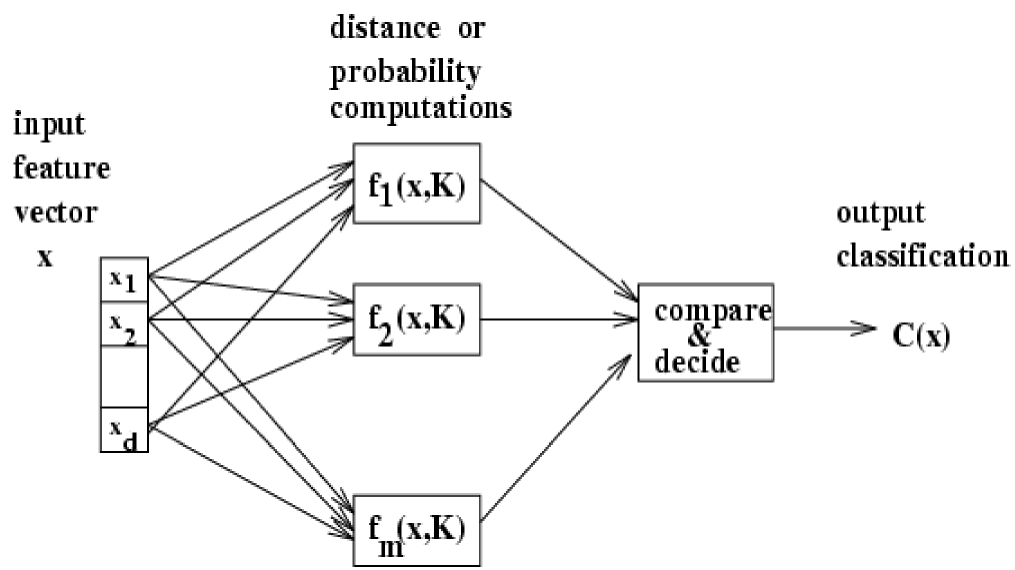
  (b) might have a set of samples for each
- Reject Class:

  a generic class for objects not in any of the designated known classes
- Classifier:

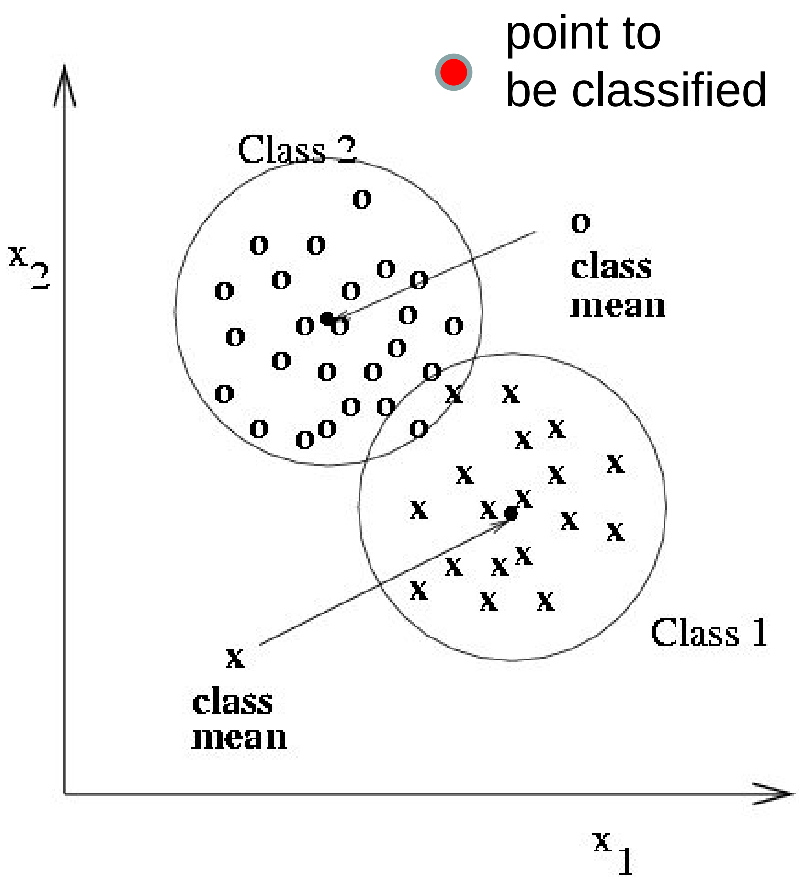
  Assigns object to a class based on features

Discriminant functions

- Functions f(x, K) perform some computation on feature vector x
- Knowledge K from training or programming is used
- Final stage determines class

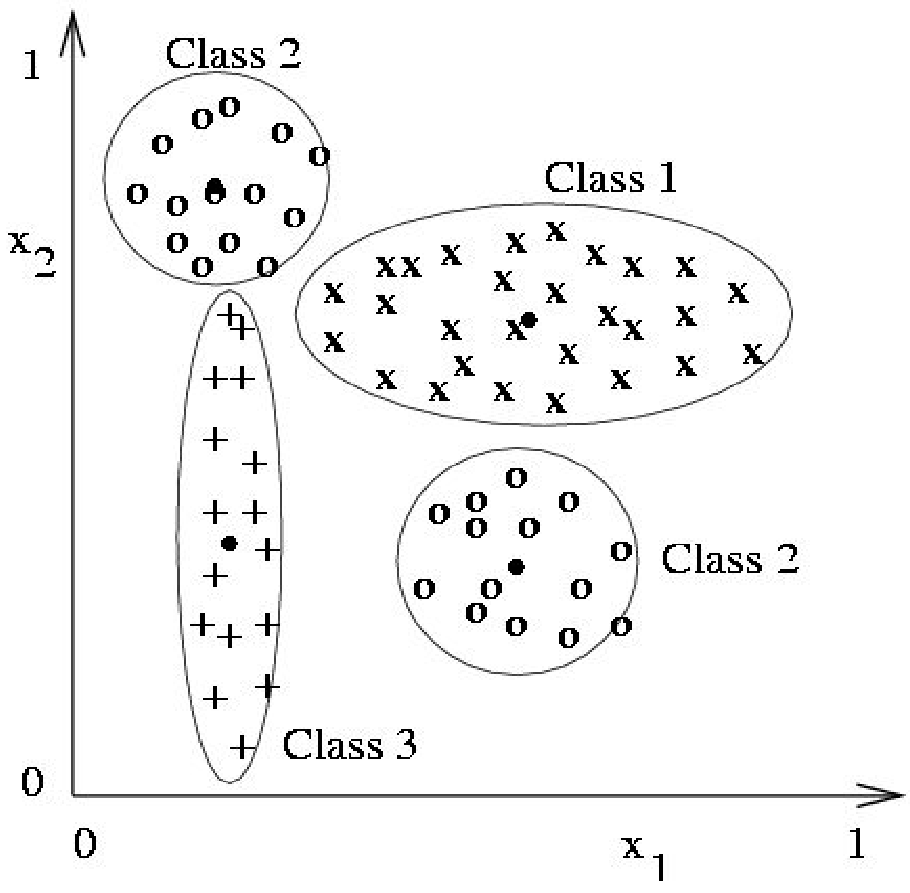


Classification using Nearest Class Mean



- Compute the Euclidean distance between feature vector X and the mean of each class.
- Choose closest class, if close enough (reject otherwise)

Nearest mean might yield poor results with complex structure



- Class 2 has two modes; where is its mean?
- But if modes are detected, two subclass mean vectors can be used

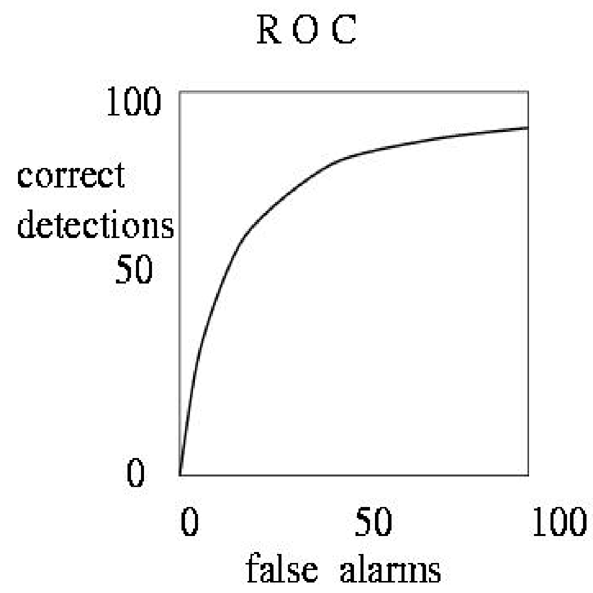
Nearest Neighbor Classification

- Keep all the training samples in some efficient look-up structure.
- Find the nearest neighbor of the feature vector to be classified and assign the class of the neighbor.
- Can be extended to K nearest neighbors.

Receiver Operating Curve ROC

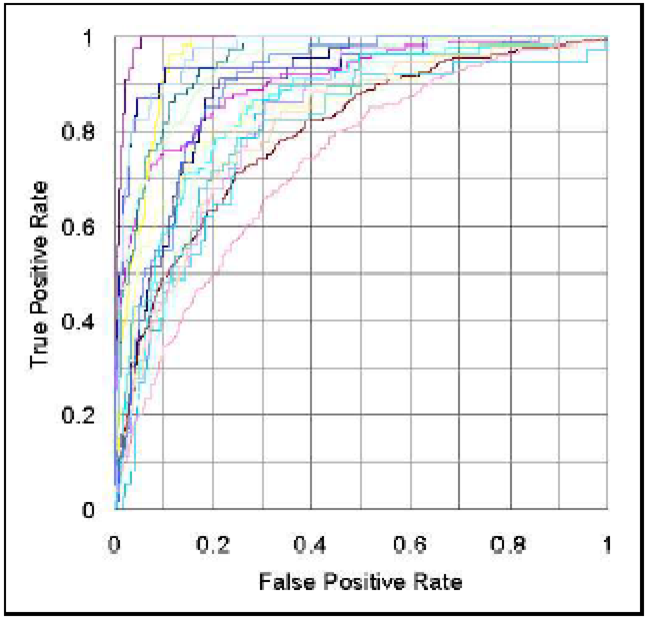
- Plots correct detection rate versus false alarm rate

- Generally, false alarms go up with attempts to detect higher percentages of known objects



actual input object	decision	error type?
frack	frack	correct alarm (no error)
not a frack	frack	false alarm (error)
frack	not a frack	false dismissal (error)
not a frack	not a frack	correct dismissal (no error)

A recent ROC from our work:



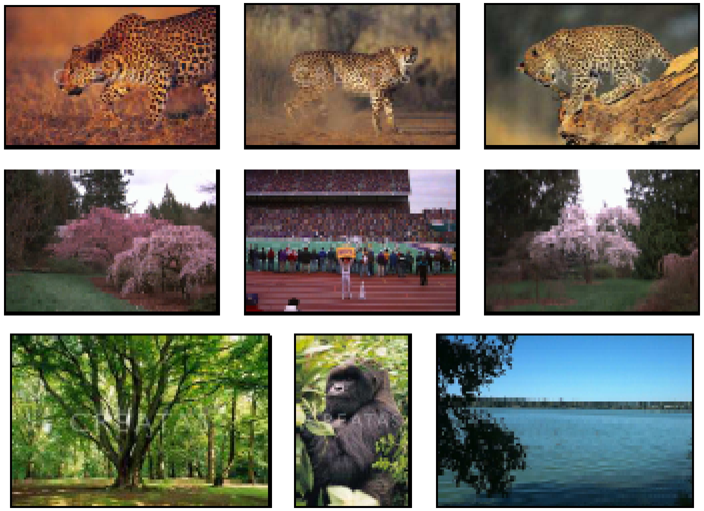


Figure 4: The top 3 test results for cheetah, cherry tree, and tree.

Confusion matrix shows empirical performance

		class j output by the pattern recognition system										
		’0’	’1’	’2’	’3’	’4’	’5’	’6’	’7’	’8’	’9’	’R’
	’0’	97	0	0	0	0	0	1	0	0	1	1
	’1’	0	98	0	0	1	0	0	1	0	0	0
true	’2’	0	0	96	1	0	1	0	1	0	0	1
object	’3’	0	0	2	95	0	1	0	0	1	0	1
class	’4’	0	0	0	0	98	0	0	0	0	2	0
	’5’	0	0	0	1	0	97	0	0	0	0	2
i	’6’	1	0	0	0	0	1	98	0	0	0	0
	’7’	0	0	1	0	0	0	0	98	0	0	1
	’8’	0	0	0	1	0	0	1	0	96	1	1
	’9’	1	0	0	0	3	0	0	0	1	95	0

Confusion may be unavoidable between some classes,
for example, between 9’s and 4’s.

face or not face

In a 2-class problem where the class is either C or not C the confusion matrix looks like this:

True Class	Classifier Output: C	Classifier Output: not C
C	TP	FN
not C	FP	TN

- TP is the number of true positives. It's a C, and classifier output is C
- FN is the number of false negatives. It's a C, and classifier output is not C.
- TN is the number of true negatives. It's not C, and classifier output is not C.
- FP is the number of false positives. It's not C, and classifier output is C.

Classifiers often used in CV

- Naive Bayes Classifier
- Decision Tree Classifiers
- Artificial Neural Net Classifiers
- Support Vector Machines
- EM as a Classifier
- Bayesian Networks (Graphical Models)

Naive Bayes Classifier

- Uses Bayes rule for classification
- One of the simpler classifiers
- Worked well for face detection in 576
- Part of the free WEKA suite of classifiers

Bayes Rule

$$P(Y|X) = \frac{P(X|Y)P(Y)}{P(X)}$$

Which is shorthand for:

$$P(Y = y_i|X = x_j) = \frac{P(X = x_j|Y = y_i)P(Y = y_i)}{P(X = x_j)}$$

This slide and those following are from Tom Mitchell's course in Machine Learning.

Bayes Theorem

Does patient have cancer or not?

A patient takes a lab test and the result comes back positive. The test returns a correct positive result in only 98% of the cases in which the disease is actually present, and a correct negative result in only 97% of the cases in which the disease is not present. Furthermore, .008 of the entire population have this cancer.

$$P(cancer) = .008 \qquad P(\neg cancer) = .992$$
$$P(+|cancer) = .980 \qquad P(-|cancer) = .020$$
$$P(+|\neg cancer) = .030 \qquad P(-|\neg cancer) = .970$$

Basic Formulas for Probabilities

- *Product Rule*: probability $P(A \wedge B)$ of a conjunction of two events A and B:

$$P(A \wedge B) = P(A|B)P(B) = P(B|A)P(A)$$

- *Sum Rule*: probability of a disjunction of two events A and B:

$$P(A \vee B) = P(A) + P(B) - P(A \wedge B)$$

- *Theorem of total probability*: if events $A_1, \ldots, A_n$ are mutually exclusive with $\sum_{i=1}^{n} P(A_i) = 1$, then

$$P(B) = \sum_{i=1}^{n} P(B|A_i)P(A_i)$$

Naive Bayes Classifier

Assume target function $f : X \rightarrow V$, where each instance x described by attributes $\langle a_1, a_2 \ldots a_n \rangle$.
Most probable value of $f(x)$ is:

$$
\begin{aligned}
v_{MAP} &= \underset{v_j \in V}{\operatorname{argmax}} P(v_j | a_1, a_2 \ldots a_n) \\
&= \underset{v_j \in V}{\operatorname{argmax}} \frac{P(a_1, a_2 \ldots a_n | v_j) P(v_j)}{P(a_1, a_2 \ldots a_n)} \\
&= \underset{v_j \in V}{\operatorname{argmax}} P(a_1, a_2 \ldots a_n | v_j) P(v_j)
\end{aligned}
$$

MAP: maximum a posteriori probability.

by Bayes Rule

Assume $P(a_1,...,a_n)$ same for all $a_1,...a_n$.

Naive Bayes assumption:

$$P(a_1, a_2 \ldots a_n | v_j) = \prod_i P(a_i | v_j)$$

Conditional independence

which gives

Naive Bayes classifier: $v_{NB} = \underset{v_j \in V}{\operatorname{argmax}} P(v_j) \prod_i P(a_i | v_j)$

Naive Bayes Algorithm

Naive_Bayes_Learn(*examples*)

For each target value v_j

$\hat{P}(v_j) \leftarrow$ estimate $P(v_j)$

For each attribute value a_i of each attribute a

$\hat{P}(a_i|v_j) \leftarrow$ estimate $P(a_i|v_j)$

Classify_New_Instance(x)

$$v_{NB} = \underset{v_j \in V}{\operatorname{argmax}} \hat{P}(v_j) \prod_{a_i \in x} \hat{P}(a_i|v_j)$$

Elaboration

The set of examples is actually a set of preclassified feature vectors called the **training set**.

From the training set, we can estimate the a priori probability of each class:

P(C) = # training vectors from class C / total # of training vectors

For each class **C**, attribute **a,** and possible value for that attribute $\mathbf{a_i}$, we can estimate the conditional probability:

$P(a_i | C_j)$ = # training vectors from class C_j in which value(a) = a_i

Training Examples

features | class | some estimates

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
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

P(y) = 9/14

P(n) = 5/14

P(sun | y) = 2/9

P(cool | y) = 3/9

P(high | y) = 3/9

P(strong | y) = 3/9

P(y)P(sun | y)P(cool | y)P(high | y)P(strong | y) =
(9/14) * (2/9) * (3/9) * (3/9) * (3/9) =
.005

Naive Bayes: Example

Consider *PlayTennis* again, and new instance

$\langle Outlk = sun, Temp = cool, Humid = high, Wind = strong$

Want to compute:

$$v_{NB} = \underset{v_j \in V}{\operatorname{argmax}} P(v_j) \prod_i P(a_i|v_j)$$

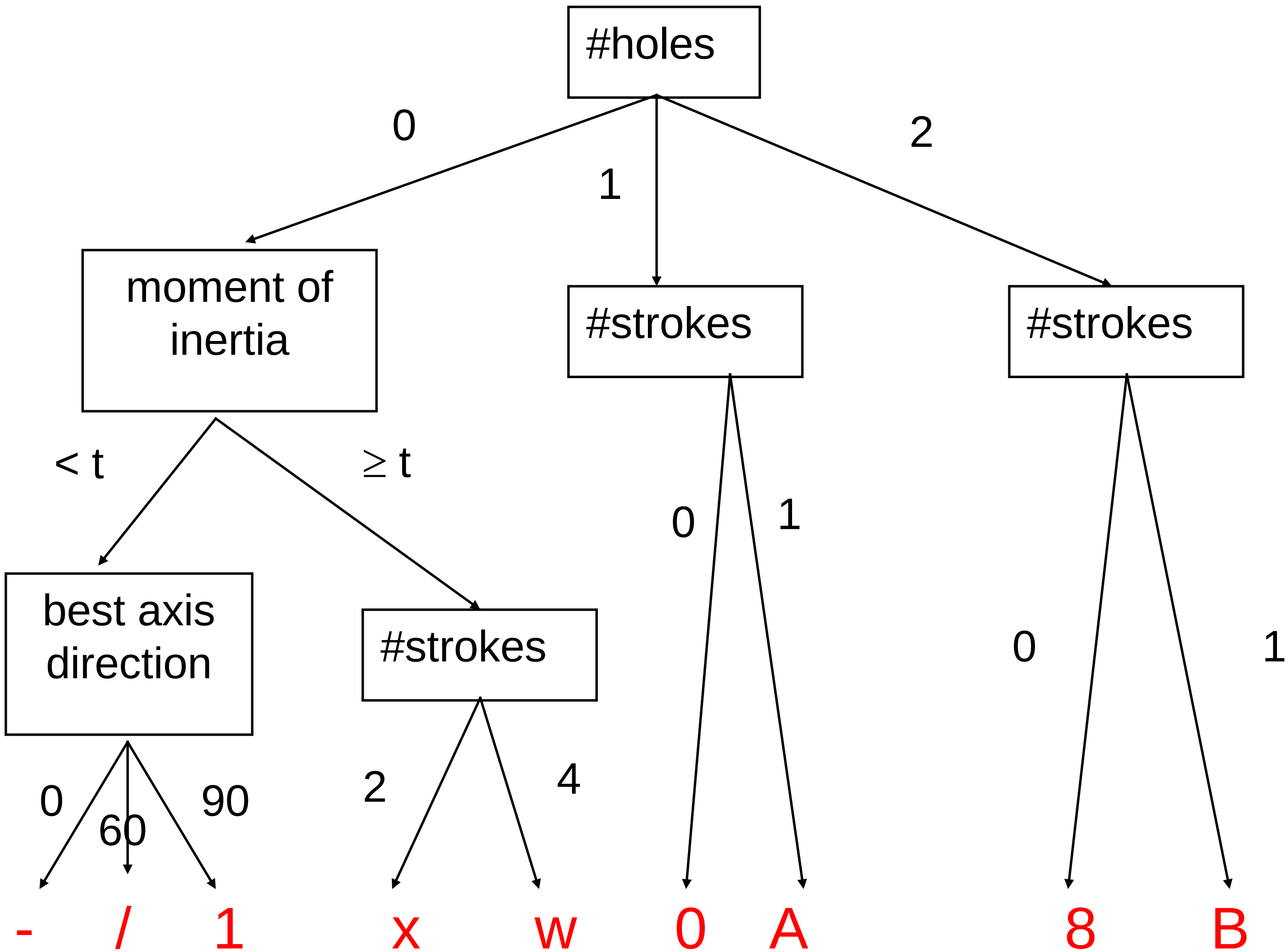
$$P(y)\ P(sun|y)\ P(cool|y)\ P(high|y)\ P(strong|y) = .005$$

$$P(n)\ P(sun|n)\ P(cool|n)\ P(high|n)\ P(strong|n) = .021$$

$$\rightarrow v_{NB} = n$$

This is a prediction. If it is sunny, cool, highly humid, and strong wind, it is more likely that we won't play tennis than that we will.

Decision Trees



Decision Tree Characteristics

1. Training

   How do you construct one from training data?
   Entropy-based Methods

2. Strengths

   Easy to Understand

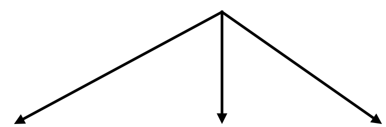
3. Weaknesses

   Overfitting (the classifier fits the training data very well, but not new unseen data)

Entropy-Based Automatic Decision Tree Construction

Training Set S
x1=(f11,f12,…f1m)
x2=(f21,f22,   f2m)
.
.
xn=(fn1,f22,   f2m)

Node 1
What feature
should be used?



What values?

Quinlan suggested information gain in his ID3 system and later the gain ratio, both based on entropy.

Entropy

Given a set of training vectors S, if there are c classes,

$$\text{Entropy}(S) = \sum_{i=1}^{c} -p_i \log_2 (p_i)$$

Where p_i is the proportion of category i examples in S.

If all examples belong to the same category, the entropy is 0 (no discrimination).

The greater the discrimination power, the larger the entropy will be.

Information Gain

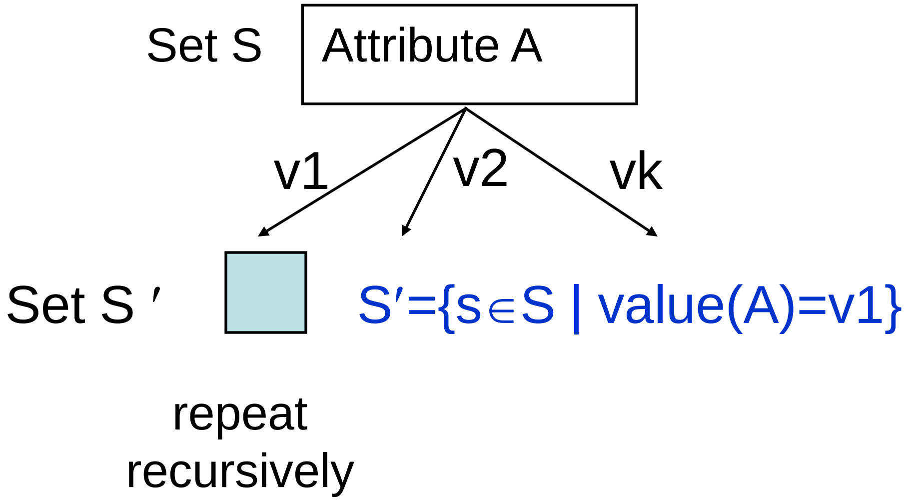
The information gain of an attribute A is the expected reduction in entropy caused by partitioning on this attribute.

$$\text{Gain}(S,A) = \text{Entropy}(S) - \sum_{v \in \text{Values}(A)} \frac{|Sv|}{|S|} \text{Entropy}(S_v)$$

where S_v is the subset of S for which attribute A has value v.

Choose the attribute A that gives the maximum information gain.

Information Gain (cont)



The attribute A selected at the top of the tree is the one with the highest information gain.

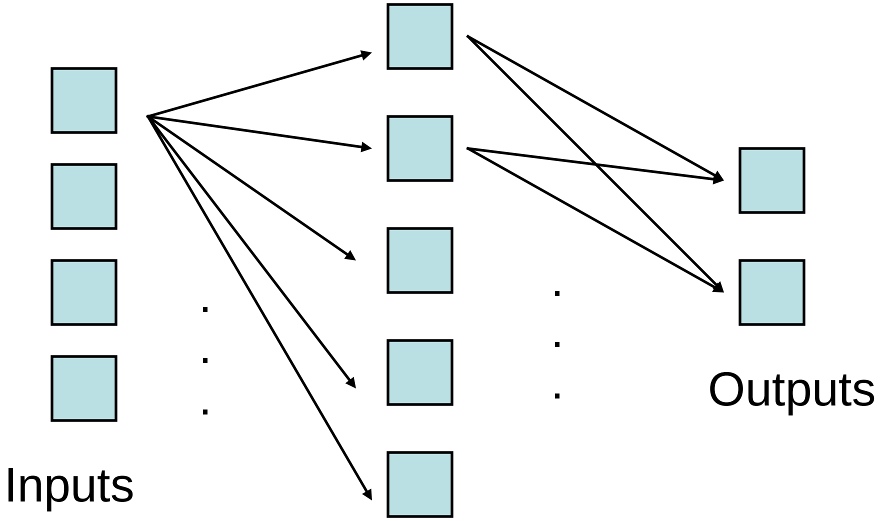
Subtrees are constructed for each possible value vi of attribute A.

The rest of the tree is constructed in the same way.

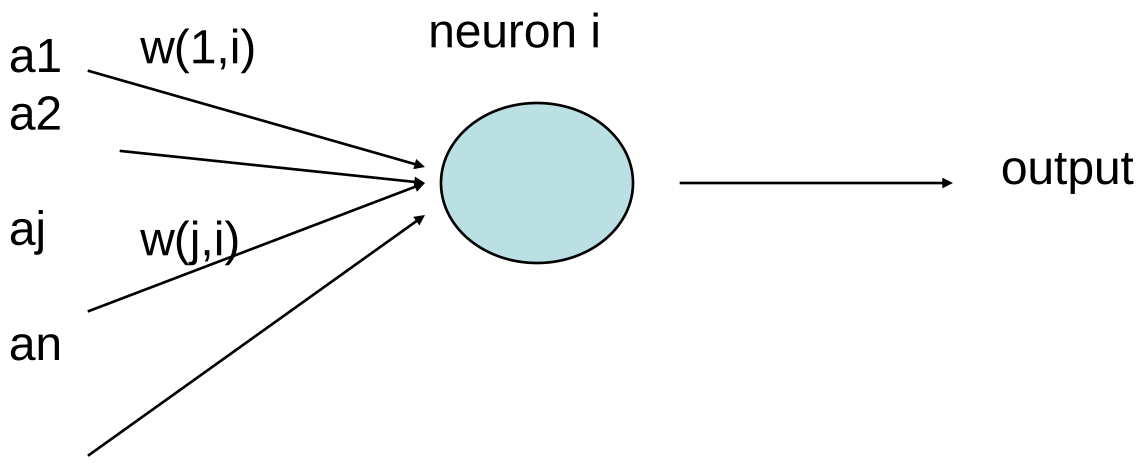
Artificial Neural Nets

Artificial Neural Nets (ANNs) are networks of artificial neuron nodes, each of which computes a simple function.

An ANN has an input layer, an output layer, and "hidden" layers of nodes.



Node Functions



output = g (∑ aj * w(j,i))

Function g is commonly a step function, sign function, or sigmoid function (see text).

Neural Net Learning

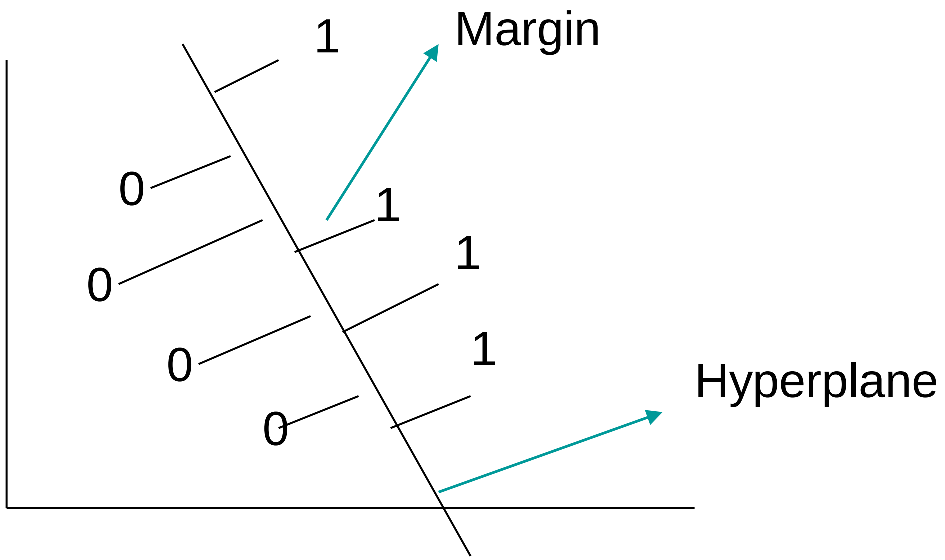
Beyond the scope of this course.

Support Vector Machines (SVM)

Support vector machines are learning algorithms that try to find a hyperplane that separates the differently classified data the most. They are based on two key ideas:

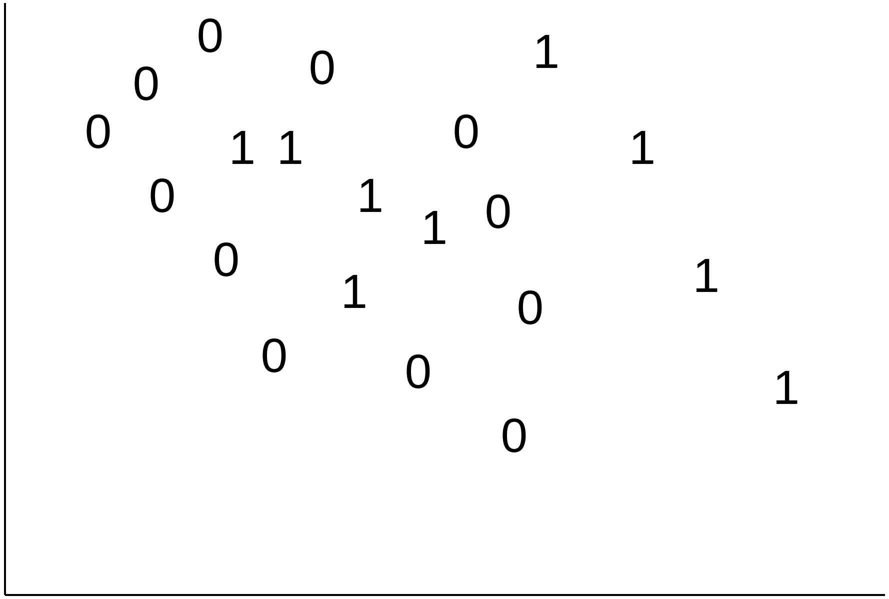
- Maximum margin hyperplanes
- A kernel ‘trick’.

Maximal Margin



Find the hyperplane with maximal margin for all the points. This originates an optimization problem which has a unique solution (convex problem).

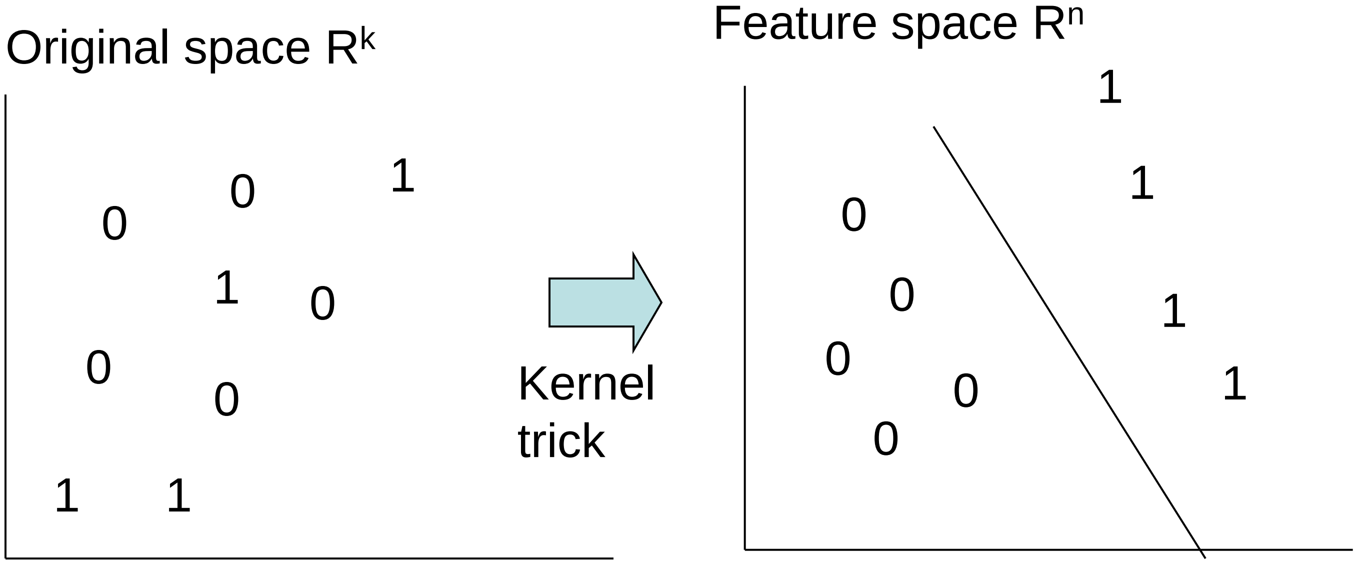
Non-separable data



What can be done if data cannot be separated with a hyperplane?

The kernel trick

The SVM algorithm implicitly maps the original data to a feature space of possibly infinite dimension in which data (which is not separable in the original space) becomes separable in the feature space.



EM for Classification

- The EM algorithm was used as a clustering algorithm for image segmentation.

- It can also be used as a classifier, by creating a Gaussian “model” for each class to be learned.