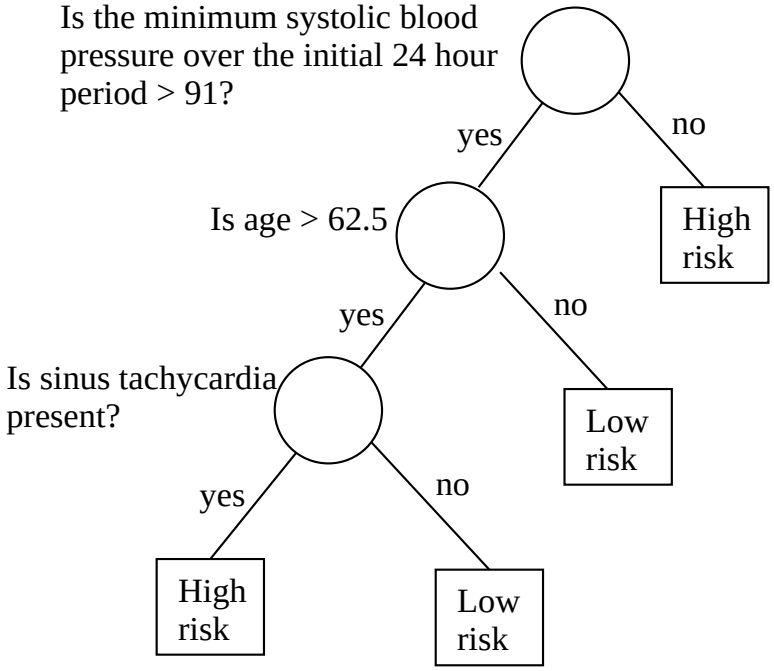
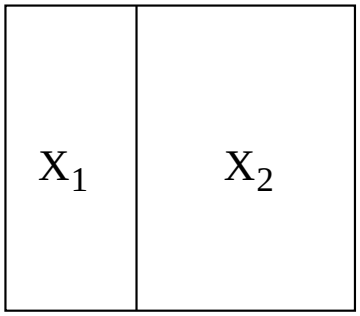
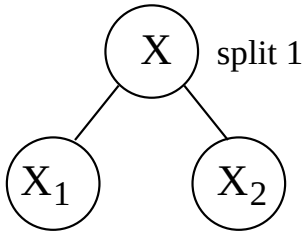
Tree Structured Classifier

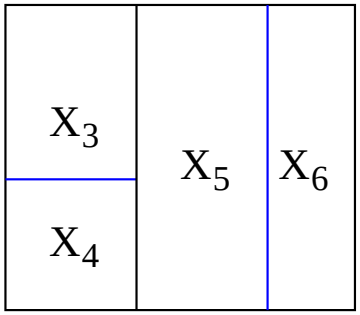
- Reference: *Classification and Regression Trees* by L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone, Chapman & Hall, 1984.
- A Medical Example (CART):
  - Predict high risk patients who will not survive at least 30 days on the basis of the initial 24-hour data.
  - 19 variables are measured during the first 24 hours. These include blood pressure, age, etc.
  - A tree structure classification rule is as follows:

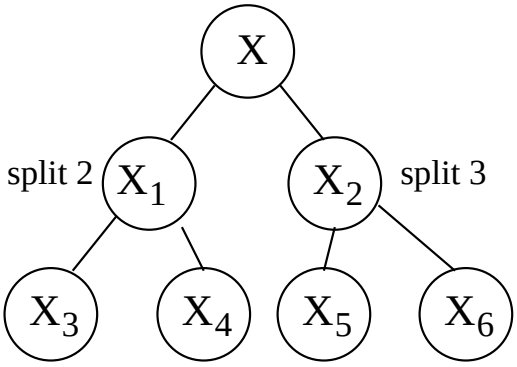


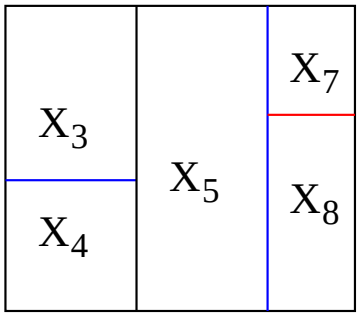
- Denote the feature space by $\mathcal{X}$. The input vector $X \in \mathcal{X}$ contains p features X_1, X_2, ..., X_p, some of which may be categorical.
- Tree structured classifiers are constructed by repeated splits of subsets of $\mathcal{X}$ into two descendant subsets, beginning with $\mathcal{X}$ itself.
- Definitions: *node*, *terminal node (leaf node)*, *parent node*, *child node*.
- The union of the regions occupied by two child nodes is the region occupied by their parent node.
- Every leaf node is assigned with a class. A query is associated with class of the leaf node it lands in.
- Notation:
  - A node is denoted by t. Its left child node is denoted by t_L and right by t_R.
  - The collection of all the nodes is denoted by T; and the collection of all the leaf nodes by $\tilde{T}$.
  - A split is denoted by s. The set of splits is denoted by $\mathcal{S}$.

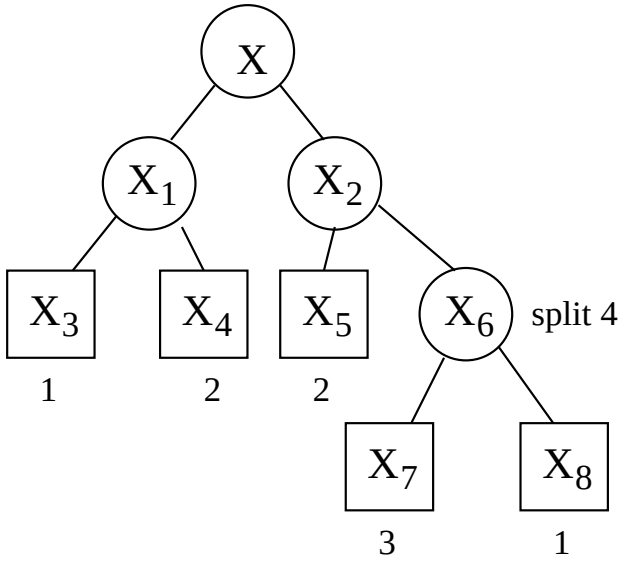

X1
X2


X
split 1
X1
X2


X3
X4
X5
X6


X
split 2
X1
X2
split 3
X3
X4
X5
X6


X3
X4
X5
X7
X8


X
X1
X2
X3
X4
X5
X6
split 4
X7
X8
1
2
2
3
1

The Three Elements

- The construction of a tree involves the following three elements:
  1. The selection of the splits.
  2. The decisions when to declare a node terminal or to continue splitting it.
  3. The assignment of each terminal node to a class.
- In particular, we need to decide the following:
  1. A set $\mathcal{Q}$ of binary questions of the form $\{\text{Is } X \in A?\}$, $A \subseteq \mathcal{X}$.
  2. A goodness of split criterion $\Phi(s, t)$ that can be evaluated for any split s of any node t.
  3. A stop-splitting rule.
  4. A rule for assigning every terminal node to a class.

Standard Set of Questions

- The input vector $X = (X_1, X_2, ..., X_p)$ contains features of both categorical and ordered types.
- Each split depends on the value of only a *unique* variable.
- For each ordered variable X_j, $\mathcal{Q}$ includes all questions of the form

  $$\{\text{Is } X_j \leq c?\}$$

  for all real-valued c.

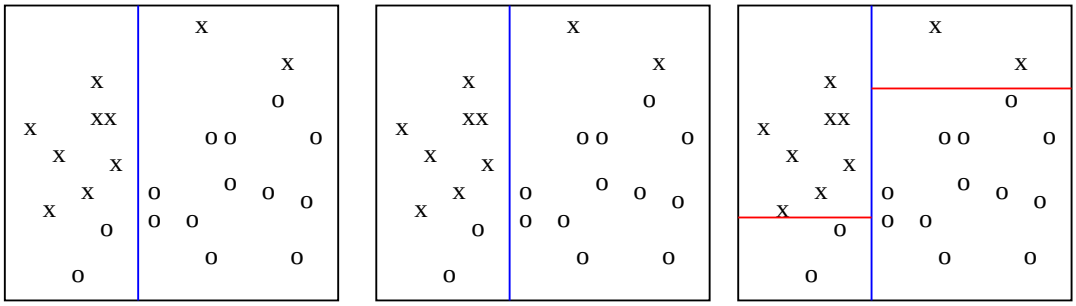
  – Since the training data set is finite, there are only finitely many distinct splits that can be generated by the question $\{\text{Is } X_j \leq c?\}$.
- If X_j is categorical, taking values, say in$\{1, 2, ..., M\}$, then $\mathcal{Q}$ contains all questions of the form

  $$\{\text{Is } X_j \in A?\} \ .$$

  A ranges over all subsets of $\{1, 2, ..., M\}$.
- The splits for all p variables constitute the standard set of questions.

Goodness of Split

- The goodness of split is measured by an impurity function defined for each node.



- Intuitively, we want each leaf node to be "pure", that is, one class dominates.
- *Definition*: *An impurity function is a function ϕ defined on the set of all K-tuples of numbers $(p_1, ..., p_K)$ satisfying $p_j \geq 0$, $j = 1$, ..., K, $\sum_j p_j = 1$ with the properties:*
  1. *ϕ is a maximum only at the point $(\frac{1}{K}, \frac{1}{K}, ..., \frac{1}{K})$.*
  2. *ϕ achieves its minimum only at the points $(1, 0, ..., 0)$, $(0, 1, 0, ..., 0)$, ..., $(0, 0, ..., 0, 1)$.*
  3. *ϕ is a symmetric function of p_1, ..., p_K, i.e., if you permute p_j, ϕ remains constant.*

- *Definition: Given an impurity function ϕ, define the impurity measure $i(t)$ of a node t as*

$$i(t) = \phi(p(1 \mid t), p(2 \mid t), ..., p(K \mid t)) \ ,$$

where $p(j \mid t)$ is the estimated probability of class j within node t.

- Goodness of a split s for node t, denoted by $\Phi(s, t)$, is defined by

$$\Phi(s, t) = \Delta i(s, t) = i(t) - p_R i(t_R) - p_L i(t_L) \ ,$$

where p_R and p_L are the proportions of the samples in node t that go to the right node t_R and the left node t_L respectively.

- Define $I(t) = i(t)p(t)$, that is, the impurity function of node t weighted by the estimated proportion of data that go to node t.
- The impurity of tree T, $I(T)$ is defined by

$$I(T) = \sum_{t \in \tilde{T}} I(t) = \sum_{t \in \tilde{T}} i(t)p(t) \ .$$

- Note for any node t the following equations hold:

$$\begin{gathered} p(t_L) + p(t_R) = p(t) \\ p_L = p(t_L)/p(t), \quad p_R = p(t_R)/p(t) \\ p_L + p_R = 1 \end{gathered}$$

- Define

$$\begin{aligned} \Delta I(s,t) &= I(t) - I(t_L) - I(t_R) \\ &= p(t)i(t) - p(t_L)i(t_L) - p(t_R)i(t_R) \\ &= p(t)(i(t) - p_L i(t_L) - p_R i(t_R)) \\ &= p(t)\Delta i(s,t) \end{aligned}$$

- Possible impurity function:
  1. Entropy: $\sum_{j=1}^{K} p_j \log \frac{1}{p_j}$. If $p_j = 0$, use the limit $\lim_{p_j \to 0} p_j \log p_j = 0$.
  2. Misclassification rate: $1 - \max_j p_j$.
  3. Gini index: $\sum_{j=1}^{K} p_j(1 - p_j) = 1 - \sum_{j=1}^{K} p_j^2$.
- Gini index seems to work best in practice for many problems.
- The *twoing rule*: At a node t, choose the split s that maximizes

$$\frac{p_L p_R}{4} \left[\sum_j |p(j \mid t_L) - p(j \mid t_R)| \right]^2 .$$

Estimate the posterior probabilities of classes in each node:

- The total number of samples is N and the number of samples in class j, $1 \leq j \leq K$, is N_j.
- The number of samples going to node t is $N(t)$; the number of samples with class j going to node t is $N_j(t)$.
  - $\sum_{j=1}^{K} N_j(t) = N(t)$.
  - $N_j(t_L) + N_j(t_R) = N_j(t)$.
  - For a full tree (balanced), the sum of $N(t)$ over all the t's at the same level is N.
- Denote the prior probability of class j by π_j.
  - The priors π_j can be estimated from the data by N_j/N.
  - Sometimes priors are given before-hand.
- The estimated probability of a sample in class j going to node t is $p(t \mid j) = N_j(t)/N_j$.
  - $p(t_L \mid j) + p(t_R \mid j) = p(t \mid j)$.
  - For a full tree, the sum of $p(t \mid j)$ over all t's at the same level is 1.

- The joint probability of a sample being in class j and going to node t is thus:

$$p(j,t) = \pi_j p(t \mid j) = \pi_j N_j(t)/N_j \ .$$

- The probability of any sample going to node t is:

$$p(t) = \sum_{j=1}^{K} p(j,t) = \sum_{j=1}^{K} \pi_j N_j(t)/N_j \ .$$

  Note $p(t_L) + p(t_R) = p(t)$.

- The probability of a sample being in class j given that it goes to node t is:

$$p(j \mid t) = p(j,t)/p(t) \ .$$

  For any t, $\sum_{j=1}^{K} p(j \mid t) = 1$.

- When $\pi_j = N_j/N$, we have the following simplification:
  - $p(j \mid t) = N_j(t)/N(t)$.
  - $p(t) = N(t)/N$.
  - $p(j,t) = N_j(t)/N$.

Stopping Criteria

- A simple criteria: stop splitting a node t when

$$\max_{s \in \mathcal{S}} \Delta I(s,t) < \beta \ ,$$

  where β is a chosen threshold.

- The above stopping criteria is unsatisfactory.
  - A node with a small decrease of impurity after one step of splitting may have a large decrease after multiple levels of splits.

Class Assignment Rule

- A class assignment rule assigns a class $j = \{1, ..., K\}$ to every terminal node $t \in \tilde{T}$. The class assigned to node $t \in \tilde{T}$ is denoted by $\kappa(t)$.
- For 0-1 loss, the class assignment rule is:

$$\kappa(t) = \arg\max_j p(j \mid t) \ .$$

- The *resubstitution estimate* $r(t)$ of the probability of misclassification, given that a case falls into node t is

$$r(t) = 1 - \max_j p(j \mid t) = 1 - p(\kappa(t) \mid t) \ .$$

- Denote $R(t) = r(t)p(t)$.
- The resubstitution estimate for the overall misclassification rate $R(T)$ of the tree classifier T is:

$$R(T) = \sum_{t \in \tilde{T}} R(t) \ .$$

- *Proposition: For any split of a node t into t_L and t_R,*

$$R(t) \geq R(t_L) + R(t_R) \ .$$

Proof:
Denote $j^* = \kappa(t)$.

$$\begin{aligned} p(j^* \mid t) &= p(j^*, t_L \mid t) + p(j^*, t_R \mid t) \\ &= p(j^* \mid t_L)p(t_L \mid t) + p(j^* \mid t_R)p(t_R \mid t) \\ &= p_L p(j^* \mid t_L) + p_R p(j^* \mid t_R) \\ &\leq p_L \max_j p(j \mid t_L) + p_R \max_j p(j \mid t_R) \end{aligned}$$

Hence,

$$\begin{aligned} r(t) &= 1 - p(j^* \mid t) \\ &\geq 1 - \left[p_L \max_j p(j \mid t_L) + p_R \max_j p(j \mid t_R) \right] \\ &= p_L(1 - \max_j p(j \mid t_L)) + p_R(1 - \max_j p(j \mid t_R)) \\ &= p_L r(t_L) + p_R r(t_R) \end{aligned}$$

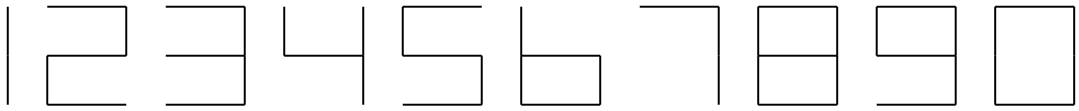
Finally,

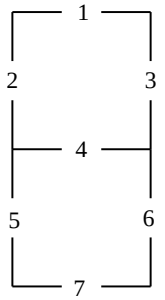
$$\begin{aligned} R(t) &= p(t)r(t) \\ &\geq p(t)p_L r(t_L) + p(t)p_R r(t_R) \\ &= p(t_L)r(t_L) + p(t_R)r(t_R) \\ &= R(t_L) + R(t_R) \end{aligned}$$

Digit Recognition Example (CART)

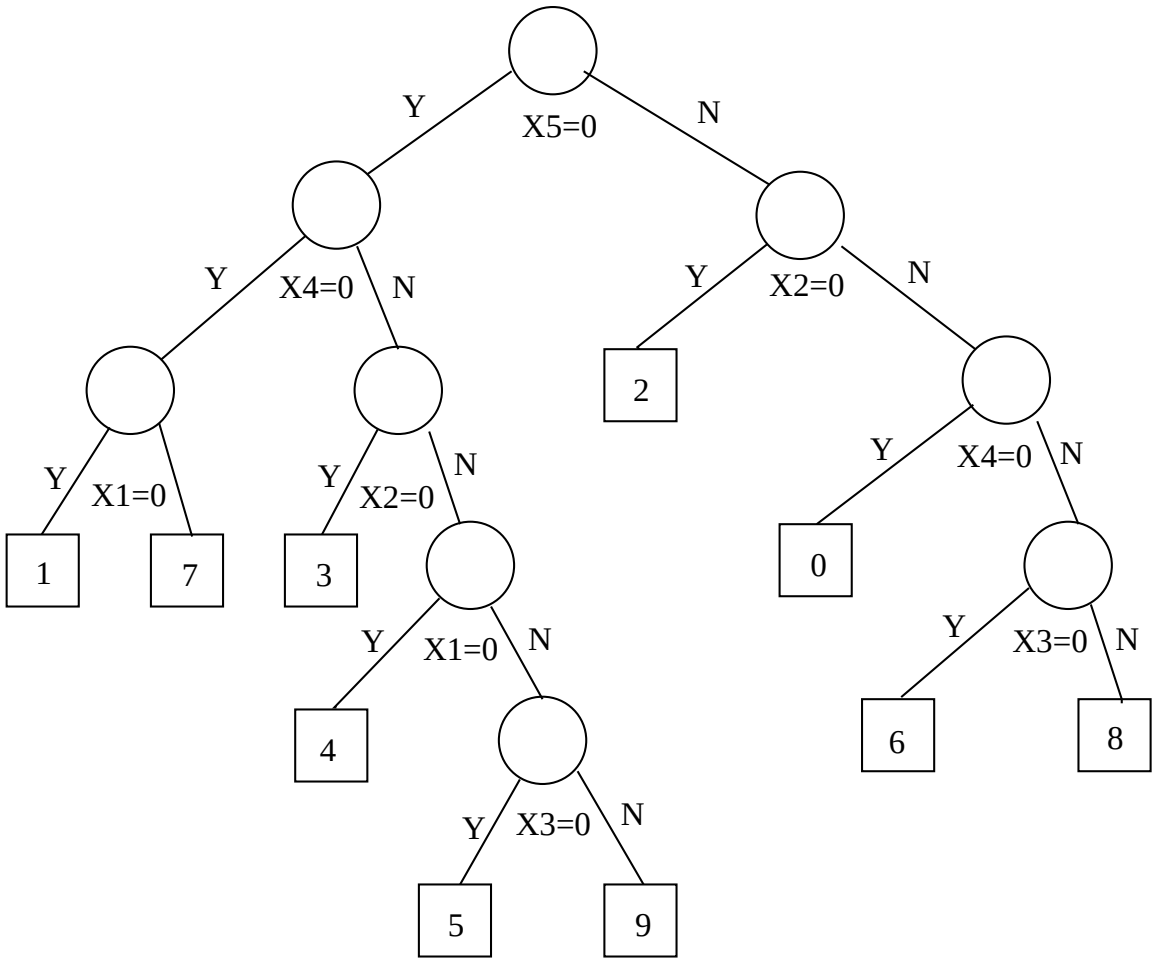
- The 10 digits are shown by different on-off combinations of seven horizontal and vertical lights.
- Each digit is represented by a 7-dimensional vector of zeros and ones. The ith sample is $x_i = (x_{i1}, x_{i2}, ..., x_{i7})$. If $x_{ij} = 1$, the jth light is on; if $x_{ij} = 0$, the jth light is off.

Digit	$x_{\cdot 1}$	$x_{\cdot 2}$	$x_{\cdot 3}$	$x_{\cdot 4}$	$x_{\cdot 5}$	$x_{\cdot 6}$	$x_{\cdot 7}$
1	0	0	1	0	0	1	0
2	1	0	1	1	1	0	1
3	1	0	1	1	0	1	1
4	0	1	1	1	0	1	0
5	1	1	0	1	0	1	1
6	1	1	0	1	1	1	1
7	1	0	1	0	0	1	0
8	1	1	1	1	1	1	1
9	1	1	1	1	0	1	1
0	1	1	1	0	1	1	1





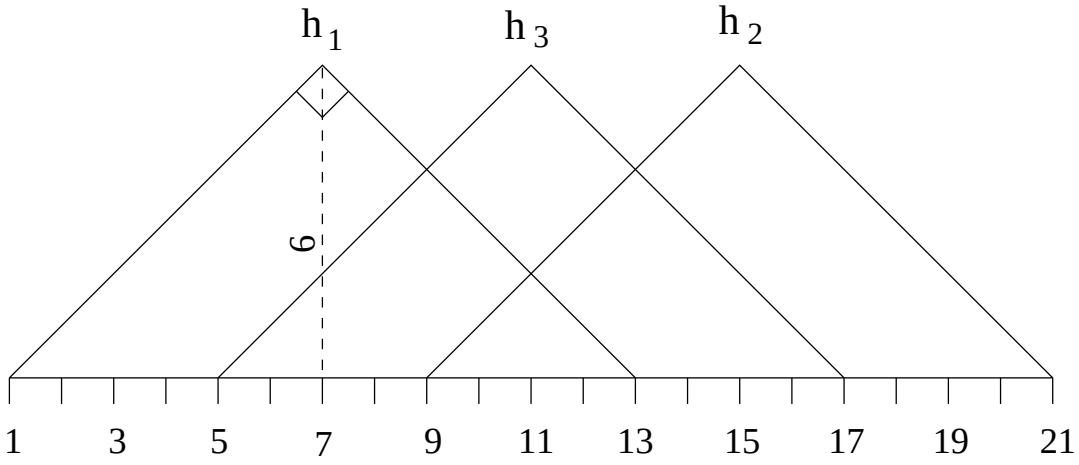
- The data for the example are generated by a malfunctioning calculator.
- Each of the seven lights has probability 0.1 of being in the wrong state independently.
- The training set contains 200 samples generated according to the specified distribution.
- A tree structured classifier is applied.
  - The set of questions $\mathcal{Q}$ contains:
    Is $x_{.j} = 0$?, $j = 1, 2, ..., 7$.
  - The twoing rule is used in splitting.
  - The pruning cross-validation method is used to choose the right sized tree.
- Classification performance:
  - The error rate estimated by using a test set of size 5000 is 0.30.
  - The error rate estimated by cross-validation using the training set is 0.30.
  - The resubstitution estimate of the error rate is 0.29.
  - The Bayes error rate is 0.26.
  - There is little room for improvement over the tree classifier.



- Accidently, every digit occupies one leaf node.
  - In general, one class may occupy any number of leaf nodes and occasionally no leaf node.
- $X_{\cdot 6}$ and $X_{\cdot 7}$ are never used.

Waveform Example (CART)

- Three functions $h_1(\tau)$, $h_2(\tau)$, $h_3(\tau)$ are shifted versions of each other, as shown in the figure.



- Each h_j is specified by the equal-lateral right triangle function. Its values at integers $\tau = 1 \sim 21$ are measured.

- The three classes of waveforms are random convex combinations of two of these waveforms plus independent Gaussian noise. Each sample is a 21 dimensional vector containing the values of the random waveforms measured at $\tau = 1, 2, ..., 21$.
  - To generate a sample in class 1, a random number u uniformly distributed in $[0, 1]$ and 21 random numbers ϵ_1, ϵ_2, ..., ϵ_{21} normally distributed with mean zero and variance 1 are generated.

    $$x_{.j} = uh_1(j) + (1 - u)h_2(j) + \epsilon_j, \quad j = 1, ..., 21.$$

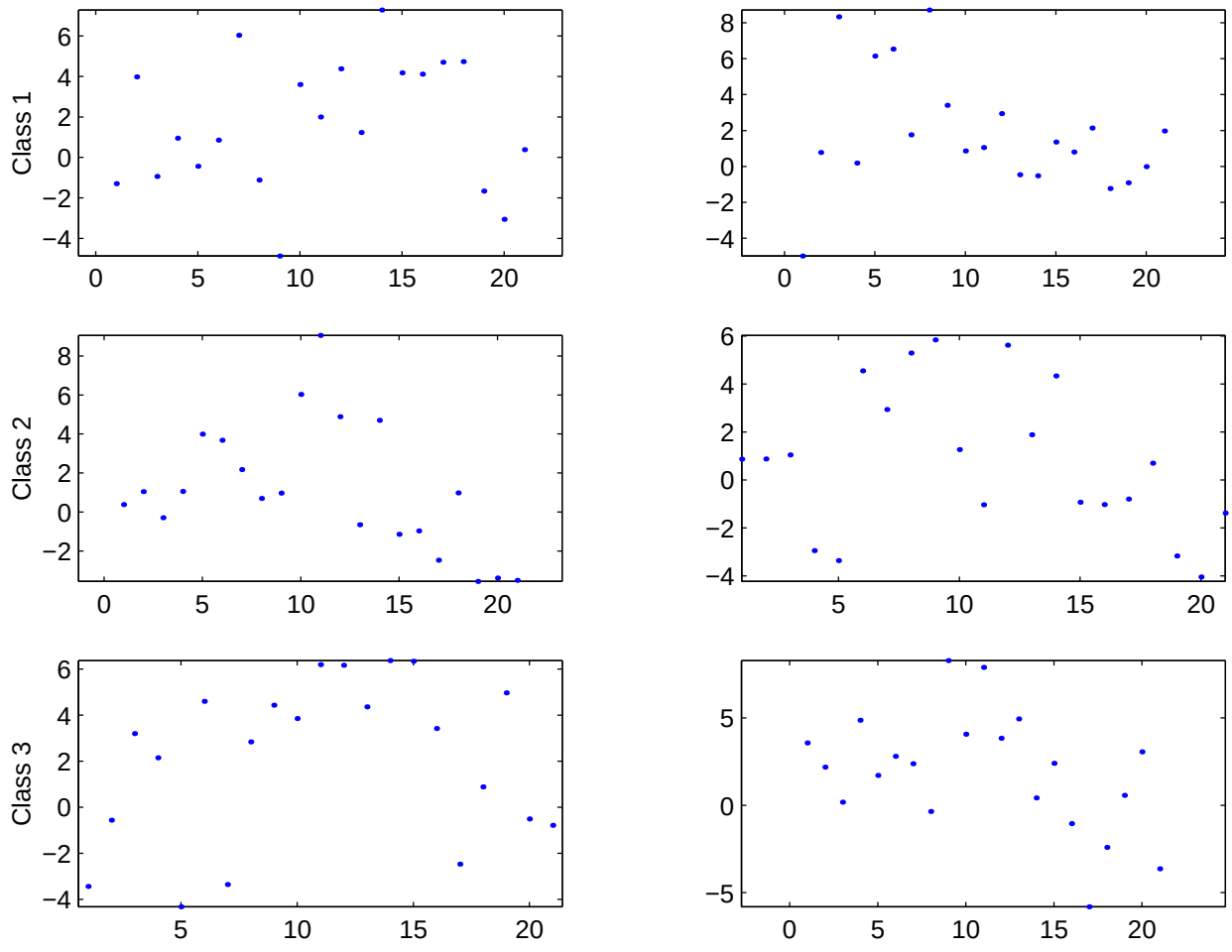
  - To generate a sample in class 2, repeat the above process to generate a random number u and 21 random numbers ϵ_1, ..., ϵ_{21} and set

    $$x_{.j} = uh_1(j) + (1 - u)h_3(j) + \epsilon_j, \quad j = 1, ..., 21.$$

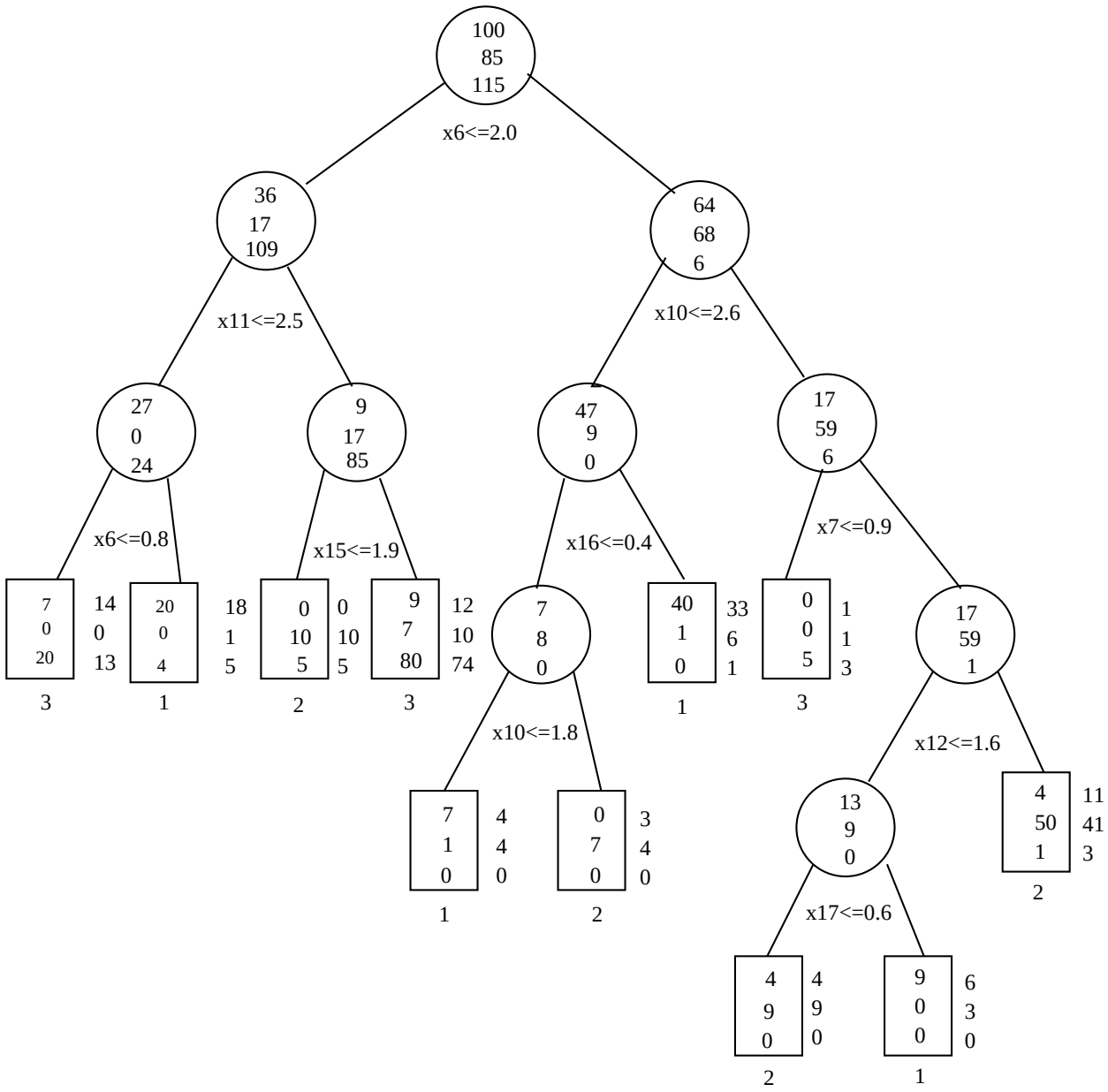
  - Class 3 vectors are generated by

    $$x_{.j} = uh_2(j) + (1 - u)h_3(j) + \epsilon_j, \quad j = 1, ..., 21.$$

- Example random waveforms are shown below.


Class 1
Class 2
Class 3

- 300 random samples are generated using prior probabilities $(\frac{1}{3}, \frac{1}{3}, \frac{1}{3})$ for training.
- Construction of the tree:
  - The set of questions: $\{\text{Is } \ x_{.j} \leq c?\}$ for c ranging over all real numbers and $j = 1, ..., 21$.
  - Gini index is used for measuring goodness of split.
  - The final tree is selected by pruning and cross-validation.
- Results:
  - The cross-validation estimate of misclassification rate is 0.29.
  - The misclassification rate on a separate test set of size 5000 is 0.28.
  - The Bayes classification rule can be derived. Applying this rule to the test set yields a misclassification rate of 0.14.

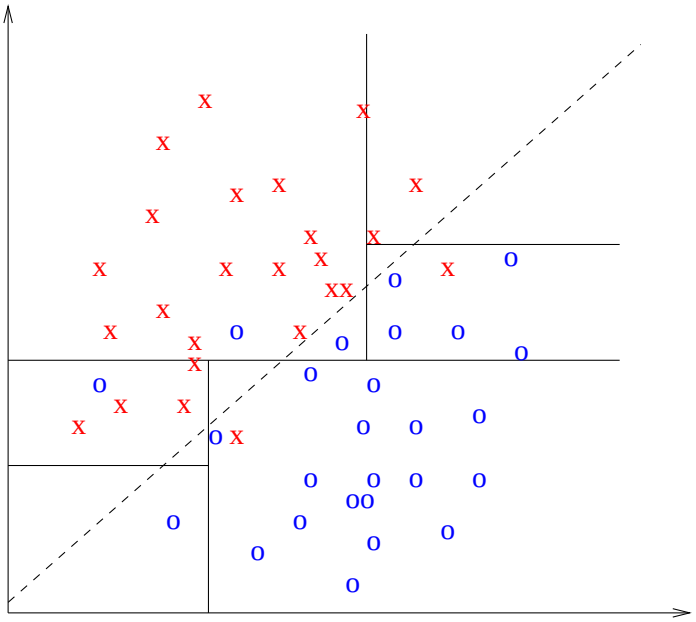

100
85
115
x6<=2.0
36
17
109
64
68
6
x11<=2.5
x10<=2.6
27
0
24
9
17
85
47
9
0
17
59
6
x6<=0.8
x15<=1.9
x16<=0.4
x7<=0.9
7
0
20
14
0
13
3
20
0
4
18
1
5
1
0
10
5
0
10
5
2
9
7
80
12
10
74
3
7
8
0
40
1
0
33
6
1
1
0
0
5
1
1
3
3
17
59
1
x10<=1.8
x12<=1.6
7
1
0
4
4
0
1
0
7
0
3
4
0
2
13
9
0
4
50
1
11
41
3
2
x17<=0.6
4
9
0
4
9
0
2
9
0
0
6
3
0
1

Advantages of the Tree-Structured Approach

- Handles both categorical and ordered variables in a simple and natural way.
- Automatic stepwise variable selection and complexity reduction.
- It provides an estimate of the misclassification rate for a query sample.
- It is invariant under all monotone transformations of individual ordered variables.
- Robust to outliers and misclassified points in the training set.
- Easy to interpret.

Variable Combinations

- Splits perpendicular to the coordinate axes are inefficient in certain cases.



- Use linear combinations of variables:

$$\text{Is } \sum a_j x_{.j} \leq c?$$

  - The amount of computation is increased significantly.
  - Price to pay: model complexity increases.

Missing Values

- Certain variables are missing in some training samples.
  - Often occurs in gene-expression microarray data.
  - Suppose each variable has 5% chance being missing independently. Then for a training sample with 50 variables, the probability of missing some variables is as high as 92.3%.
- A query sample to be classified may have missing variables.
- Find *surrogate splits*.
  - Suppose the best split for node t is s which involves a question on X_m. Find another split s' on a variable X_j, $j \neq m$, which is most similar to s in a certain sense. Similarly, the second best surrogate split, the third, and so on, can be found.