# Decision Tree Learning

## Decision Tree

- Examples of systems that learn decision trees: c4.5, CLS, IDR, ASSISTANT, ID5, CART, ID3.
- Suitable problems:
  - Instances are described by attribute-value couples
  - The target function has discrete values
  - Disjunctive descriptions of concepts may be required
  - The training set may contain errors (noise)
  - The training set may contain incomplete data



## c4.5

- c4.5 [Qui93b,Qui96]: evolution of ID, also by J. R. Quinlan
- Inspired to one of the first decision tree learning system, CLS (Concept Learning Systems) by E.B. Hunt
- Benchmark for many learning systems



## Example

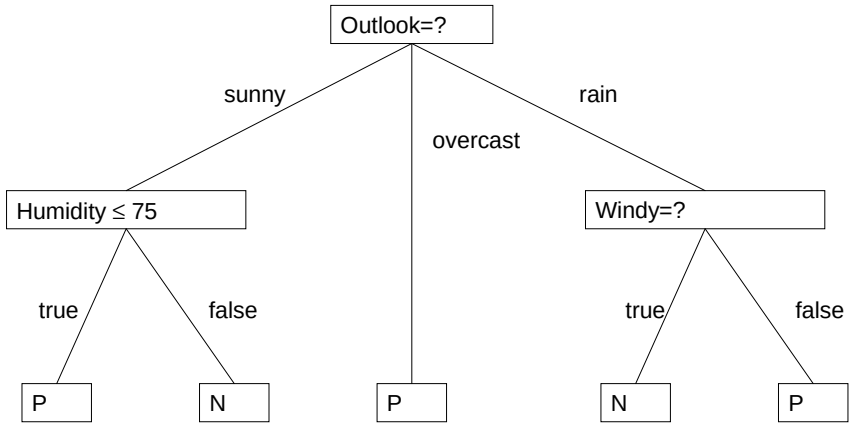
- Instances: Saturday mornings
- Classes:
  - Good day for playing tennis
  - Bad day for playing tennis
- Attributes
  - outlook, discrete, values={sunny,overcast,rain}
  - temperature, continuous
  - humidity, continuous
  - windy, discrete, values={true, false}

# Training set

| No | Outlook | Temp (°F) | Humid (%) | Windy | Class |
|---|---|---|---|---|---|
| D1 | sunny | 75 | 70 | T | P |
| D2 | sunny | 80 | 90 | T | N |
| D3 | sunny | 85 | 85 | F | N |
| D4 | sunny | 72 | 95 | F | N |
| D5 | sunny | 69 | 70 | F | P |
| D6 | overcast | 72 | 90 | T | P |
| D7 | overcast | 83 | 78 | F | P |
| D8 | overcast | 64 | 65 | T | P |
| D9 | overcast | 81 | 75 | F | P |
| D10 | rain | 71 | 80 | T | N |
| D11 | rain | 65 | 70 | T | N |
| D12 | rain | 75 | 80 | F | P |
| D13 | rain | 68 | 80 | F | P |
| D14 | rain | 70 | 96 | F | P |



# Decision Tree






# Decision Tree

Outlook=sunny
| Humidity ≤ 75: P
| Humidity > 75: N
Outlook=overcast: P
Outlook=rain
| Windy=True: N
| Windy=False: P



# Notation

- Let T be the training set,
- Let $\{C_1,C_2,\ldots,C_k\}$ be the set of classes;

# Tree Building Algorithm

- build_tree(T) returns a tree:
  - T contains examples from the same class
    - Return a leaf with label the class
  - T contains examples from more than one class
    - T is partitioned into subsets $T_1,T_2,\ldots,T_n$ according to a test on an attribute
    - Call the algorithm recursively on the subsets:
      - $child_i$=build_tree($T_i$) for i=1,…n
    - Return a subtree with the root associated to the test and childs $child_1,\ldots,child_n$.



# Example of build_tree

T=

| No | Outlook | Temp (°F) | Humid (%) | Windy | Class |
|---|---|---|---|---|---|
| D1 | sunny | 75 | 70 | T | P |
| D2 | sunny | 80 | 90 | T | N |
| D3 | sunny | 85 | 85 | F | N |
| D4 | sunny | 72 | 95 | F | N |
| D5 | sunny | 69 | 70 | F | P |
| D6 | overcast | 72 | 90 | T | P |
| D7 | overcast | 83 | 78 | F | P |
| D8 | overcast | 64 | 65 | T | P |
| D9 | overcast | 81 | 75 | F | P |
| D10 | rain | 71 | 80 | T | N |
| D11 | rain | 65 | 70 | T | N |
| D12 | rain | 75 | 80 | F | P |
| D13 | rain | 68 | 80 | F | P |
| D14 | rain | 70 | 96 | F | P |



# Test on Outlook

- $T_{sunny}$=

| No | Outlook | Temp (°F) | Humid (%) | Windy | Class |
|---|---|---|---|---|---|
| D1 | sunny | 75 | 70 | T | P |
| D2 | sunny | 80 | 90 | T | N |
| D3 | sunny | 85 | 85 | F | N |
| D4 | sunny | 72 | 95 | F | N |
| D5 | sunny | 69 | 70 | F | P |

- $T_{overcast}$=

| No | Outlook | Temp (°F) | Humid (%) | Windy | Class |
|---|---|---|---|---|---|
| D7 | overcast | 83 | 78 | F | P |
| D8 | overcast | 64 | 65 | T | P |
| D9 | overcast | 81 | 75 | F | P |



# Test on Outlook

- $T_{rain}$=

| No | Outlook | Temp (°F) | Humid (%) | Windy | Class |
|---|---|---|---|---|---|
| D10 | rain | 71 | 80 | T | N |
| D11 | rain | 65 | 70 | T | N |
| D12 | rain | 75 | 80 | F | P |
| D13 | rain | 68 | 80 | F | P |
| D14 | rain | 70 | 96 | F | P |

# build_tree($T_{sunny}$)

- Test: Humidity ≤ 75
- $T_{sunny,\ Humidity \le 75}$

| No | Outlook | Temp (°F) | Humid (%) | Windy | Class |
|---|---|---|---|---|---|
| D1 | sunny | 75 | 70 | T | P |
| D5 | sunny | 69 | 70 | F | P |

- Leaf, P label
- $T_{sunny,\ Humidity > 75}$

| No | Outlook | Temp (°F) | Humid (%) | Windy | Class |
|---|---|---|---|---|---|
| D2 | sunny | 80 | 90 | T | N |
| D3 | sunny | 85 | 85 | F | N |
| D4 | sunny | 72 | 95 | F | N |

- Leaf, N label



# build_tree($T_{overcast}$)

- $T_{overcast}$=

| No | Outlook | Temp (°F) | Humid (%) | Windy | Class |
|---|---|---|---|---|---|
| D7 | overcast | 83 | 78 | F | P |
| D8 | overcast | 64 | 65 | T | P |
| D9 | overcast | 81 | 75 | F | P |

- Leaf, P label



# build_tree($T_{rain}$)

- Test: Windy=?
- $T_{rain,true}$=

| No | Outlook | Temp (°F) | Humid (%) | Windy | Class |
|---|---|---|---|---|---|
| D10 | rain | 71 | 80 | T | N |
| D11 | rain | 65 | 70 | T | N |

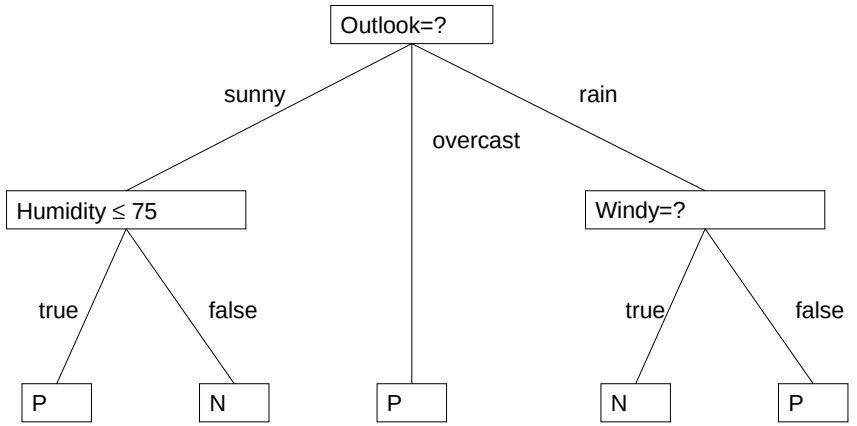
- Leaf, N label
- $T_{rain,false}$=

| No | Outlook | Temp (°F) | Humid (%) | Windy | Class |
|---|---|---|---|---|---|
| D12 | rain | 75 | 80 | F | P |
| D13 | rain | 68 | 80 | F | P |
| D14 | rain | 70 | 96 | F | P |

- Leaf, P label



# Decision Tree



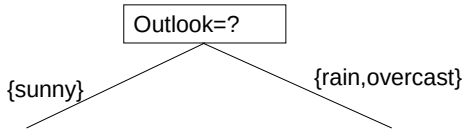
## Tests on Attributes

- Discrete attribute X with n possible values $x_1,\ldots,x_n$:
  - Equality with a constant: X=cost, 2 possible outcomes: yes, no
  - Equality test: X=?, n possible outcomes
  - Membership in a set: $X \in S$, 2 possible outcomes: yes, no
  - Membership in a set of a partition of $\{x_1,\ldots,x_n\}$: one outcome per set



## Test on Discrete Attributes

- Example of membership in a set of a partition:
  - Attribute Outlook, partition of the set of values {{sunny},{rain,overcast}}




- Continuous attribute X
  - Comparison with a threshold X≤cost, 2 possible outcomes: yes, no



## Termination Condition

- c4.5 stops
  - When an uniform set is found
  - When an empty set is found
    - A leaf is returned with label the most frequent class in the father
  - When no test is such that at least two subsets contain a minimum number of cases.
    - The minimum number of cases is a user-defined parameter assuming value 2 by default



## Building the Tree

- Search in space of all possible trees
  - Once a test is assigned to a node it is possible to backtrack
  - Infeasible
- Greedy search
  - Tests on nodes chosen irrevocably: once a test is assigned to a node it is not possible to backtrack
  - Choice on the basis of a heuristic
  - Most used heuristics
    - Entropy
    - Gini index

## Choice of the Test

- Choice of the attribute
- Discrete attributes:
  - Choice of the type of test
  - Possibly choice of the constant or partition
- Continuous attributes
  - Choice of the threshold
- Usually only the equality test X=? is used for discrete attributes
  - Only the attribute must be chosen
- Constraints that the test must satisfy: at least two among $T_1, T_2, \ldots, T_n$ must contain a minimum number of examples



## Entropy

- In information theory, entropy is a measure of the uncertainty associated with a discrete random variable.
- Random variable C with k possible values $C_1, \ldots, C_k$, entropy H(C) is given by

$$H(C) = -\sum_{j=1}^{k} P(C_j) \log P(C_j) = E[-\log_2 P(C)]$$

- Also known as Shannon entropy



## Information Theory Interpretation

- Suppose you want to transmit a series of messages made of the values of N random variables independent and identically distributed (i.i.d.)
- What is the minimum number of bits necessary to transmit these messages?
- Source coding theorem (Shannon 1948)
  - "The minimum number of bits necessary to encode the values of N i.i.d. random variables with negligible risk of information loss is N×H(X) as N tends to infinity, where H(X) is the entropy of the random variables"
  - H(X) is the minimum number of bits to encode the value of a random variable X



## Application to Decision Trees

- Random variable C: class of an instance randomly selected from a set T
- In this case

$$P(C_j) = \frac{|T_j|}{|T|}$$

- where $T_j$ is the set of examples from T that belong to class $C_j$
- We can define the entropy of T as the entropy of the random variable C

H(T)=H(C)

## Entropy

- H(T) measures the minimum number of bits necessary for encoding, without loss, a message of the form
  - "The present example, randomly selected from the training set, belongs to class $C_j$"
- H(T) is also called info(T)



## Entropy for Two Classes

- In the case of two classes, + and -, with probabilities $p_+$=P(+) and $p_-$=P(-)

$$H(T) = -p_+ \times \log_2 p_+ - p_- \times \log_2 p_-$$

- Only one variable is independent: $p_-$=1-$p_+$

$$H(T) = -p_+ \times \log_2 p_+ - (1-p_+) \times \log_2 (1-p_+)$$



## Entropy for Two Classes

- For $p_+$=0.5: $p_-$=0.5

$$H(T) = -0.5 \times \log_2 0.5 - 0.5 \times 0.5 =$$
$$-0.5 \times (-1) - 0.5 \times (-1) = 1$$

- For $p_+$=0: $p_-$=1.0

$$H(T) = -0 \times \log_2 0 - 1 \times \log_2 1 = -0 \times -\infty - 0$$

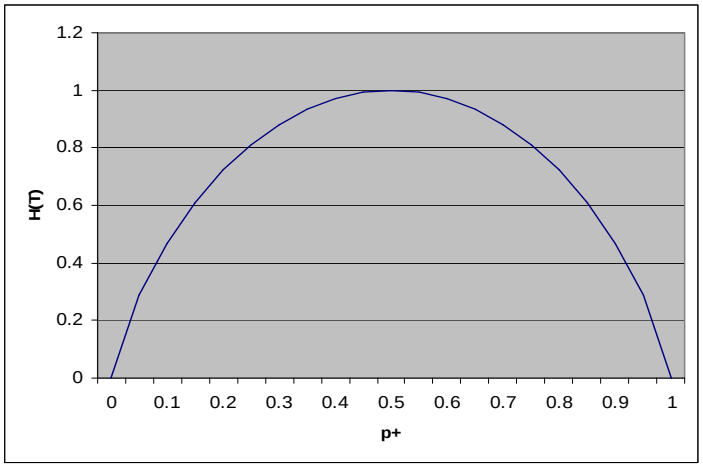
- In this case, we define

$$p_+ \times \log_2 p_+ |_{p_+=0} \stackrel{def}{=} \lim_{p_+ \to 0} p_+ \times \log_2 p_+ = 0$$

- So H(T)=0-0=0
- Similarly, if $p_+$=1 H(T)=0



## Entropy for Two Classes



## Entropy

- Entropy measures also the non-uniformity or impurity of a set:
  - It is minimal when the set is most pure, i.e., when it contains examples from only one class
  - It is maximal when the set is most impure, i.e., when it contains an equal number of examples of the two classes



## Intuition for Two Classes

- If all the examples belongs to the same class, I do not need to communicate the class of a randomly drawn example
- If the examples are uniformly distributed over classes in the training set, I need to use 1 bit on average: message 0 encodes one class, message 1 the other



## Example

- Play tennis problem
- Training set T: 14 examples, 9 positive, 5 negative
- Entropy of T
- Remember: $\log_a x=\log_{10} x/\log_{10} a=\ln x/\ln a$
- $\log_{10} 2=0.301$

$H(T)=-(9/14)\log_2(9/14)-(5/14)\log_2(5/14)=$

$=-(9/14)\log_{10}(9/14)/0.301-(5/14)\log_{10}(5/14)/0.301=$

$=-0.643*(-0.192)/0.301-0.357*(-0.447)/0.301=$

$=0.410+0.530=0.940$

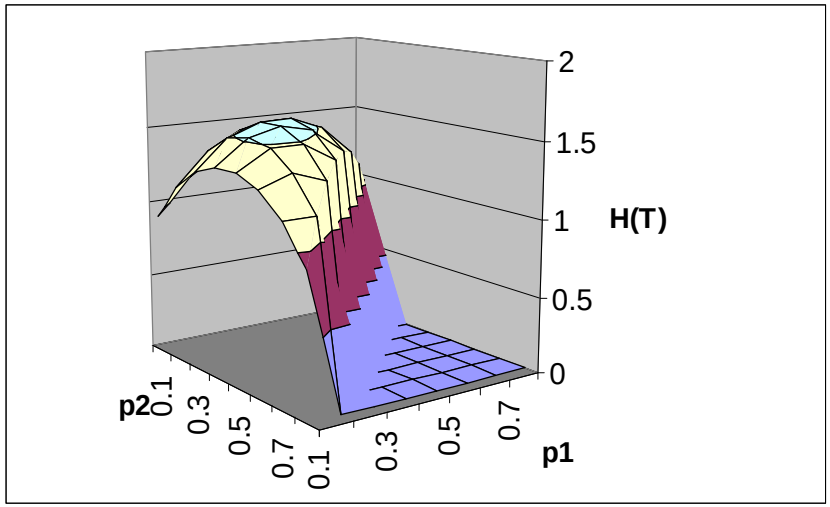


## Entropy for Three Classes

- Three classes $\{C_1,C_2,C_3\}$:
  - $p_1=P(C_1)$
  - $p_2=P(C_2)$
  - $p_3=P(C_3)$
  - Only two independent variables: $p_3=1-p_1-p_2$

$$H(T)=-p_1\times\log_2 p_1-p_2\times\log_2 p_2-$$
$$-(1-p_1-p_2)\times\log_2(1-p_1-p_2)$$

## Entropy for Three Classes




## Maximum of the Entropy

- The maximum of the entropy is obtained for a uniform distribution of the examples over classes
- For three classes: $p_1=p_2=p_2=1/3=0{,}333$
- $H(T)=-3*1/3*\log_2 1/3=\log_2 3=1{,}585$
- For k classes, we get the maximum for $P(C_j)=1/k$

$$H_{\max}(T) = -\sum_{j=1}^{k} \frac{1}{k} \times \log_2\left(\frac{1}{k}\right) = k \times \frac{1}{k} \times \log_2 k = \log_2 k$$



## c4.5 Heuristic

- Assign to a test on an attribute the value given by the decrease of entropy (**information gain**) due to the partitioning of the set of examples according to the test
- Test X with n possible outcomes
- Set of examples T is partitioned into n subsets $T_1,\ldots,T_n$
- Entropy after the partitioning: weighted average entropy in the subsets

$$H_X(T) = \sum_{i=1}^{n} \frac{|T_i|}{|T|} \times H(T_i)$$



## Information Gain

$$gain(X) = H(T) - H_X(T)$$

- It is equivalent to the number of bits necessary for encoding the class of examples that we save when we also know the result of the test X on the examples
- Entropy decreases as more and more uniform subsets are obtained

## Attribute Choice

- The attribute with the highest information gain is selected for splitting the current set of examples



## Example

- Test on the attribute Windy:
- Windy=F => $T_F$ contains 6 positive and 2 negative examples
- Windy=T => $T_T$ contains 3 positive and 3 negative examples
- T=[9,5]
- $T_F$=[6,2]
- $T_T$=[3,3]



## Example

| Windy | Play | Don't Play | Total |
|---|---|---|---|
| F | 6 | 2 | 8 |
| T | 3 | 3 | 6 |
| Total | 9 | 5 | 14 |

- $H(T_F)=-6/8*\log_2(6/8)-2/8*\log_2(2/8)=0.811$
- $H(T_T)=-3/6*\log_2(3/6)-3/6*\log_2(3/6)=1$

$gain(Windy)=H(T)-(8/14*H(T_F)+6/14*H(T_T))=$
$0.940-(8/14)*0.811-(6/14)*1=0.048$



## Example

- Test on the attribute outlook:
- Outlook=sunny=> $T_{sunny}$contains 2 pos and 3neg
- Outlook=overcast=> $T_{overcast}$contains 4 pos and 0neg
- Outlook=rain=> $T_{rain}$contains 3 pos and 2 neg
- T=[9,5], $T_{rain}$ =[3,2], $T_{sunny}$=[2,3], $T_{overcast}$=[4,0]

## Example

| Outlook | Play | Don't Play | Total |
|---|---|---|---|
| Sunny | 2 | 3 | 5 |
| Overcast | 4 | 0 | 4 |
| Rain | 3 | 2 | 5 |
| Total | 9 | 5 | 14 |

gain(Outlook)=H(T)-((5/14)*H($T_{rain}$)+(5/14)*H($T_{sunny}$)+(4/14)*H($T_{overcast}$))=

0.940-0.357*0.971-0.357*0.971-0.286*0=0.246



## Problems of Information Gain

- High tendency to favor tests with many results
- Example: test on an attribute that is a key: $|T_i|=1$, $n = |T|$

$$H(T_i)=0 \forall i \in [1,2,\ldots,n] \Rightarrow H_X(T)=0$$

- So the gain is maximum:

$$gain(X)=H(T)$$

- Example: diagnosis problem, examples=patients, attribute="Patient's name"
  - Useless attribute but has the highest gain



## Normalized Gain

- The gain is normalized with respect to the entropy of the test itself
- The random variable in this case is the value of the attribute X

$$H(X)=-\sum_{i=1}^{n}\frac{|T_i|}{|T|}\times\log_2\left(\frac{|T_i|}{|T|}\right)$$

- Gain ratio

$$gainratio(X)=\frac{gain(X)}{H(X)}$$



## Gain Ratio in the Diagnosis example

- X=patient name

$$gain(X)=H(T)\le\log_2 k$$

$$H(X)=-\sum_{i=1}^{n}\frac{1}{n}\log_2\frac{1}{n}=n\frac{1}{n}\log_2 n=\log_2 n$$

$$gainratio(X)\le\frac{\log_2 k}{\log_2 n}$$

- If n>>k$\Rightarrow$gain ratio(X) is small
- H(X) is also called splitinfo

## Example

gainratio(Windy)=gain(Windy)/H(Windy)

H(Windy)=-8/14*$\log_2$(8/14)-6/14*$\log_2$(6/14)=0.985

gainratio(Windy)=0.048/0.985=0.048

gainratio(Outlook)=gain(Outlook)/H(Outlook)

H(Outlook)=-5/14*$\log_2$(5/14)-5/14*$\log_2$(5/14) - 4/14*$\log_2$(4/14)=1.577

gainratio(Outlook)=0.246/1.577=0.156



## Gini Index

- Other impurity measure

$$gini(T) = 1 - \sum_{j=1}^{k} P(C_j)^2$$

- For two classes

$$gini(T) = 1 - p_+^{\ 2} - (1 - p_+)^2 = 2p_+ - 2p_+^2$$



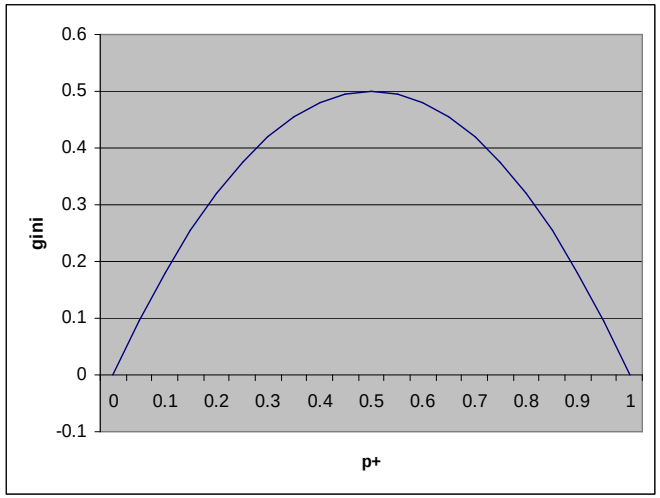
## Gini Index

- The max of Gini index is achieved for $p_+=p_-=0,5$: gini(T)=0,5
- The min is achieved for $p_+=0$ or $p_-=0$: gini(T)=0
- For three classes

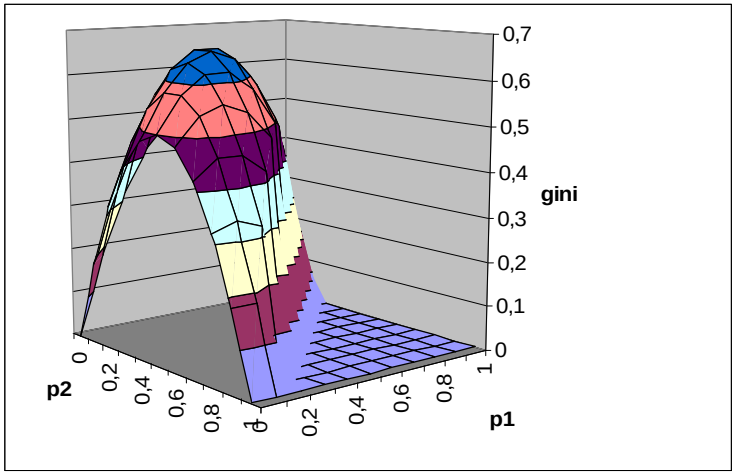
$$gini(T) = 1 - p_1^{\ 2} - p_2^{\ 2} - (1 - p_1 - p_2)^2$$



## Gini index



## Gini Index for Three Classes






## Gini Index

- In general, the Gini index has a maximum for $p_1=p_2=...p_k=1/k$

$$gini(T) = 1 - \sum_{i=1}^{k} \left(\frac{1}{k}\right)^2 = 1 - k \times \frac{1}{k^2} = 1 - \frac{1}{k}$$

- and a minimum for $p_1=1$, $p_2=...p_k=0$: gini(T)=0



## Gini index

- Gini index after the split on attribute X

$$gini_X(T) = \sum_{j=1}^{n} \frac{|T_i|}{|T|} \times gini(T_i)$$

- The attribute *X* that gives the lowest $gini_X(T)$ is chosen for the split
- Gini index is used by CART



## Tests on Continuous Attributes

- Suppose continuous attribute X assumes m values in T
- Let $<V_1,V_2,….,V_m>$ be the values ordered from the smallest to the largest
- The test $X \le V_i$ divides the m values into two groups:
- $X \le V_i$ : $\{V_1,…,V_i\}$
- $X > V_i$ : $\{V_{i+1},…,V_m\}$
- So we have m-1 candidates for the threshold: $V_1$, ..., $V_{m-1}$
- Each candidate must be evaluated

## Tests on Continuous Attributes

- Evaluation of a test:
  - Sort the examples on the basis of the attribute X
  - Count the examples
    - Let $e_{j,i}$ be the number of examples that have value $V_i$ for X and belong to class $C_j$ for j=1,…,k, i=1,…,m-1
    - Let $e_j$ be the number of examples that belong to class $C_j$ for k=1,…,k
  - Each test has two possible outcomes: yes or no



## Tests on Continuous Attributes

- $d_{j,yes}$=examples of class j in branch $X \leq V_i$
- $d_{j,no}$=examples of class j in branch $X > V_i$
- For j=1 to k $d_{j,yes}=0$ $d_{j,no}=e_j$
- For i=1 to m-1
  - Let the test be $X \leq V_i$
  - For j=1 to k
    - $d_{j,yes}:=d_{j,yes}+e_{j,i}$
    - $d_{j,no}:=d_{j,no}-e_{j,i}$
  - The heuristic for attribute X, threshold $V_i$ can be computed from the values $d_{1,yes},\ldots,d_{k,yes},d_{1,no},\ldots,d_{k,no}$



## Attributes with Unknown Value

- If the training set contains examples with one or more attributes unspecified
- For example

<?,72,90,T,P>

- How do we take into account this example when computing the heuristic and when splitting the example set?



## Attributes with Unknown Value

- Test evaluation:
  - Consider a discrete attribute X with n values $\{x_1,\ldots,x_n\}$
  - Let F be the set of examples of T that have X known
  - F provides us with a distribution of examples into values and class $P(x_i,C_j)$
  - We assume that the unknown values have the same distribution $P(x_i,C_j)$ with respect to the attribute and the class
  - Therefore the examples with unknown values do not alter the value of the entropy or of the Gini index

## Attributes with Unknown Value

- Entropy of the partitioning: the unknown value is considered as a value of its own
- Let $T_{n+1}$ be the set of examples of T with unknown value for X

$$H(X) = -\sum_{i=1}^{n+1} \frac{|T_i|}{|T|} \times \log_2 \left( \frac{|T_i|}{|T|} \right)$$



## Attributes with Unknown Value

- c4.5 further penalizes attributes with unknown values by multiplying the gain by the probability that the attribute is known
- If F is the set of examples of T with X known

$$gain(X) = \frac{|F|}{|T|} \times (H(F) - H_X(F))$$



## Example

- Suppose that in the example database the case D6 :
  Outlook=overcast, Temperature=72, Humidity=90, Windy=T
- is replaced by
  Outlook=?, Temperature=72, Humidity=90, Windy=T
- The frequencies of Outlook over F are

| Outlook | Play | Don't Play | Total |
|---|---|---|---|
| Sunny | 2 | 3 | 5 |
| Overcast | 3 | 0 | 3 |
| Rain | 3 | 2 | 5 |
| Total | 8 | 5 | 13 |



## Example

| Outlook | Play | Don't Play | Total |
|---|---|---|---|
| Sunny | 2 | 3 | 5 |
| Overcast | 3 | 0 | 3 |
| Rain | 3 | 2 | 5 |
| Total | 8 | 5 | 13 |

- $H(F)=-8/13*\log_2(8/13)-5/13*\log_2(5/13)=0.961$
- $H_{Outlook}(F)=5/13*(-2/5*\log_2(2/5)-3/5*\log_2(3/5))$
  $+3/13*(-3/3*\log_2(3/3)-0/3*\log_2(0/3))$
  $+5/13*(-3/5*\log2(3/5)-2/5*\log2(2/5))$
  $=0.747$
- gain(Outlook)=13/14*(0.961-0.747)=0.199 (slightly less than before)

## Example

- H(Outlook)=-5/14*$\log_2$(5/14)-3/14*$\log_2$(3/14)-5/14*$\log_2$(5/14)-1/14*$\log_2$(1/14)=1.809
- gainratio(Outlook)=0.199/1.809=0.110



## Partitioning

- In which subset of a node with test on X do we put an example with X unknown?
- Partitioning is generalized in a probabilistic sense:
  - Each example of T is assigned a weight, initially 1
  - Each example is inserted in every subset $T_i$ with a weight
- If we partition on discrete attribute X
  - If example e, with weight w in T, has $X=x_i$, e is put in $T_i$ with weight w and in $T_j$ with $j \neq i$ with weight 0
  - If example e, with weight w in T, has X=?, e is put in $T_i$ with weight $w \times P(x_i)$
  - $P(x_i)$ can be estimated by relative frequency in F



## Example

- Partitioning according to Outlook
- No problem for the 13 cases of F
- Example D6 is assigned to sets $T_{sunny}$, $T_{overcast}$ and $T_{rain}$ with weights 5/13, 3/13 e 5/13 respectively



## Example

- Consider $T_{sunny}$

| No | Outlook | Temp | Humidity | Windy | Class | Weight |
|---|---|---|---|---|---|---|
| D1 | sunny | 75 | 70 | T | P | 1 |
| D2 | sunny | 80 | 90 | T | N | 1 |
| D3 | sunny | 85 | 85 | F | N | 1 |
| D4 | sunny | 72 | 95 | F | N | 1 |
| D5 | sunny | 69 | 70 | F | P | 1 |
| D6 | ? | 72 | 90 | T | P | 5/13 |

- If $T_{sunny}$ is partitioned on Humidity with threshold 75, we get the following distribution:
  - Humidity $\leq$ 75 2 class P, 0 class N
  - Humidity > 75 5/13 class P, 3 class N

## Example

- The second subset still contains examples from two classes but no test produces two subsets with at least two examples=> stop

Outlook=sunny

| Humidity ≤ 75: P (2.0)

| Humidity > 75: N (3.4/0.4)

Outlook=overcast: P (3.2)

Outlook=rain

| Windy=True: N (2.4/0.4)

| Wndy=False: P (3.0)



## Output Interpretation

Numbers (A/B) associated to leaves

- A=total weight of examples associated to the leaf
- B=total weight of examples associated to the leaf that are misclassified
- For example

N (3.4/0.4)

- Means that 3.4 cases belong to the leaf of which 0.4 do not belong to class N



## Classification of Unseen Cases with All Values Known

- An unseen (new) case e with all attributes known is classified by
  - Traversing the tree from the root by following the branches that correspond to the values of the case
  - Suppose that the leaf

Pos (A/B)

  - is reached, then e is classified as Pos with probability (A-B)/A and Neg with probability B/A
  - In the case of more than two classes, the distribution of examples into classes at the leaf is used



## Classification of Unseen Cases with Unkown Values

- If e has X unknown
- e is associated with a weight w, initially set to 1
- When traversing the tree, if the attribute at the current node is unknown in e we explore all subtrees.
- Subtree $T_i$ is explored by assigning e to $T_i$ with weight $w \times P(x_j)$
- $P(x_j)$ is estimated by considering by relative frequency over T
- The information about relative frequencies are stored in the leaves that are descendants of T

## Classification of Unseen Cases with Unkown Values

- In the end more than one leaf will be reached
- Let L be the set of leaves that are reached
- Let $w_l$ be the weight of e that reaches l
- Let $P(C_j|l)$ be the relative frequency of examples in l belonging to class $C_j$
- Then

$$P(C_j \mid e) = \sum_{l \in L} w_l \times P(C_j \mid l)$$



## Example

- Unseen example e

  Outlook=sunny, Temperature=70, Humidity=?, Windy=F
- Outlook=sunny => first subtree
- Humidity=?=> we cannot determine whether Humidity ≤ 75
- We follow both branches with weights
  - Branch Humidity ≤ 75: w=2.0/5.4=0.370
  - Branch Humidity > 75: w=3.4/5.4=0.630



## Example

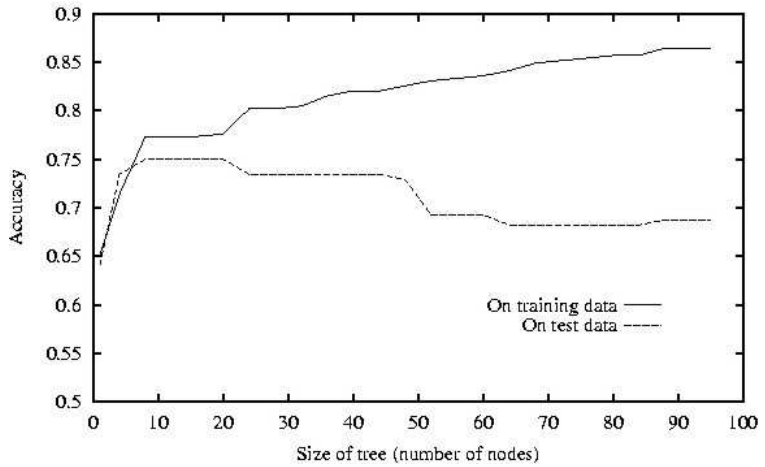
- Leaf humidity ≤ 75:
  - P(P|l)=100%
  - P(N|l)=0%
- Leaf humidity > 75
  - P(N|l)= 3/3.4=88%
  - P(P|l)= 0.4/3.4=12%
- Overall we get
  - P(P|e)=0.370*100%+0.630*12%=44%
  - P(N|e)=0.630*88%=56%



## Overfitting

- A hypothesis overfits the data when there exists a simpler hypothesis less accurate on the training set but more accurate on the universe
- It may happen when the training set contains errors or when the training set is small

## Overfitting






## Overfitting

- Example: consider the previous training set plus

  <sunny,80,70,strong,N>

- which should be positive but it is erroneously classified as negative
- The tree h' learned from the new training set is more complex than the tree h learned from the original training set
- However, since the new example is incorrect, h' will make more mistakes than h on U



## Approaches to Avoid Overfitting

- Stop the growth before reaching a tree that perfectly classifies the training examples (prepruning)
- Grow the tree and prune it afterwards (postpruning)



## How to Determine the Optimal Dimension of the Tree

- Using a separate example set for evaluating the tree ("training and validation set" approach)
- Using all the available examples for training but using a statistical test to decide whether to keep a node or not
- Using an explicit measure of the complexity of encoding the tree and the examples that are exceptions and choose a tree that (locally) minimizes this measure (minimum description length principle)

## Observation on c4.5 Search

- c4.5 performs a hill-climbing search in the space of possible trees from the simplest hypothesis towards more and more complex hypothesis
- The space of all possible trees is equivalent to the powerset of the set of examples so the target concept is surely included in the search space
- c4.5 mantains only a single hypothesis during the search and does not perform backtracking, so it may find a solution only locally optimal



## Observations on c4.5 Search

- c4.5 uses all the training examples in each step for deciding how to refine the tree differently from other methods that make decisions in an incremental way on the basis of individual examples. As a result, c4.5 is less sensible to errors in single examples.



## Useful Links

- C4.5

  http://www.rulequest.com/Personal/

  C source code
- Weka: open source suite of machine learning algorithms written in Java

  http://www.cs.waikato.ac.nz/ml/weka/



## References

[Mit97] T. M. Mitchell, *Machine Learning*, McGraw-Hill, 1997

[Qui93b] J. R. Quinlan, *C4.5: Programs for machine learning*, Morgan Kaufmann Publishers, San Mateo, California, 1993

[Qui96] J. R. Quinlan, *Improved Use of Continuous Attributes in C4.5*, Journal of Artificial Intelligence Research, 4, pag. 77--90, 1996. ftp://ftp.cs.cmu.edu/project/jair/volume4/quinlan96a.ps

[Wit05] I.H. Witten, E. Frank, *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations*, Second Edition, Morgan Kaufmann, 2005.