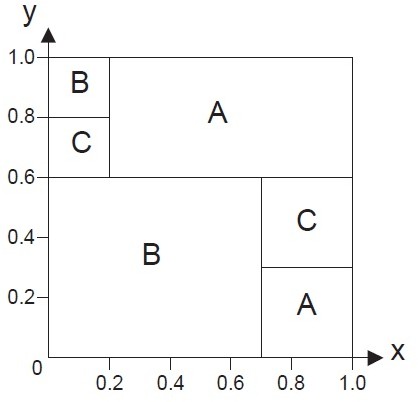
# 3

## Classification

### Decision Tree

* + 1. Consider a training set sampled uniformly from the the two-dimensional space shown in Figure 3.1.



**Figure 3.1.** 2D region

Assume that the training set size is large enough so that the probabilities can be calculated accurately based on the areas of the selected regions.

The space is divided into three classes—A, B, and C. In this exercise, you will build a decision tree from the training set.

* + - 1. Compute the entropy for the overall data.

**Answer:** For overall data, *p*(*A*) = 0*.*32 + 0*.*09 = 0*.*41, *p*(*B*) = 0*.*42 + 0*.*04 = 0*.*46, and *p*(*C*) = 0*.*04 + 0*.*09 = 0*.*13. Therefore the overall entropy is

*−*0*.*41 log2 0*.*41 *−* 0*.*46 log2 0*.*46 *−* 0*.*13 log2 0*.*13 = 1*.*4254*.*

* + - 1. Compare the entropy when the data is split at *x <*= 0*.*2, *x <*= 0*.*7, and *y <*= 0*.*6.

##### Answer:

* + - * 1. Split at *x* = 0*.*2:

For the child node *x ≤* 0*.*2, *p*(*A*) = 0, *p*(*B*) = 0*.*8, and *p*(*C*) = 0*.*2. Its entropy is *−*0*.*8 log2 0*.*8 *−* 0*.*2 log2 0*.*2 = 0*.*7219. For the child node *x >* 0*.*2, *p*(*A*) = 0*.*41*/*0*.*80 = 0*.*5125, *p*(*B*) =

0*.*3*/*0*.*8 = 0*.*3750, and *p*(*C*) = 0*.*09*/*0*.*80 = 0*.*1125. Its entropy is *−*0*.*5125 log2 0*.*5125 *−* 0*.*375 log2 0*.*375 *−* 0*.*1125 log2 0*.*1125 =

1*.*3795. Therefore, the average entropy for the children is 0*.*2 *×*

0*.*7219 + 0*.*8 *×* 1*.*3795 = 1*.*2480.

* + - * 1. Split at *x* = 0*.*7:

For the child node *x ≤* 0*.*7, *p*(*A*) = 0*.*2*/*0*.*7 = 0*.*2857, *p*(*B*) = 0*.*46*/*0*.*7 = 0*.*6571, and *p*(*C*) = 0*.*04*/*0*.*7 = 0*.*0571. Its entropy is *−*0*.*2857 log2 0*.*2857*−*0*.*6571 log2 0*.*6571*−*0*.*0571 log2 0*.*0571 =

1*.*1503. For the child node *x >* 0*.*7, *p*(*A*) = 0*.*7, *p*(*B*) = 0, and

*p*(*C*) = 0*.*3. Its entropy is *−*0*.*7 log2 0*.*7 *−* 0*.*3 log2 0*.*3 = 0*.*8813. Therefore the average entropy for the children is 0*.*7 *×* 1*.*1503 + 0*.*3 *×* 0*.*8813 = 1*.*0696.

* + - * 1. Split at *y* = 0*.*6.

For the child node *y ≤* 0*.*6, *p*(*A*) = 0*.*09*/*0*.*6 = 0*.*15, *p*(*B*) = 0*.*42*/*0*.*6 = 0*.*7, and *p*(*C*) = 0*.*09*/*0*.*6 = 0*.*15. Its entropy is

*−*0*.*15 log2 0*.*15 *−* 0*.*7 log2 0*.*7 *−* 0*.*15 log2 0*.*15 = 1*.*1813. For the

child node *y >* 0*.*6, *p*(*A*) = 0*.*32*/*0*.*4 = 0*.*8 and *p*(*B*) = *p*(*C*) =

0*.*04*/*0*.*4 = 0*.*1. Its entropy is *−*0*.*8 log2 0*.*8 *−* 0*.*1 log2 0*.*1 *−*

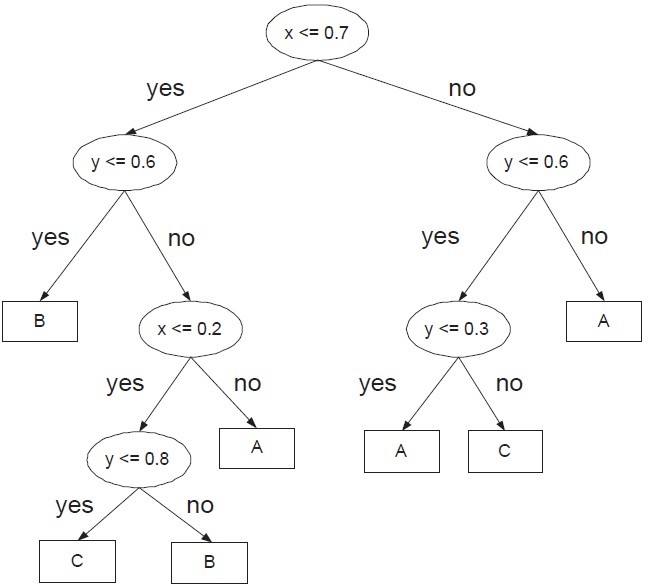
0*.*1 log2 0*.*1 = 0*.*9219. Therefore, the average entropy for the children is 0*.*6 *×* 1*.*1813 + 0*.*4 *×* 0*.*9219 = 1*.*0776.

* + - 1. Based on your answer in part (b), which attribute split condition do you think should be used as the root of the decision tree.

**Answer:** Comparing their entropy values, the split at *x* = 0*.*7 has the highest gain.

* + - 1. Draw the full decision tree for the data set.

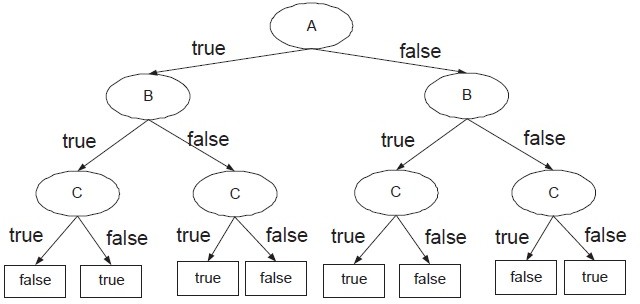
**Answer:** Full decision tree:



**Figure 3.2.** Decision tree for 2D region.

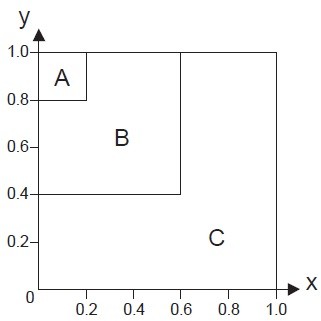
* + 1. Draw the full decision tree that perfectly classifies each of the data sets given below. There could be more than one answer to each question (you only need to draw one). You do not have to consider the impurity measure used by the decision tree algorithm. Ignore pre-pruning and post-pruning. Assume there are no noise and missing attribute values.
       1. Consider a data set with three Boolean attributes, A, B, and C, and a binary class label *y* whose value is True if the number of at- tributes with True values is even and False otherwise. For example, if A=True, B=True, C=False, then *y*=True (because there are two attributes with True values).

**Answer:** This corresponds to a parity function for 3 Boolean at- tributes.



**Figure 3.3.** Decision tree for parity function

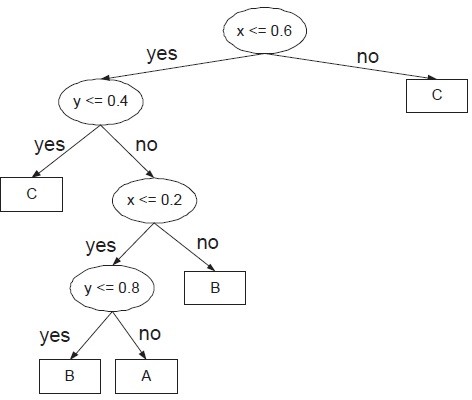
* + - 1. Consider the 2-dimensional data set shown in Figure 3.4, where A, B, and C are the class labels for the respective regions.



**Figure 3.4.** 2D region

**Answer:** The decision tree is shown in Figure 3.5.

* + 1. Show that the gini index of a node never increases after it has been split into smaller successor nodes. To simplify the problem, you can assume



**Figure 3.5.** Decision tree for 2D region

that both the splitting attribute associated with the node and the class label are binary valued.

##### Answer:

Suppose there are *N* training examples, divided into 2 classes, with *n*+

positive and *n—* negative examples. The gini index before splitting is:

*g*1 = 1 *−*

*n*+ 2

*N*

*n—* 2

*N*

(3.1)

After splitting on a binary attribute *X*, let *n*1+ and *n*1*—* be the number of positive and negative examples associated with the left child whereas *n*2+ and *n*2*—* be the number of positive and negative examples associated with the right child. Furthermore,

*−*

|  |  |  |
| --- | --- | --- |
| *n*1  *n*2 *n*+ | =  =  = | *n*1+ + *n*1*—*  *n*2+ + *n*2*— n*1+ + *n*2+ |
| *n—* | = | *n*1*—* + *n*2*—* |
| *n*1 + *n*2 | = | *N* |

The gini index after splitting is given as follows:

*g*2 =

*n*1

*N* 1 *−*

*n*1+ 2

*n*1

*n*1*—* 2

*n*1

+ *n*2 1

*N*

*−*

*n*2+ 2

*n*2

*n*2*—* 2

*n*2

*n*1 + *n*2 *n*2

*−*

*−*

*n*2*— n*2 *n*2

= 1+

*−*

*— −*

1 2+ 2*—*

*−*

*N*

*n*2

*—* 1+

= 1

*n*1*N*

*n*1*N*

*n*2

*—* 1*—*

*n*1*N*

*n*1*N*

*n*2

2+

* *n*2*N*

*n*2*N*

*n*2

*—* 2*—*

*n*2*N*

*n*2*N*

(3.2)

Comparing Equations (3.1) and (3.2), we can prove that gini index never increases after splitting (i.e., *g*1 *≥ g*2) by showing that:

or,

+

*≤*

*n*+ 2

*N*

*n—* 2

*N*

2

1+

*n*

*n*1*N*

2

+ 2+

*n*

*n*2*N*

*n*2

+ 1*—*

*n*1*N*

*n*2

+ 2*—*

*n*2*N*

*n*2

+

*N ≤*

*n*2

2 2

1+ + 2+

*n*

*n*

*n*1 *n*2

*n*2 *n*2

and (3.3)

*—*  1*—* + 2*—*

*≤*

(3.4)

*N n*1 *n*2

To prove the inequality given in (3.3):

*n*2 *n*2 *n*2

*n*2 *n*2

(*n*1+ + *n*2+)2

1+ + 2+ *−*  + = 1+ + 2+ *−*

*n*1 *n*2 *N*

2+

*n*1 *n*2

*n*1 + *n*2

2

*n*

= 1+

*n*2(*n*1 + *n*2) + *n*2

*n*1(*n*1 + *n*2) *− n*1*n*2(*n*1+ + *n*2+)2

*n*1*n*2(*n*1 + *n*2)

*n*2 *n*2 + *n*2 *n*2 *−* 2*n*1*n*2*n*1+*n*2+

= 1+ 2 2+ 1

*n*1*n*2(*n*1 + *n*2)

(*n*1+*n*2 *− n*2+*n*1)2

=

*n*1*n*2(*n*1 + *n*2)

*≥* 0

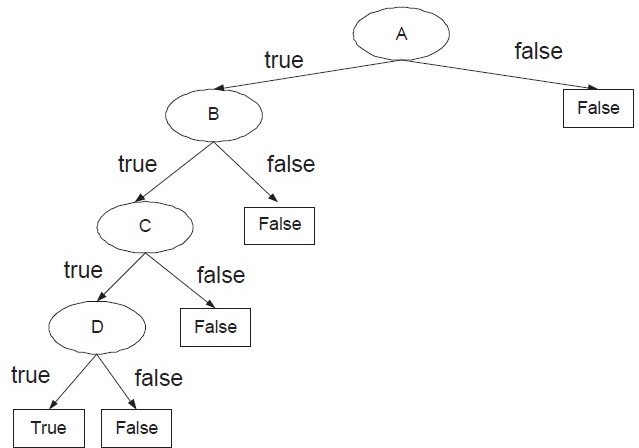
The last step follows from the fact that (*n*1+*n*2 *− n*2+*n*1)2 *≥* 0 (i.e., the square of any real numbers must be non-negative) and *n*1*n*2(*n*1+*n*2) *≥* 0. Thus, the inequality in (3.3) holds. A similar proof can be given for the

inequality in (3.4) by replacing *n*1+, *n*2+, and *n*+ with *n*1*—*, *n*2*—*, and

*n—*, respectively.

* + 1. Consider a data set that contains 4 Boolean attributes *A*, *B*, *C*, and *D*, and a Boolean class *y*. For each Boolean expression below (between the class *y* and the rest of the attributes), state whether it is possible to construct a smaller decision tree that perfectly classifies the data without generating the complete tree (i.e., the number of leave nodes is less than 16). If possible, draw the tree.
       1. *y* = *A ∧ B ∧ C ∧ D*.

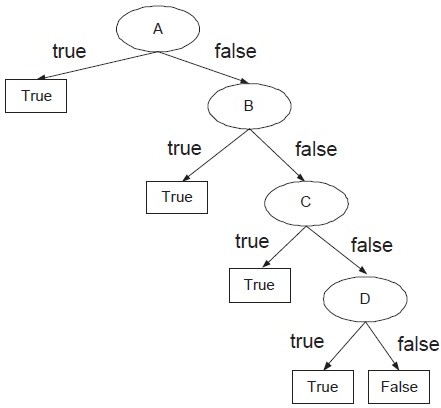
**Answer:** Yes, it is possible to construct a smaller tree.



**Figure 3.6.** Decision tree for *y* = *A ∧ B ∧ C ∧ D*.

* + - 1. *y* = *A ∨ B ∨ C ∨ D*.

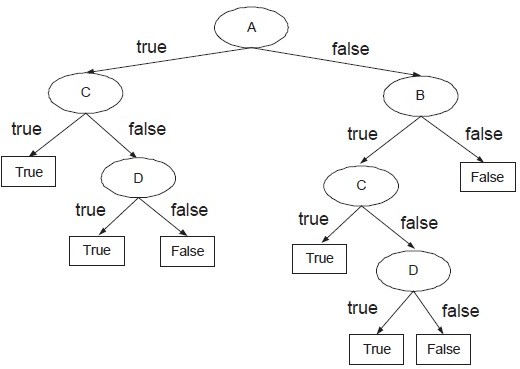
**Answer:** Yes, it is possible to construct a smaller tree.



**Figure 3.7.** Decision tree for *y* = *A ∨ B ∨ C ∨ D*.

(c) *y* = (*A ∨ B*) *∧* (*C ∨ D*).

**Answer:** Yes, it is possible to construct a smaller tree.



**Figure 3.8.** Decision tree for *y* = (*A ∨ B*) *∧* (*C ∨ D*).

* + 1. Consider the training set given below for predicting lung cancer in pa- tients based on their symptoms (chronic cough and weight loss) and other lifestyle and environmental attributes (tobacco smoking and expo- sure to radon). Draw a two-level decision tree obtained using entropy as the impurity measure. Show your steps clearly (i.e., the computation of information gain for every candidate attribute at the first and second levels of the decision tree must be shown). Compute the training error of the decision tree.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Tobacco  Smoking | Radon  Exposure | Chronic  Cough | Weight  Loss | Lung  Cancer |
| Yes | Yes | Yes | No | Yes |
| Yes | No | Yes | No | Yes |
| Yes | No | Yes | Yes | Yes |
| Yes | No | Yes | Yes | Yes |
| No | Yes | No | Yes | Yes |
| Yes | No | No | No | No |
| No | No | Yes | No | No |
| No | No | Yes | Yes | No |
| No | No | Yes | No | No |
| No | No | No | Yes | No |

##### Answer:

Before splitting: *p*(+) = *p*(*−*) = 0*.*5. Therefore, the overall entropy is

*−*0*.*5 log(0*.*5) *−* 0*.*5 log(0*.*5) = 1. The contingency tables and entropies after splitting on the attributes are:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | | Lung Cancer | | Entropy  (Child) | Entropy  Total | Information  Gain |
| Yes | No |
| Tobacco  Smoking | Yes | 4 | 1 | 0.7219 | 0.7219 | 0.2781 |
| No | 1 | 4 | 0.7219 |
| Radon  Exposure | Yes | 2 | 0 | 0 | 0.7635 | 0.2365 |
| No | 3 | 5 | 0.9544 |
| Chronic  Cough | Yes | 4 | 3 | 0.9852 | 0.9651 | 0.0349 |
| No | 1 | 2 | 0.9183 |
| Weight  Loss | Yes | 3 | 2 | 0.9710 | 0.9710 | 0.0290 |
| No | 2 | 3 | 0.9710 |

So, the attribute with highest information gain is tobacco smoking. Next, for tobacco smoking = yes, the contingency tables and entropies after splitting on the remaining attributes are:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Tobacco smoking  = yes | | Lung Cancer | | Entropy  (Child) | Entropy  Total | Information  Gain |
| Yes | No |
| Radon  Exposure | Yes | 1 | 0 | 0 | 0.6490 | 0.0729 |
| No | 3 | 1 | 0.8113 |
| Chronic  Cough | Yes | 4 | 0 | 0 | 0 | 0.7219 |
| No | 0 | 1 | 0 |
| Weight  Loss | Yes | 2 | 0 | 0 | 0.5510 | 0.1709 |
| No | 2 | 1 | 0.9183 |

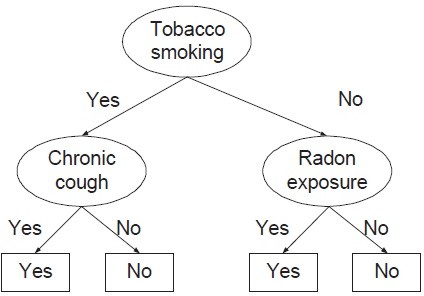
Therefore, the best attribute to split the data (at level 2) for tobacco smoking = yes is Chronic cough. If chronic cough = yes, the leaf node is labeled as lung cancer = yes. If chronic cough = no, the leaf node is labeled as lung cancer = no.

For tobacco smoking = no, the contingency tables and entropies after splitting on the remaining attributes are:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Tobacco smoking  = no | | Lung Cancer | | Entropy  (Child) | Entropy  Total | Information  Gain |
| Yes | No |
| Radon  Exposure | Yes | 1 | 0 | 0 | 0 | 0.7219 |
| No | 0 | 4 | 0 |
| Chronic  Cough | Yes | 0 | 3 | 0 | 0.4000 | 0.3219 |
| No | 1 | 1 | 1 |
| Weight  Loss | Yes | 1 | 2 | 0.9183 | 0.5510 | 0.1709 |
| No | 0 | 2 | 0 |

Therefore, the best attribute to split the data (at level 2) for tobacco smoking = no is radon exposure. If radon exposure = yes, the leaf node is labeled as lung cancer = yes. If radon exposure = no, the leaf node is labeled as lung cancer = no. The 2-level decision tree is shown below and its training error is 0.

* + 1. Show that the error rate of a decision tree never increases if one of the nodes is split into smaller successor nodes. To simplify the problem, you can assume that both the splitting attribute associated with the node and the class label are binary valued. It is sufficient to assume that the tree originally has only 1 node. After splitting, the new decision tree has 3 nodes (1 root node and 2 leave nodes). Show that the error rate of the new decision tree cannot be larger than the error rate of the initial tree.



**Figure 3.9.** Decision tree for lung cancer prediction problem.

**Answer:** Consider a node (before splitting) with the following class distribution: *n*+ + *n—* = *N* , where *n*+ and *n—* are the number of training examples that belong to the positive and negative classes, respectively. The error rate of the node is:

*b*

Error rate (before splitting)*, E*

= 1 *−* max *n*+ *, n—* (3.5)

Suppose the node is split into two children. The class distribution for the first child is *n*1+ and *n*1*—*, while the class distribution for the second child is *n*2+ and *n*2*—*, respectively. The error rates of the children are

*N*

*N*

*E* = 1 *−* max *n*1+ *, n*1*—* *, E* = 1 *−* max *n*2+ *, n*2*—*

*c*1

*n*1

*n*1

*c*2

*n*2

*n*2

Therefore, the error rate after splitting is:

Error rate (after splitting)*, Ea*

= *n*1 *E*

*N c*1

+ *n*2 *E*

*N c*2

= *n*1 1 *−* max *n*1+ *, n*1*—* + *n*2 1 *−* max *n*2+ *, n*2*—*

*N*

*n*1

*n*1

*N*

*n*2

*n*2

= *n*1 + *n*2 *−* max *n*1+ *, n*1*—* *−* max *n*2+ *, n*2*—*

*N*

*N*

*N*

*N*

*N*

= 1 *−* max *n*1+ *, n*1*—* + max *n*2+ *, n*2*—* (3.6)

*N*

*N*

*N*

*N*

To complete the proof, we need to show that

max *n*1+ *, n*1*—* + max *n*2+ *, n*2*—* *≥* max *n*+ *, n—*

*N*

*N*

*N*

*N*

*N*

*N*

Note that

max *n*1+ *, n*1*—* + max *n*2+ *, n*2*—*

*N*

*N*

*N*

*N*

= max *n*1+ + *n*2+ *, n*1*—* + *n*2*— , n*1+ + *n*2*— , n*1*—* + *n*2+

*N*

*N*

*N*

*N*

*N*

*N*

*N*

*N*

= max *n*+ *, n— , n*1+ + *n*2*— , n*1*—* + *n*2+

*N*

*N*

*N*

*N*

*N*

*N*

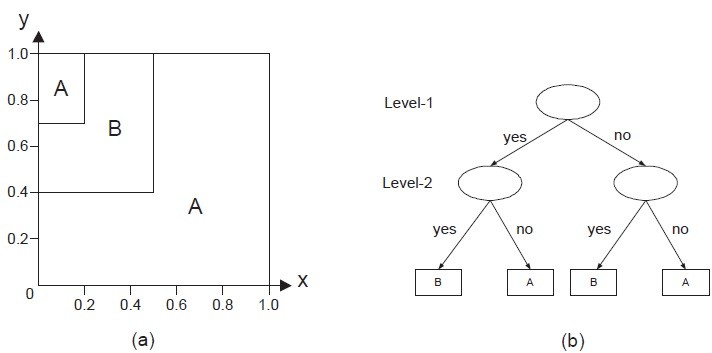
*≤* max *n*+ *, n—* (3.7)

*N*

*N*

where, on the last line, we have used the following property of the max function, that adding any number to a list can only make the maximum value larger. Thus, *Ea ≤ Eb*.

* + 1. Consider the two-dimensional data shown in Figure 3.10(a). The data consists of two classes: A and B.



**Figure 3.10.** (a) A 2-d data set, (b) a 2-level decision tree.

* + - 1. Draw a 2-level decision tree for the data (see Figure 3.10(b)). Use gini index as the splitting criterion. Assume the classifier uses a

binary split, i.e., the splitting criterion at each internal node must be specified either as *x ≤ c* or *y ≤ c*, where *c* is some constant. In other words, do not specify the splitting criteria as 0*.*5 *≤ x ≤* 1*.*0

or *x* + *y ≤* 1.

**Answer:** There are two important points to note here. First, the

probabilities associated with each class is proportional to its area in the diagram. Second, the best split position is always located at the boundary between the two classes. Thus, the candidate split positions you need to consider at level 1 of the decision tree are

*x ≤* 0*.*2, *x ≤* 0*.*5, *y ≤* 0*.*7, and *y ≤* 0*.*4.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | | Class | | Gini  (Child) | Gini  Total |
| A | B |
| *x ≤* 0*.*2 | Yes | 0.14 | 0.06 | 0.4200 | 0.3630 |
| No | 0.62 | 0.18 | 0.3488 |
| *x ≤* 0*.*50 | Yes | 0.26 | 0.24 | 0.4992 | 0.2496 |
| No | 0.5 | 0 | 0 |
| *y ≤* 0*.*4 | Yes | 0.40 | 0 | 0 | 0.2880 |
| No | 0.36 | 0.24 | 0.4800 |
| *y ≤* 0*.*7 | Yes | 0.55 | 0.15 | 0.3367 | 0.3617 |
| No | 0.21 | 0.09 | 0.4200 |

Thus, we should split at *x* = 0*.*5 because it has the lowest gini. Furthermore, for *x >* 0*.*5, notice that the entire region is classified as A, so it does not have to be split any further. For *x ≤* 0*.*5, we

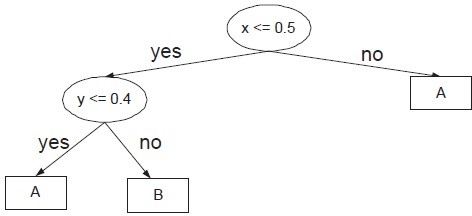
need to consider the following candidate split positions: *x ≤* 0*.*2,

*y ≤* 0*.*4, and *y ≤* 0*.*7.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | | Class | | Gini  (Child) | Gini  Total |
| A | B |
| *x ≤* 0*.*2 | Yes | 0.14 | 0.06 | 0.4200 | 0.4560 |
| No | 0.12 | 0.18 | 0.4800 |
| *y ≤* 0*.*4 | Yes | 0.20 | 0 | 0 | 0.1920 |
| No | 0.06 | 0.24 | 0.3200 |
| *y ≤* 0*.*7 | Yes | 0.20 | 0.15 | 0.4898 | 0.4869 |
| No | 0.06 | 0.09 | 0.4800 |

Clearly, the best split position is *y ≤* 0*.*4. If true, then the node is classified as A. Otherwise, it is classified as B. The resulting decision

tree is shown in Figure 3.11.



**Figure 3.11.** Decision tree for 2D problem.

* + - 1. Compute the expected error rate of your decision tree when it is applied to a test set randomly sampled from the same 2-d space. **Answer:** The expected error rate of the tree is equal to the area of the upper left-hand corner box labeled as A, which is equal to 0.06 or 6%.
    1. Consider the decision trees shown in Figures 3.12(a) and (b). For each approach described below, you need to compute the generalization er- rors for both trees and decide which tree is better. The training and validation data sets are shown in Figures 3.12(c) and (d), respectively.
       1. Optimistic approach (assumes generalization error is given by the training error).

**Answer:** Error rates for trees A and B and 10% and 20%, respec- tively. So tree A is better.

* + - 1. Pessimistic approach using the upper bound on generalization error with *α* = 0*.*25 (or *Z*1*—α/*2 = 1*.*15).

**Answer:** For this approach, you need to compute the expected error of each node and then add them up. Given a leaf node with *N* training examples that reach the node and an error rate of *e*, the upper bound on generalization error is:

*z*2 *α/*2

r *e*(1*—e*)

*z*2 *α/*2

+ 1*−*

*er*(*e, N* ) *≤*

2*N* 1*—α/*2 *N* 4*N* 2

*z* 2

1 + 1*−α/*2

*e* + 1*−* + *z*

*N*

For tree A, there are five leaf nodes. The upper bound on error rates of the nodes (going from left to right) are: *e*1(0*,* 2) = *e*2(0*,* 2) =

A picture containing diagram

Description automatically generated

**Figure 3.12.** Question 8.

0*.*3980, *e*3(0*,* 1) = 0*.*5694, *e*4(0*,* 2) = 0*.*3980, and *e*5(1*/*3*,* 3) = 0*.*6500.

So, the total expected error for tree A is 2 *×* 0*.*3980 + 2 *×* 0*.*3980 + 1 *×* 0*.*5694 + 2 *×* 0*.*3980 + 3 *×* 0*.*6500 = 4*.*9077 and its expected error rate is 4*.*9077*/*10 = 0*.*4908.

For tree B, there are only three leaf nodes. The upper bound on error rates of the nodes (going from left to right) are: *e*1(0*,* 2) = *e*2(0*,* 2) = 0*.*3980 and *e*3(1*/*3*,* 3) = *e*4(1*/*3*,* 3) = 0*.*6500. So, the total

expected error for tree B is 2 *×* 0*.*3980 + 2 *×* 0*.*3980 + 3 *×* 0*.*6500 + 3 *×* 0*.*6500 = 5*.*4923 and its expected error rate is 4*.*9077*/*10 = 0*.*5492.

So, tree A is better.

* + - 1. Reduced error pruning approach (generalization error is computed using the validation set shown in Figure 3.12(d)).

**Answer:** Error rates for trees A and B and 50% and 30%, respec- tively. So tree B is better.

* + - 1. minimum description length (MDL) approach. The total descrip- tion length of a tree is given by:

*Cost*(*tree, data*) = *Cost*(*tree*) + *Cost*(*data|tree*)*,*

* Each internal node of the tree is encoded by the ID of the split- ting attribute. If there are *m* attributes, the cost of encoding

each attribute is log2 *m* bits.

* Each leaf node is encoded using the ID of the class it is associ- ated with. If there are *k* classes, the cost of encoding a class is

log2 *k* bits.

* *Cost*(*tree*) is the cost of encoding all the nodes in the tree. To simplify the computation, you can assume that the total cost

of the tree is obtained by adding up the costs of encoding each internal node and each leaf node.

* *Cost*(*data|tree*) is encoded using the classification errors the tree commits on the training set. Each error is encoded by

log2 *n* bits, where *n* is the total number of training examples.

##### Answer:

Total description length for tree A is

4 *× [*log2 3*|* + 5 *× [*log2 2*|* + 1 *× [*log2 10*|* = 17 bits Total description length for tree B is

3 *× [*log2 3*|* + 4 *× [*log2 2*|* + 2 *× [*log2 10*|* = 18 bits So tree A is better.

* + 1. Draw a decision tree that perfectly classifies each of the data sets de- scribed below. There could be more than one answer to each question (you only need to draw one tree). You do not have to create a sample of the data to answer this question. Assume there are no noise nor missing values.
       1. Consider a data set with three Boolean attributes, A, B, and C, with a binary class label *y* whose value is positive if the number of attributes with True values is exceeds those with False values, and

negative otherwise. For example, if A=True, B=True, C=False, then *y*=+ (because there are more attributes with True values). Draw the full decision tree (with 8 leaf nodes) for the data. State whether it is possible to construct a smaller tree (with number of leaf nodes less than 8) that perfectly classifies the data. If possible, show the tree.

**Answer:** Full decision tree is shown in Figure 3.13

Diagram

Description automatically generated

**Figure 3.13.** Full decision tree.

It is possible to construct a smaller tree, which is shown in Figure 3.14.

Diagram

Description automatically generated

**Figure 3.14.** Pruned decision tree.

* + - 1. Consider the diagram shown in Figure 3.15, where A, B, and C are the class labels associated with each region. Assuming a sufficiently

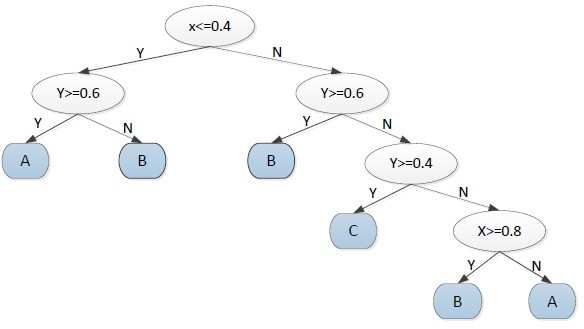
large number of training examples are sampled from each region (enough to learn the correct decision boundaries), draw a decision tree that would perfectly classify the data. You may assume the decision tree algorithm uses only binary splits (instead of multiway splits).

Chart

Description automatically generated with low confidence

**Figure 3.15.** 2-D data set.

**Answer:** The decision tree is shown in Figure 3.16



**Figure 3.16.** Decision tree for 2-D data set.

* + 1. Consider the following data set that contains 100 training examples (50 labeled as positive class while the remainder labeled as negative class).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *X* | *Y* | *Z* | No. of + Examples | No. of *−* Examples |
| 1 | 1 | 1 | 5 | 0 |
| 1 | 1 | 0 | 0 | 20 |
| 1 | 0 | 1 | 20 | 0 |
| 1 | 0 | 0 | 0 | 5 |
| 0 | 1 | 1 | 10 | 0 |
| 0 | 1 | 0 | 15 | 0 |
| 0 | 0 | 1 | 0 | 10 |
| 0 | 0 | 0 | 0 | 15 |

* + - 1. Build a *two-level* decision tree using gini index as the criterion for splitting. You need to show your computations for each candidate splitting attribute at each level clearly to obtain full credit. What is the overall training error rate of the induced tree? Note: we consider a tree with only 1 internal node and two leaf nodes as a *one-level* decision tree.

##### Answer:

For level 1,

Before splitting: *Gini* = 1*−*( 5+20+10+15 )2+( 20+5+10+15 )2 =

0*.*5

5+20+10+15+20+5+10+15

5+20+10+15+20+5+10+15

If we split the node on attribute *X*,

|  |  |  |
| --- | --- | --- |
|  | *N*1 | *N*2 |
| + | 25 | 25 |
| *−* | 25 | 25 |

*Gini*(*N* ) = 1 *−* ( 25 )2 *−* ( 25 )2 = 0*.*5

1 50 50

*Gini*(*N*2) = 0*.*5

50 50

*Gini*(*childern*) = 100 *×* 0*.*5 + 100 *×* 0*.*5 = 0*.*5

If we split the node on attribute *Y* ,

|  |  |  |
| --- | --- | --- |
|  | *N*1 | *N*2 |
| + | 30 | 20 |
| *−* | 20 | 30 |

*Gini*(*N* ) = 1 *−* ( 30 )2 *−* ( 20 )2 = 0*.*48

1

50

50

*Gini*(*N* ) = 1 *−* ( 20 )2 *−* ( 30 )2 = 0*.*48

2

50

50

50 50

*Gini*(*childern*) = 100 *×* 0*.*48 + 100 *×* 0*.*48 = 0*.*48

If we split the node on attribute *Z*,

|  |  |  |
| --- | --- | --- |
|  | *N*1 | *N*2 |
| + | 35 | 15 |
| *−* | 10 | 40 |

*Gini*(*N* ) = 1 *−* ( 35 )2 *−* ( 10 )2 = 0*.*3457

1

45

45

*Gini*(*N* ) = 1 *−* ( 15 )2 *−* ( 40 )2 = 0*.*3967

2 55 55

45 55

*Gini*(*childern*) = 100 *×* 0*.*3457 + 100 *×* 0*.*3967 = 0*.*3737

The *Gini*(*children*) of *Z* is the smallest, hence we should split the node on *Z*. The first level tree is shown in Figure 3.17.

Diagram

Description automatically generated

**Figure 3.17.** First split for 2(a).

For level 2:

For node *N*1:

If we split the node on attribute *X*,

|  |  |  |
| --- | --- | --- |
|  | *N*1 | *N*2 |
| + | 25 | 10 |
| *−* | 0 | 10 |

*Gini*(*N* ) = 1 *−* ( 25 )2 *−* ( 0 )2 = 0

1

25

25

*Gini*(*N* ) = 1 *−* ( 10 )2 *−* ( 10 )2 = 0*.*5

2 20 20

25 20

*Gini*(*childern*) = 45 *×* 0 + 45 *×* 0*.*5 = 0*.*222

If we split the node on attribute *Y* ,

|  |  |  |
| --- | --- | --- |
|  | *N*1 | *N*2 |
| + | 15 | 20 |
| *−* | 0 | 10 |

*Gini*(*N* ) = 1 *−* ( 15 )2 *−* ( 0 )2 = 0

1

15

15

*Gini*(*N* ) = 1 *−* ( 10 )2 *−* ( 10 )2 = 0*.*444

2 30 30

15 30

*Gini*(*childern*) = 45 *×* 0 + 45 *×* 0*.*444 = 0*.*296

The *Gini*(*children*) of *X* is the smaller, hence we should split the node on *X*.

For node *N*2:

If we split the node on attribute *X*,

|  |  |  |
| --- | --- | --- |
|  | *N*1 | *N*2 |
| + | 0 | 15 |
| *−* | 25 | 15 |

*Gini*(*N* ) = 1 *−* ( 0 )2 *−* ( 25 )2 = 0

1

25

25

*Gini*(*N* ) = 1 *−* ( 15 )2 *−* ( 15 )2 = 0*.*5

2 30 30

25 30

*Gini*(*childern*) = 55 *×* 0 + 55 *×* 0*.*5 = 0*.*273

If we split the node on attribute *Y* ,

|  |  |  |
| --- | --- | --- |
|  | *N*1 | *N*2 |
| + | 15 | 0 |
| *−* | 20 | 20 |

*Gini*(*N* ) = 1 *−* ( 15 )2 *−* ( 20 )2 = 0*.*4898

1

35

35

*Gini*(*N* ) = 1 *−* ( 0 )2 *−* ( 20 )2 = 0

2 20 20

35 20

*Gini*(*childern*) = 55 *×* 0*.*4898 + 55 *×* 0 = 0*.*3117

The *Gini*(*children*) of *X* is the smaller, hence we should split the node on *X*.

The two level decision tree is shown in Figure 3.18 The error rate is: 10+15 = 0*.*25.

100

Diagram

Description automatically generated

**Figure 3.18.** Two level tree for 2(a).

1. Use variable *X* as the first splitting attribute, then choose the best available splitting attribute at each of the two successor nodes. What is the training error rate of the induced tree?

##### Answer:

If we split on *X* first, For node *N*1:

If we split the node on attribute *Y* ,

|  |  |  |
| --- | --- | --- |
|  | *N*1 | *N*2 |
| + | 5 | 20 |
| *−* | 20 | 5 |

*Gini*(*N* ) = 1 *−* ( 5 )2 *−* ( 20 )2 = 0*.*32

1

25

25

*Gini*(*N* ) = 1 *−* ( 20 )2 *−* ( 5 )2 = 0*.*32

2 25 25

25 25

*Gini*(*childern*) = 50 *×* 0*.*32 + 50 *×* 0*.*32 = 0*.*32

If we split the node on attribute *Z*,

|  |  |  |
| --- | --- | --- |
|  | *N*1 | *N*2 |
| + | 25 | 0 |
| *−* | 0 | 25 |

*Gini*(*N* ) = 1 *−* ( 25 )2 *−* ( 0 )2 = 0

1

25

25

*Gini*(*N* ) = 1 *−* ( 0 )2 *−* ( 25 )2 = 0

2

25

25

25 25

*Gini*(*childern*) = 50 *×* 0 + 50 *×* 0 = 0

The *Gini*(*children*) of *Z* is the smaller, hence we should split the node on *Z*.

For node *N*2:

If we split the node on attribute *Y* ,

|  |  |  |
| --- | --- | --- |
|  | *N*1 | *N*2 |
| + | 25 | 0 |
| *−* | 0 | 25 |

*Gini*(*N* ) = 1 *−* ( 25 )2 *−* ( 0 )2 = 0

1

25

25

*Gini*(*N* ) = 1 *−* ( 0 )2 *−* ( 25 )2 = 0

2 25 25

25 25

*Gini*(*childern*) = 50 *×* 0 + 50 *×* 0 = 0

If we split the node on attribute *Y* ,

|  |  |  |
| --- | --- | --- |
|  | *N*1 | *N*2 |
| + | 10 | 15 |
| *−* | 10 | 15 |

*Gini*(*N* ) = 1 *−* ( 10 )2 *−* ( 10 )2 = 0*.*5

1

20

20

*Gini*(*N* ) = 1 *−* ( 15 )2 *−* ( 15 )2 = 0*.*5

2 30 30

20 30

*Gini*(*childern*) = 50 *×* 0*.*5 + 50 *×* 0*.*5 = 0*.*5

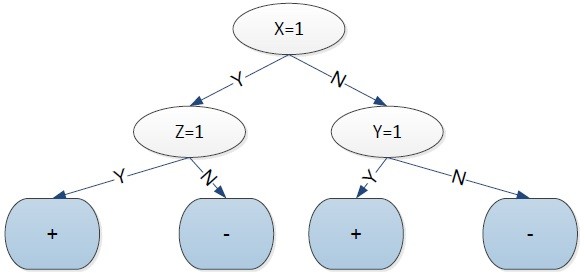
The *Gini*(*children*) of *Y* is the smaller, hence we should split the node on *Y* .

The two level decision tree is shown in Figure 3.19 The error rate is: 10+15 = 0.

100

1. Discuss the results obtained in parts (a) and (b) above. Comment on the suitability of the greedy heuristic used as the splitting at- tribute selection.

**Answer:** Greedy heuristic method cannot guarantee to produce the optimal decision tree.



**Figure 3.19.** Two level tree for 2(a).

* + 1. Consider the problem of predicting how well a baseball player will bat against a particular pitcher. The training set contains ten positive and ten negative examples. Assume there are two candidate attributes for splitting the data—ID (which is unique for every player) and Handedness (left or right). Among the left-handed players, nine of them are from the positive class and one from the negative class. On the other hand, among the right-handed players, only one of them is from the positive class, while the remaining nine are from the negative class.
       1. Compute the information gain if we use ID as the splitting attribute.

**Answer:** The entropy for the parent node is:

*Entropyparent* = *−*(0*.*5 log 0*.*5 + 0*.*5 log 0*.*5) = 1 If split using ID,

*Gain* = *Entropyparent*

20

( *Entropy*(*i*)) = 1 20 1 ( 0 log 0 1 log 1) = 1

Σ *− − × − −*

20

*i*=1

* + - 1. Repeat part (a) using Handedness as the splitting attribute.

**Answer:** If split using ID,

10

*Gain* = *Entropyparent−* 20 (*−*0*.*9 log 0*.*9*−*0*.*1 log 0*.*1)*×*2 = 1*−*0*.*469 = 0*.*531

* + - 1. Based on your answers in parts (a) and (b), which attribute will be chosen according to information gain?

**Answer:** According to information gain, we should choose ID to split the node.

* + - 1. Repeat part (a) using gain ratio (instead of information gain).

##### Answer:

*GainRATIOsplit* = *Gain* = 1 = 1 = 0*.*231

*SplitInfo* 20*×*(*—*0*.*05 log 0*.*05) 4*.*322

* + - 1. Repeat part (b) using gain ratio (instead of information gain).

##### Answer:

*GainRATIOsplit* = *Gain* = 0*.*531 = 0*.*531 = 0*.*531

*SplitInfo* 2*×*(*—*0*.*5 log 0*.*5) 1

* + - 1. Based on your answers in parts (d) and (e), which attribute will be chosen according to gain ratio?

**Answer:** According to the gain ratio, we should choose Handedness to split the node.

* + 1. Consider the training set given below for determining whether a loan application should be approved or rejected. Draw the full decision tree obtained using entropy as the impurity measure. Show your steps clearly (i.e., the computation of information gain for every candidate attribute must be shown). Compute the training error of the decision tree.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Long-Term  Debt | Unemployed | Credit  Rating | Down Payment  *<* 20% | Class |
| No | No | Good | Yes | Approve |
| No | No | Bad | No | Approve |
| No | No | Bad | Yes | Approve |
| No | No | Bad | No | Approve |
| Yes | No | Good | No | Approve |
| No | Yes | Good | Yes | Reject |
| Yes | No | Bad | No | Reject |
| Yes | No | Bad | Yes | Reject |
| Yes | No | Bad | Yes | Reject |
| Yes | Yes | Bad | No | Reject |

##### Answer:

Before splitting: *p*(+) = *p*(*−*) = 0*.*5. Therefore, the overall entropy is

*−*0*.*5 log(0*.*5) *−* 0*.*5 log(0*.*5) = 1. The contingency tables and entropies after splitting on the attributes are:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | | Class | | Entropy  (Child) | Entropy  Total | Info  Gain |
| Reject | Approve |
| Long-Term  Debt | Yes | 4 | 1 | 0.7219 | 0.7219 | 0.2781 |
| No | 1 | 4 | 0.7219 |
| Unemployed | Yes | 2 | 0 | 0 | 0.7635 | 0.2365 |
| No | 3 | 5 | 0.9544 |
| Credit  Rating | Bad | 4 | 3 | 0.9852 | 0.9651 | 0.0349 |
| Good | 1 | 2 | 0.9183 |
| Down Payment  *<* 20% | Yes | 3 | 2 | 0.9710 | 0.9710 | 0.0290 |
| No | 2 | 3 | 0.9710 |

So, the attribute with highest information gain is long-term debt. Next, for long-term debt = yes, the contingency tables and entropies after splitting on the remaining attributes are:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Long-term debt  = yes | | Class | | Entropy  (Child) | Entropy  Total | Info  Gain |
| Reject | Approve |
| Unemployed | Yes | 1 | 0 | 0 | 0.6490 | 0.0729 |
| No | 3 | 1 | 0.8113 |
| Credit  Rating | Bad | 4 | 0 | 0 | 0 | 0.7219 |
| Good | 0 | 1 | 0 |
| Down Payment  *<* 20% | Yes | 2 | 0 | 0 | 0.5510 | 0.1709 |
| No | 2 | 1 | 0.9183 |

Therefore, the best attribute to split the data (at level 2) for long-term debt = yes is credit rating. If credit rating = bad, the leaf node is labeled as class = reject. If credit rating = good, the leaf node is labeled as class

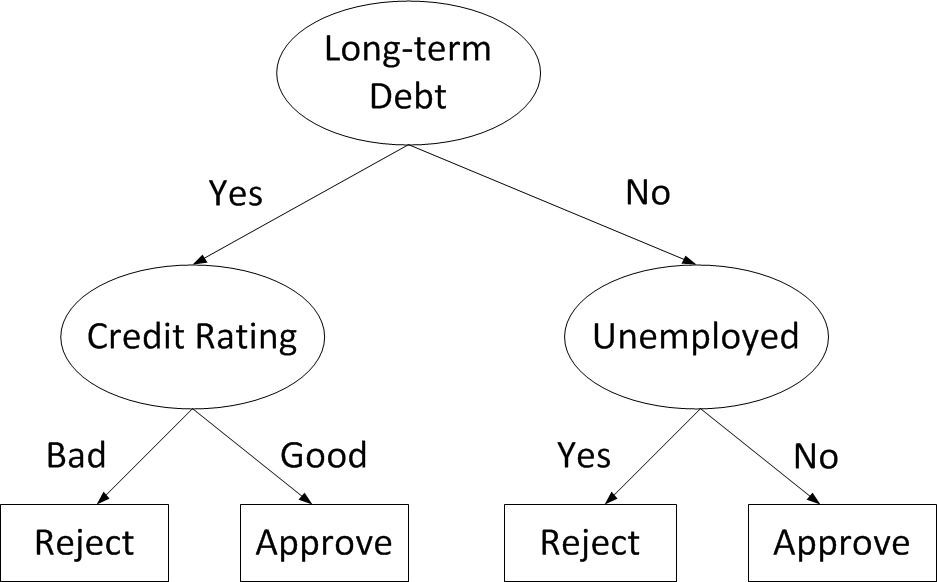
= approve.

For long-term debt = no, the contingency tables and entropies after splitting on the remaining attributes are:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Long-term debt  = no | | Class | | Entropy  (Child) | Entropy  Total | Info  Gain |
| Reject | Approve |
| Unemployed | Yes | 1 | 0 | 0 | 0 | 0.7219 |
| No | 0 | 4 | 0 |
| Credit  Rating | Bad | 0 | 3 | 0 | 0.4000 | 0.3219 |
| Good | 1 | 1 | 1 |
| Down Payment  *<* 20% | Yes | 1 | 2 | 0.9183 | 0.5510 | 0.1709 |
| No | 0 | 2 | 0 |

Therefore, the best attribute to split the data (at level 2) for long-term debt = no is unemployed. If unemployed = yes, the leaf node is labeled as class = reject. If unemployed = no, the leaf node is labeled as class

= approve. The decision tree is shown below and its training error is 0.



**Figure 3.20.** Decision tree for loan approval problem.

* + 1. This question examines property of the entropy measure.
       1. Show that the entropy measure *−p*(*x*) log *p*(*x*) is non-negative.

**Answer:** Since 0 *≤ p*(*x*) *≤* 1, hence log *p*(*x*) *≤* 0. Hence, *−p*(*x*) log *p*(*x*) *≥*

0.

* + - 1. Consider a pair of binary variables, *X* and *Y* . Suppose we need to estimate their joint probability distribution, *P* (*X, Y* ) as shown in the following 2 *×* 2 table:

|  |  |  |
| --- | --- | --- |
|  | *Y* = 1 | *Y* = 0 |
| *X* = 1 | *P* (*X* = 1*, Y* = 1) = *a* | *P* (*X* = 1*, Y* = 0) = *b* |
| *X* = 0 | *P* (*X* = 0*, Y* = 1) = *c* | *P* (*X* = 0*, Y* = 0) = *d* |

Find the joint probabilities (i.e., values of *a*, *b*, *c*, and *d*) that will maximize the entropy of the distribution assuming we know that *P* (*X* = 1) = *a* + *b* = 0*.*7 and *X,Y P* (*X, Y* ) = *a* + *b* + *c* + *d* = 1.

Σ

Hint: solve the constraint optimization problem where the objective

function corresponds to the total entropy of the distribution (refer to lecture 3 on how to solve a constraint optimization problem with the Lagrange multiplier method).

##### Answer:

The entropy of the distribution is:

*Entropy* = *−*(*a* log *a* + *b* log *b* + *c* log *c* + *d* log *d*)*.*

*L* = *−*(*a* log *a*+*b* log *b*+*c* log *c*+*d* log *d*)*−λ*1(*a*+*b*+*c*+*d−*1)*−λ*2(*a*+*b−*0*.*7) Take derivative with respect to *a*,*b*,*c*,*d*,*λ*1,*λ*2, we get:

*∂L* 1

*∂a* = log *a* + *ln*2 *− λ*1 *− λ*2 = 0

*∂L* 1

*∂b* = log *b* + *ln*2 *− λ*1 *− λ*2 = 0

*∂L* 1

*∂c* = log *c* + *ln*2 *− λ*1 = 0

*∂L* 1

*∂d* = log *d* + *ln*2 *− λ*1 = 0

*∂L*

*∂λ*1

= *−*(*a* + *b* + *c* + *d −* 1) = 0

*∂L*

*∂λ*2

= *−*(*a* + *b −* 0*.*7) = 0

The solutions are: *a* = *b* = 0*.*35, *c* = *d* = 0*.*15.

* + - 1. Consider a nominal attribute *X* that has three possible values, *x*1, *x*2, and *x*3. Suppose you have a decision tree classifier that can produce either a multi-way split or a binary split on attribute *X*. Show that the average entropy of the successors for node *X* in a multi-way split is always smaller than or equal to the average entropy of the successors of node *X* in a binary split. Hint: you can apply the following Gibbs inequality for a given pair of probability distributions, *p* and *q*, for the proof:

*—* Σ *pi* log *pi ≤* Σ *pi* log *qi.*

*i*

*i*

##### Answer:

Assume this is a binary classifier, and the classes are labeled as ”+” and ”*−*”. The numbers of samples in the nodes split by *xi* is denoted by *ni* in the multi-way split and in the binary split the

numbers of samples are denoted as *nr*1 and *nr*2*,*3. *N* is the total number of samples. It is obvious that *nr*1 = *n*1 and *n*2*,*3 = *n*2 + *n*3.

”*n*+” and ”*n—*” are represented as the number of samples belonging

*i* *i*

to class ”+” and class ”+” respectively in node *ni*.

For multi-way split, the entropy is:

*n*1 *n*+

*n*+ *n— n—*

1 1 1 1

*Entropy*1 = *−* ( log + log )

*N*

*n*2

* *N* (

*n*3

* *N* (

*n*1

*n*+

2

*n*2

*n*+

3

*n*3

log log

*n*1

*n*+

2

*n*2

*n*+

3

*n*3

*n*1

+ *n—*2

*n*2

+ *n—*3

*n*3

log log

*n*1

*n—*2 )

*n*2

*n—*3 )

*n*3

For binary split, the entropy is:

*n*1 *n*+

*n*+ *n— n—*

*Entropy*2 = *−*

( 1 log 1 + 1 log 1 )

*N n*1 *n*1 *n*1 *n*1

2 3

*n*2 + *n*3 *n*+ + *n*+

2 3

*−*

*n*+ + *n*+

*n—*2 + *n—*3

*n—*2 + *n—*3

(

*N n*2

+ *n*3

log

*n*2 + *n*3

+

*n*2 + *n*3

log

)

*n*2 + *n*3

If we want to compare the above two equations, we only need to compare these two equations:

*n*2 *n*+

*n*+ *n—*

*n— n*3 *n*+

*n*+ *n— n—*

*E*1 = *−*

( 2 log 2 + 2 log 2 ) *−* ( 3 log 3 + 3 log 3 )

*N n*2 *n*2 *n*2 *n*2 *N n*3

*n*3 *n*3 *n*3

and

*E*2 = *−*

*n*2 + *n*3

*n*+ + *n*+ *n*+ + *n*+ *n—*2 + *n—*3 *n—*2 + *n—*3

*N n*2 + *n*3

( 2 3 log 2 3 + log )

*n*2 + *n*3

*n*2 + *n*3

*n*2 + *n*3

Using Gibbs’ inequality, we can show that,

*n*2 *n*+

*n*+ *n—*

*n— n*3 *n*+

*n*+ *n— n—*

*E*1 = *−*

( 2 log 2 + 2 log 2 ) *−* ( 3 log 3 + 3 log 3 )

*N n*2 *n*2 *n*2 *n*2 *N n*3

*n*3 *n*3 *n*3

*n*2 *n*+

2

*≤ − N* ( *n*

2

*n*3 *n*+

3

*— N* ( *n*

3

log

log

*n*+ + *n*+ *n*2 + *n*3

*n*+ + *n*+ *n*2 + *n*3

2 3

2 3

+ *n—*2

*n*2

+ *n—*3

*n*3

log

log

*n—*2 + *n—*3 )

*n*2 + *n*3

*n—*2 + *n—*3 )

*n*2 + *n*3

*n*+ + *n*+ *n*+ + *n*+ *n—* + *n— n—* + *n—*

= *−*( 2 3 log 2 3 + 2 3 log 2 3 )

*N n*2 + *n*3 *N n*2 + *n*3

*n*2 + *n*3 *n*+ + *n*+ *n*+ + *n*+ *n—* + *n— n—* + *n—*

= *−* ( 2 3 log 2 3 + 2 3 log 2 3 )

*N*

= *E*2

*n*2 + *n*3

*n*2 + *n*3

*n*2 + *n*3

*n*2 + *n*3

Hence, *Entropy*1 *≤ Entropy*2.

* + 1. This question examines the relationship between entropy (H) and mutual information (I).
       1. Based on the following definitions:

*I*(*X, Y* ) =

Σ

*x∈X,y∈Y*

Σ

*p*(*x, y*)

*p*(*x, y*) log

*p*(*x*)*p*(*y*)

*H*(*X*) = *− p*(*x*) log *p*(*x*)

*x∈X*

Σ

*H*(*X, Y* ) = *−*

*x∈X,y∈Y*

*p*(*x, y*) log *p*(*x, y*)

Prove that *I*(*X, Y* ) = *H*(*X*) + *H*(*Y* ) *− H*(*X, Y* ).

##### Answer:

*I*(*X, Y* ) =

Σ

*x∈X,y∈Y*

Σ

=

*x∈X,y∈Y*

*p*(*x, y*)

*p*(*x, y*) log

*p*(*x*)*p*(*y*)

Σ

*p*(*x, y*) log *p*(*x, y*) *−*

*x∈X,y∈Y*

*p*(*x, y*) log *p*(*x*)

* *x∈X*Σ*,y∈Y*

*p*(*x, y*) log *p*(*y*)

= Σ *p*(*x, y*) log *p*(*x, y*) *−* Σ Σ *p*(*x, y*) log *p*(*x*)

*x∈X,y∈Y*

*x∈X*

*y∈Y*

* Σ Σ *p*(*x, y*) log *p*(*y*)

*y∈Y*

Σ

Σ

=

*x∈X,y∈Y*

Σ

*x∈X*

*p*(*x, y*) log *p*(*x, y*) *− p*(*x*) log *p*(*x*)

*x∈X*

* *p*(*y*) log *p*(*y*)

*y∈Y*

= *−H*(*X, Y* ) + *H*(*X*) + *H*(*Y* )

* + - 1. Consider a data source that generates a letter *α* from a set of al- phabets *{a, b, c, · · · , z}*. If a vowel (*a, e, i, o, u*) is four times more likely to be generated than a consonant (*b, c, d, f, · · · , z*), calculate

the entropy for *α*.

**Answer:** Let *V* be the set of vowels and *C* be the set of consonants. Based on the given information:

*P* (*α* = *α* ) = 4*p,* if *αi ∈* V;

*i*

*p,* if *αi ∈* C.

Since there are 5 vowels and 21 consonants and Σ26

*i*=1

(3.8)

*P* (*α* = *αi*) =

1:

5 *×* 4*p* + 21 *× p* = 1 =*⇒ p* = 1*/*41*.*

Therefore, entropy for the random variable *α* is

26

Σ

Entropy(*α*) = *− P* (*αi*) log2 *P* (*αi*)

*i*=1

4 4 1 1

= *−*5 *×* 41 log2 41 *−* 21 *×* 41 log2 41

= 4*.*3819 bits*.*

* + - 1. Consider a pair of binary variables, *X* and *Y* . Suppose we need to estimate their joint probability distribution, *P* (*X, Y* ) as shown in the following 2 *×* 2 table:

|  |  |  |
| --- | --- | --- |
|  | *Y* = 1 | *Y* = 0 |
| *X* = 1 | *P* (*X* = 1*, Y* = 1) = *a* | *P* (*X* = 1*, Y* = 0) = *b* |
| *X* = 0 | *P* (*X* = 0*, Y* = 1) = *c* | *P* (*X* = 0*, Y* = 0) = *d* |

Find the joint probabilities (i.e., values of *a*, *b*, *c*, and *d*) that will maximize the entropy of the distribution assuming we know that *P* (*X* = 1) = *a* + *b* = 0*.*6 and *X,Y P* (*X, Y* ) = *a* + *b* + *c* + *d* = 1.

Σ

Hint: solve the constraint optimization problem where the objective

function corresponds to the total entropy of the distribution.

**Answer:** We can pose this as the following optimization problem:

max

*a,b,c,d*

*−a* log *a − b* log *b − c* log *c − d* log *d*

*s.t. a* + *b* = 0*.*6

*a* + *b* + *c* + *d* = 1

By using the Lagrange multiplier method, define the Lagrangian as:

*L* = *−a* log *a − b* log *b − c* log *c − d* log *d − λ*(*a* + *b −* 0*.*6) *− µ*(*a* + *b* + *c* + *d −* 1)

Take the partial derivatives with respect to *a*, *b*, *c*, and *d*, and set them to zero:

*— λ − µ* = 0

|  |  |  |  |
| --- | --- | --- | --- |
|  | *∂L* |  | 1 |
| *∂a*  *∂L* | = *−* log2 *a −* | ln 2  1 |
| *∂b*  *∂L* | = *−* log2 *b −* | ln 2  1 |
| *∂c*  *∂L* | = *−* log2 *c −* | ln 2  1 |
| *∂d* | = *−* log2 *d −* | ln 2 |
| Therefore, |  |  |  |

*— λ − µ* = 0

*— µ* = 0

*— µ* = 0

*a* = *b* = 2*—*(1*/* ln 2+*λ*+*µ*)

*c* = *d* = 2*—*(1*/* ln 2+*µ*)

Since *a* + *b* = 0*.*6, this reduces to *a* = *b* = 0*.*3. Plugging this into

*a* + *b* + *c* + *d* = 1 and using *c* = *d*, this gives *c* = *d* = 0*.*2.

* + - 1. Based on your answer in part (c), calculate the mutual information between *X* and *Y* .

##### Answer:

*P* (*X* = 1) = *a* + *b* = 0*.*6*, P* (*X* = 0) = *c* + *d* = 0*.*4 *P* (*Y* = 1) = *a* + *c* = 0*.*5*, P* (*Y* = 1) = *b* + *d* = 0*.*5 *H*(*X*) = *−*0*.*6 log 0*.*6 *−* 0*.*4 log 0*.*4 = 0*.*971

*H*(*Y* ) = *−*0*.*5 log 0*.*5 *−* 0*.*5 log 0*.*5 = 1*.*000

*H*(*X, Y* ) = *−*2 *×* 0*.*3 log 0*.*3 *−* 2 *×* 0*.*2 log 0*.*2 = 1*.*971

Therefore, mutual information is *H*(*X*) + *H*(*Y* ) *− H*(*X, Y* ) = 0.

* + 1. This questions compares entropy against Gini index as impurity mea- sures for decision trees.
       1. Consider a two-class problem. Show that the entropy of a node in the decision tree is always greater than or equal to its gini index (use log2 for entropy).

**Answer:** Let *p* and (1 *− p*) be the distribution of the two classes at a given node. The entropy of the node is *E* = *−p* log2 *p −* (1 *− p*) log2(1 *− p*) whereas its gini is *G* = 1 *− p*2 *−* (1 *− p*)2. We need to show that

*E − G* = *−p* log2 *p −* (1 *− p*) log2(1 *− p*) *−* 1 + *p*2 + (1 *− p*)2 *≥* 0

First, note that the negative logarithm is a convex function. Ac- cording to Jensen inequality, for any convex function *f* (*x*):

Σ *αif* (*xi*) *≥ f* (Σ *αixi*)*,*

*i* *i*

where Σ*i αi* = 1. Thus:

*E − G* = *−p* log2 *p −* (1 *− p*) log2(1 *− p*) *−* 1 + *p*2 + (1 *− p*)2

= *−p* log2 *p −* (1 *− p*) log2(1 *− p*) *−* 2*p*(1 *− p*)

*≥ −* log2 *p*2 + (1 *− p*)2 *−* 2*p*(1 *− p*)

= *−* log2 1 *−* 2*p*(1 *− p*) *−* 2*p*(1 *− p*) (3.9)

where Jensen inequality was applied on the third line. Based on the Taylor series expansion of the logarithm function:

log(1 *− x*) = *−* Σ *x*

*∞*

*n*=1

*n*

*n*

Thus

*n*

Σ 2*p*(1 *− p*)

*∞*

*E − G ≥*

=

*n*=1 *n*Σ=2

*n −* 2*p*(1 *− p*)

*n*

*∞*

2*p*(1 *− p*)

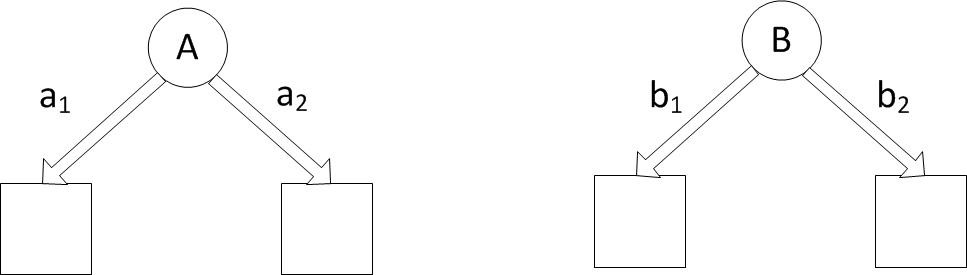
*n*

which is always non-negative since 2*p*(1 *− p*) *≥* 0 when 0 *≤ p ≤* 1.

The latter can be proved as follows. By contradiction, suppose

2*p*(1 *− p*) *<* 0 and 0 *≤ p ≤* 1. The second inequality implies *p ≥* 0 and (1*−p*) *≥* 0. But, since 2*p*(1*−p*) *<* 0, either *p <* 0 or (1*−p*) *<* 0, which is a contradiction. Thus, 2*p*(1 *− p*) must be non-negative.

* + - 1. Consider a decision tree classifier for two-class problem. Suppose we have the option of choosing either the binary attribute **A** or **B** as our splitting condition (see Figure 3.21). If attribute **A** is preferred over attribute **B** as the splitting condition according to the entropy measure, is it possible for attribute **B** to be preferred over attribute **A** according to gini index? If so, give an example; otherwise, prove that it is impossible.



**Figure 3.21.** Two options for decision tree

**Answer:** Yes, it is possible for the different measures to select dif- ferent attributes as the splitting condition. Consider the following class distribution of the child nodes for nodes A and B. Assume there are 4 positives and 6 negative examples in the training data (before splitting). The contingency tables after splitting on at- tributes *A* and *B* are:

*A* = *a*1 *A* = *a*2

+

|  |  |
| --- | --- |
| 4 | 0 |
| 3 | 3 |

*−*

*B* = *b*1 *B* = *b*2

+

|  |  |
| --- | --- |
| 3 | 1 |
| 1 | 5 |

*−*

The weighted entropy of the children after splitting on A is:

4 4 3 3

*EA*=*a*1 = *−* 7 log 7 *−* 7 log 7 = 0*.*9852

3 3 0 0

*EA*=*a*2 = *−* 3 log 3 *−* 3 log 3 = 0

*EA* = 7*/*10*EA*=*a*1 + 3*/*10*EA*=*a*2 = 0*.*6897

The weighted entropy of the children after splitting on B is:

3 3 1 1

*EB*=*b*1 = *−* 4 log 4 *−* 4 log 4 = 0*.*8113

1 1 5 5

*EB*=*b*2 = *−* 6 log 6 *−* 6 log 6 = 0*.*6500

*E* + *B* = 4*/*10*EB*=*b*1 + 6*/*10*EB*=*b*2 = 0*.*7145

Since *EA < EB*, attribute *A* will be chosen to split the node. The gini after splitting on A is:

*GA*=*a*1 = 1 *−*

*GA*=*a*2 = 1 =

4 2

3 2

7

3

3 2

0 2

7

*−*

*−*

3

= 0*.*4898

= 0

*GA* = 7*/*10*GA*=*a*2 + 3*/*10*GA*=*a*2 = 0*.*3429

The gini after splitting on B is:

1 2

3 2

*GB*=*b*1 = 1 *−*

*GB*=*b*2 = 1 =

4 *−*

1 2

*−*

6

4

5 2

6

= 0*.*3750

= 0*.*2778

*GB* = 4*/*10*GB*=*b*1 + 6*/*10*GB*=*b*2 = 0*.*3167

Since *GB < GA*, attribute *B* will be chosen to split the node.

* + 1. Consider a node in a decision tree with *n*+ positive and *n—* negative training examples. If the node is split to its *k* children, The average weighted entropy of the children is given by

Entropy(children) = *nk* Entropy(*t* )*,*

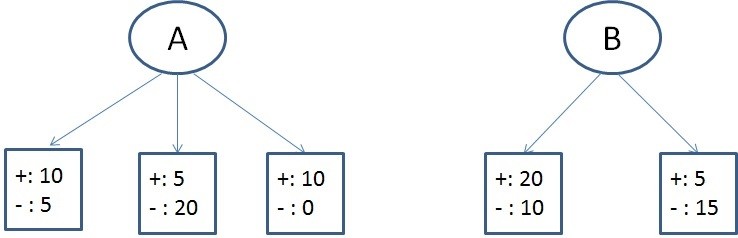
Σ

*n k*

*k*

where *nk* is the number of training examples associated with the child node *tk* and *n* = *k nk* = *n*+ + *n—*. Apply the formula to calculate the average weighted entropy for each of the candidate test conditions shown below. Based on their entropy values, which attribute, *A* or *B*, should be chosen to split the parent node?

Σ



**Answer:** For node A, the entropies of its children are

10 10 5 5

Entropy(left) = *−* 15 log2 15 *−* 15 log2 15 = 0*.*9183

5 5 20 20

Entropy(middle) = *−* 25 log2 25 *−* 25 log2 25 = 0*.*7219

10 10 0 0

Entropy(right) = *−* 10 log2 10 *−* 10 log2 10 = 0

The average weighted entropy for node A is

15 25 10

50 *×* 0*.*9183 + 50 *×* 0*.*7219 + 50 *×* 0 = 0*.*6365

Similarly, for node B, the entropies of its children are

20 20 10 10

Entropy(left) = *−* 30 log2 30 *−* 30 log2 30 = 0*.*9183

5 5 15 15

Entropy(right) = *−* 20 log2 20 *−* 20 log2 20 = 0*.*8113

The average weighted entropy for node B is

30 20

50 *×* 0*.*9183 + 50 *×* 0*.*8113 = 0*.*8755

Based on their average weighted entropy values, node *A* should be chosen because its value is smaller.

* + 1. Consider a node in a decision tree with *n*+ positive and *n—* negative training examples. If the node is split to its *k* children, The average weighted gini index of the children is given by

Gini(children) = *nk* Gini(*t* )*,*

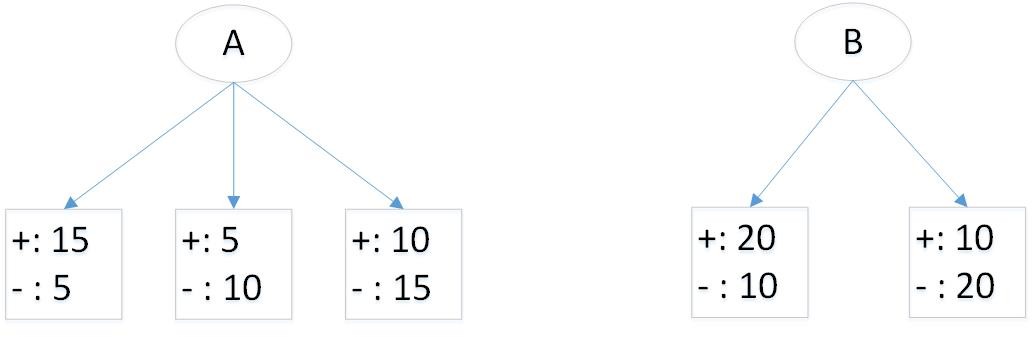
Σ

*n k*

*k*

where *nk* is the number of training examples associated with the child node *tk* and *n* = *k nk* = *n*+ + *n—*. Apply the formula to calculate the average weighted gini for each of the candidate test conditions shown below. Based on their gini values, which attribute, *A* or *B*, should be chosen to split the parent node?

Σ



##### Answer:

After splitting on attribute *A*, the gini index for each child is as follows:

Gini(left) = 1 *−* ( 15 )2 *−* ( 5 )2 = 0*.*3750

20

20

Gini(middle) = 1 *−* ( 5 )2 *−* ( 10 )2 = 0*.*4444

15

15

Gini(right) = 1 *−* ( 10 )2 *−* ( 15 )2 = 0*.*4800

25

25

Thus, the average weighted gini for the children is

20 15 25

Gini(children) = 60 *×* 0*.*3750 + 60 *×* 0*.*4444 + 60 *×* 0*.*4800 = 0*.*4361*.*

After splitting on attribute *B*, the gini index for each child is as follows: Gini(left) = 1 *−* ( 20 )2 *−* ( 10 )2 = 0*.*4444

30

30

Gini(right) = 1 *−* ( 10 )2 *−* ( 20 )2 = 0*.*4444

30

30

Thus, the average weighted gini for the children is

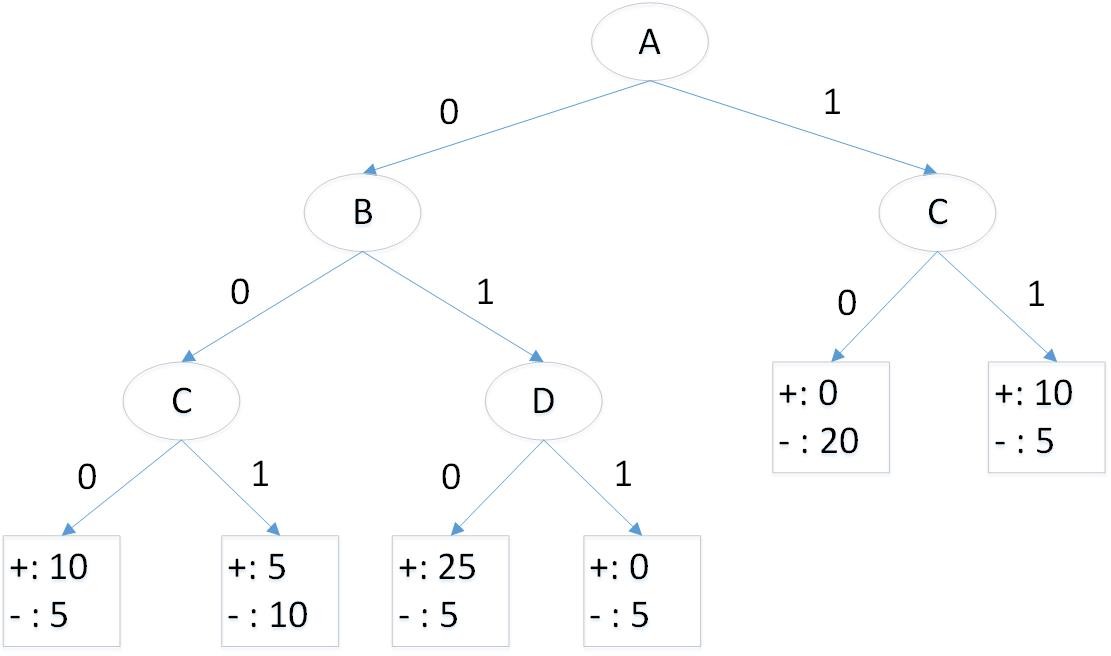
30 30

Gini(children) = 60 *×* 0*.*4444 + 60 *×* 0*.*4444 = 0*.*4444*.*

Based on the results, attribute *A* should be chosen to split the data since it has a lower gini.

* + 1. Consider the decision tree shown in Figure 3.22 for a binary classification problem. Assume the classes are denoted as + and *−*, respectively. Suppose there are four binary attributes in the data, *A*, *B*, *C*, and *D*.

The counts shown in the leaf nodes of the tree correspond to the number of training examples assigned to the nodes. Assume that the decision tree classifier assigns the majority class of training examples as the class label of each leaf node.



**Figure 3.22.** Unpruned decision tree

* + - 1. Calculate the training error rate of the decision tree.

**Answer:** There are 100 examples in the training set. Assuming the leaf nodes are assigned to the majority class of training examples,

the training error rate of the tree is  20

100

= 0*.*2.

* + - 1. Calculate the generalization error rate of the decision tree using the validation set given below. Note that the wildcard *∗* shown in the table means the value could be either 0 or 1.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| *A* | *B* | *C* | *D* | Number of + examples | Number of *−* examples |
| 0 | 0 | 0 | \* | 20 | 10 |
| 0 | 0 | 1 | \* | 0 | 5 |
| 0 | 1 | \* | 0 | 10 | 5 |
| 0 | 1 | \* | 1 | 5 | 5 |
| 1 | \* | 0 | \* | 5 | 25 |
| 1 | \* | 1 | \* | 10 | 0 |

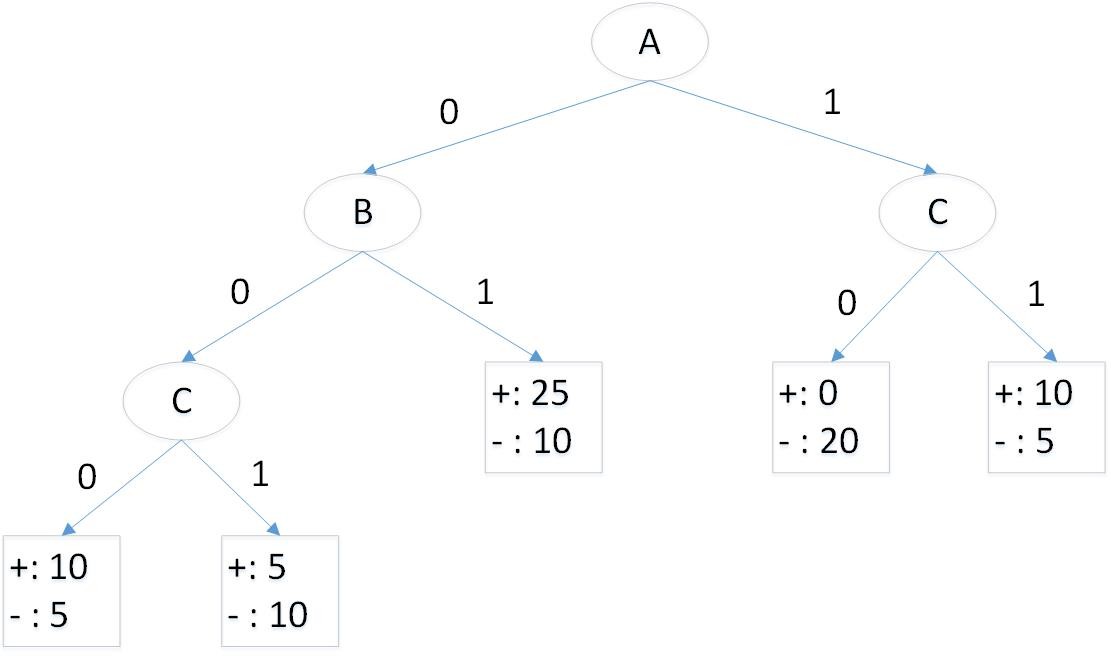
**Answer:** There are 100 examples in the validation set. The esti-

mated generalization error of the tree is  25

100

= 0*.*25.

* + - 1. Calculate the training and generalization error estimate of the pruned decision tree shown in Figure 3.23. Based on your estimate of gen-



**Figure 3.23.** Pruned decision tree

eralization error, which tree should be preferred (the unpruned tree given in Figure 3.22 or the pruned tree given in Figure 3.23)?

**Answer:** The training error of the pruned tree is  25

100

= 0*.*25 while

its generalization error is  25

100

= 0*.*25. According to estimated gener-

alization errors using the validation set, both trees are equivalent.

However, since the pruned tree is simpler, it should be preferred over the unpruned tree.

* + - 1. Apply the minimum description length principle to determine which tree should be preferred (the unpruned tree given in Figure 3.22 or the pruned tree given in Figure 3.23). Assume the tree requires log2 *c* bits to encode each leaf node (where *c* is number of classes), log2 *d* bits to encode each internal node (where *d* is number of at- tributes), and log2 *N* bits to encode each misclassified training ex- ample (where *N* is the number of training examples).

**Answer:** The unpruned tree has 5 internal and 6 leaf nodes. It also made 20 mistakes on a training set of size 100. Thus, the total

description length for the unpruned tree is

6 *×* log2 2 + 5 *×* log2 4 + 20 *× [*log2 100*|* = 156 bits

The pruned tree has 4 internal and 5 leaf nodes. It made 25 mistakes on the training set. So, the total description length for the pruned tree is

5 *×* log2 2 + 4 *×* log2 4 + 25 *× [*log2 100*|* = 188 bits

According to the MDL principle, the unpruned tree should be pre- ferred over the pruned one.

### Model Evaluation

* + 1. You have been asked to develop a classification model for diagnosing whether a patient is infected with a certain disease. To help you con- struct the models, your collaborator has provided you with a small train- ing set (*N* = 10) with equal number of positive and negative examples. You tried several approaches and found two most promising models, *C*1 and *C*2. The outputs of the models in terms of predicting whether each of the training examples belong to the “positive” class are summarized in the table below. The first row shows the probability a training example belongs to the positive class according to classifier *C*1, while the second row shows the same information for classifier *C*2. The last row indicates the true class label of the 10 training examples.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *P* (*y* = +*|C*1) | 0.1 | 0.15 | 0.2 | 0.3 | 0.31 | 0.4 | 0.62 | 0.77 | 0.81 | 0.95 |
| *P* (*y* = +*|C*2) | 0.25 | 0.49 | 0.05 | 0.35 | 0.66 | 0.6 | 0.7 | 0.65 | 0.55 | 0.99 |
| *y* | - | + | - | - | + | - | + | + | - | + |

* + - 1. Draw the corresponding ROC curves for both classifiers on the same plot.

**Answer:** See Figure 3.24.

* + - 1. Compute the area under ROC curve for each classifier. Which classifier has a larger area under the ROC curve?

##### Answer:

*AUC*(*C*1) = 0*.*68

1

Classifier1

Classifier2

0.9

0.8

0.7

0.6

True Positive

0.5

0.4

0.3

0.2

0.1

0

0 0.2 0.4 0.6 0.8 1

False Positive

**Figure 3.24.** ROC curves for both classes.

*AUC*(*C*2) = 0*.*92

Hence, C2 has larger area under the ROC curve.

* + - 1. Compute the Wilcoxon Mann Whitney statistic for both classifiers. The statistic can be computed as follows:

*m—*1 *n—*1

Σ Σ *I*(*x , y* )*i j*

*WMW* = *i*=0 *j*=0 *,* (3.10)

*mn*

where

*I*(*x, y*) = 1*, xi > yj* ;

0*,* otherwise.

Note that *{x*0*, x*1*, · · · , xm—*1*}* correspond to the classifier outputs for the *m* positive examples while *{y*0*, y*1*, · · · , yn—*1*}* correspond to the classifier outputs for the *n* negative examples (in this exercise,

*m* = *n* = 5). Which classifier has a larger WMW value? Based on your answers, state the relationship between WMW and the ROC curve.

##### Answer:

*WMW* (*C*1) =

*WMW* (*C*2) =

1 + 3 + 4 + 4 + 5

25

3 + 5 + 5 + 5 + 5

25

= 0*.*68

= 0*.*92

Hence, C2 has larger WMW. WMW is equivalent to the area under

ROC curve.

* + 1. You have been asked to develop a classification model for diagnosing whether a patient is infected with a certain disease. To help you evaluate the models, your collaborator has provided you with a small test set (*N* = 10) with equal number of positive and negative examples. You applied two classifiers, *C*1 and *C*2. The outputs of the classifiers in terms of predicting whether each of the test examples belong to the “positive” class are summarized in the table below. The first row shows the probability a test example belongs to the positive class according to classifier *C*1, while the second row shows the same information for classifier *C*2. The last row indicates the true class label of the 10 test examples.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *P* (*y* = +*|C*1) | 0.15 | 0.2 | 0.25 | 0.37 | 0.41 | 0.55 | 0.65 | 0.8 | 0.92 | 0.99 |
| *P* (*y* = +*|C*2) | 0.33 | 0.22 | 0.1 | 0.41 | 0.68 | 0.59 | 0.72 | 0.75 | 0.64 | 0.95 |
| *y* | - | - | + | - | + | - | - | + | + | + |

* + - 1. Draw the corresponding ROC curves for both classifiers on the same plot. **Answer:** See Figure 3.25.
      2. For each classifier *Ci*, what is the optimal threshold we should use for *P* (*y* = +*|Ci*) to obtain a high true positive rate and low false positive rate? Draw the confusion matrix associated with each clas-

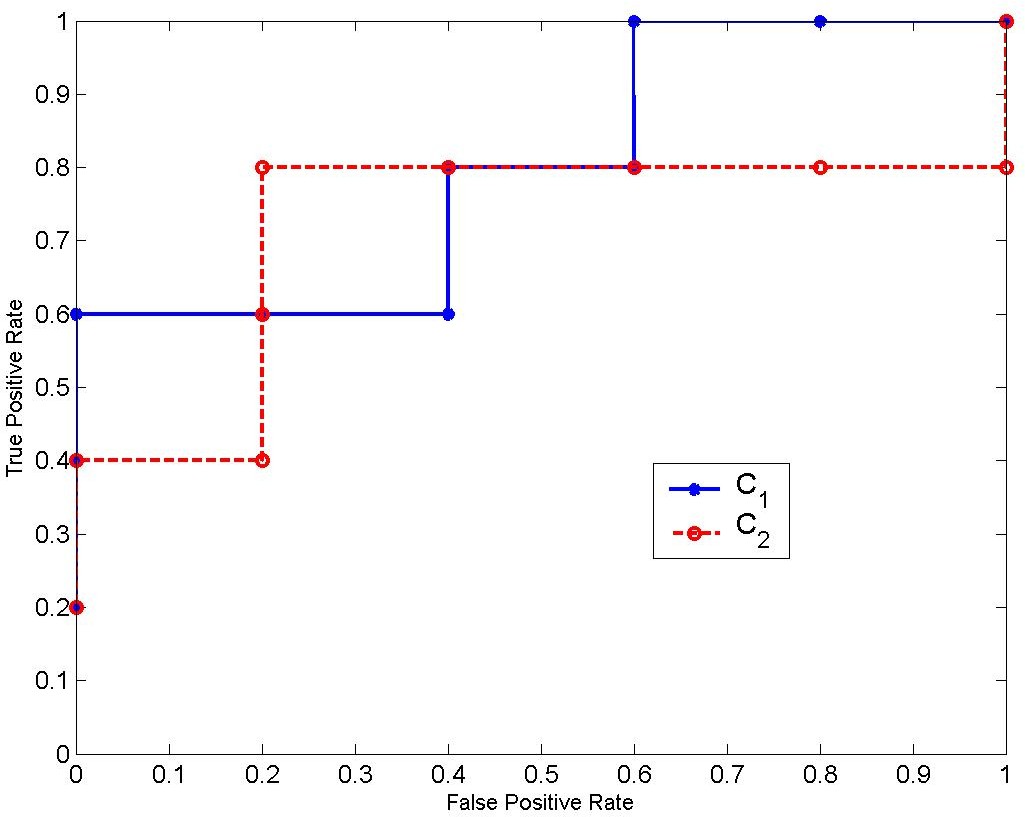
sifier (using the selected optimal threshold for predicting positive class). Based on your results, which classifier is better in terms of

* + - * 1. accuracy, and (b) F-measure?

**Answer:** For classifier *C*1, the optimal threshold is when *P* (*y* =

+*|C*1) *≥* 0*.*7. The confusion matrix for the classifier is

|  |  |  |  |
| --- | --- | --- | --- |
| *P* (*y* = +*|C*1) *≥* 0*.*7 | | Predicted | |
| + | - |
| Actual | + | 3 | 2 |
| - | 0 | 5 |



**Figure 3.25.** ROC curve for classifiers *C*1 and *C*2

*F* -measure = 2*×*3 = 6 = 0*.*75.

Accuracy =  8

2*×*3+2+0 8

= 0*.*8.

10

For classifier *C*2, the optimal threshold is when *P* (*y* = +*|C*1) *≥*

0*.*625. The confusion matrix for the classifier is

|  |  |  |  |
| --- | --- | --- | --- |
| *P* (*y* = +*|C*2) *≥* 0*.*625 | | Predicted | |
| + | - |
| Actual | + | 4 | 1 |
| - | 1 | 4 |

*F* -measure = 2*×*4 =  8

= 0*.*8.

Accuracy =  8

2*×*4+1+1 10

= 0*.*8.

10

*C*2 is better than *C*1 in terms of F-measure but they both have the same accuracy.

* + - 1. Compute the area under ROC curve for each classifier. Which classifier has a larger area under the ROC curve?

##### Answer:

For *C*1: Area under ROC curve = 0*.*4*×*0*.*6+0*.*2*×*0*.*8+0*.*4*×*1 = 0*.*8.

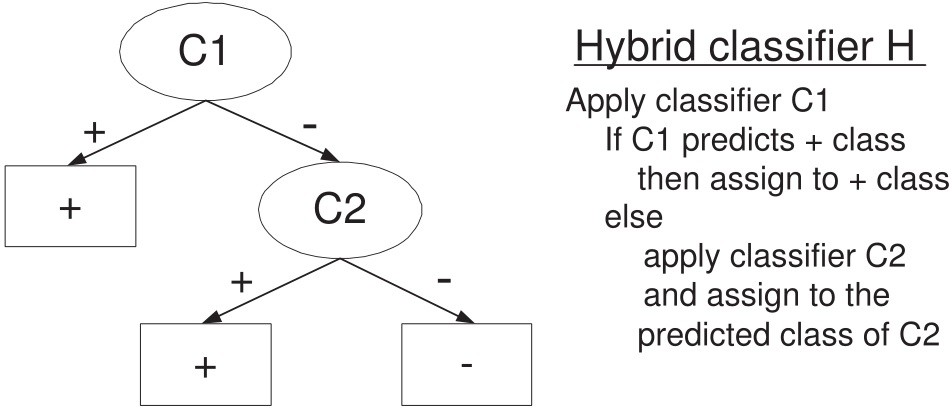
For *C*2: Area under ROC curve = 0*.*2 *×* 0*.*4 + 0*.*8 *×* 0*.*8 = 0*.*72. Classifier *C*1 has a larger area under ROC curve.

* + - 1. Suppose a binary classifier produces a true positive rate of 40% and false positive rate of 60% (i.e., worse than random guessing).

Explain a simple approach you can use to improve the performance of the classifier so that it performs better than random guessing. What is the expected true positive rate and false positive rate of the classifier using your proposed approach?

**Answer:** A simple approach would be to predict the opposite of what the classifier says. For example, if the classifier predicts it to be a positive class, we should predict it as negative class instead. Similarly, if the classifier predicts it to be a negative class, we should declare it as positive. By reversing the prediction, the true positive rate of the classifier becomes 60% and its false positive rate becomes 40%.

* + 1. Suppose we are given a pair of “independent” classifiers, *C*1and *C*2 (being independent means their errors are uncorrelated).



Assume the classifiers have been trained on a two-class problem (denoted as positive and negative class, respectively). The class distribution is skewed, i.e., the proportion of negative class outnumbers the positive class by 9:1. The precision and recall for classifier *C*1 (with respect to the positive class) are 0.5 and 0.8, respectively. On the other hand, the precision and recall for classifier *C*2 (with respect to the positive class) are both 0.6. Consider the hybrid classifier obtained by combining *C*1 and *C*2. Assume the precision and recall of *C*2 remain unchanged when used in the hybrid setting (even though the class proportion has changed). Compare the F-measure of the hybrid classifier against that for *C*1 and *C*2. Which of them is the best? Show your steps clearly to receive full credit.

**Answer:** Suppose there are *N* total examples. Based on the information given, the confusion matrix for classifiers *C*1 and *C*2 are given below:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Classifier  *C*1 | | Predicted class | |  |
| + | *−* |
| Actual  class | + | 0.08N | 0.02N | 0.1N |
| *−* | 0.08N | 0.82N | 0.9N |
|  | | 0.16N | 0.84N | N |

Precision = 0.5, Recall = 0.8, F-measure =  2*rp*

*r*+*p*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Classifier  *C*2 | | Predicted class | |  |
| + | *−* |
| Actual  class | + | 0.06N | 0.04N | 0.1N |
| *−* | 0.04N | 0.86N | 0.9N |
|  | | 0.1N | 0.9N | N |

Precision = 0.6, Recall = 0.6, F-measure =  2*rp*

*r*+*p*

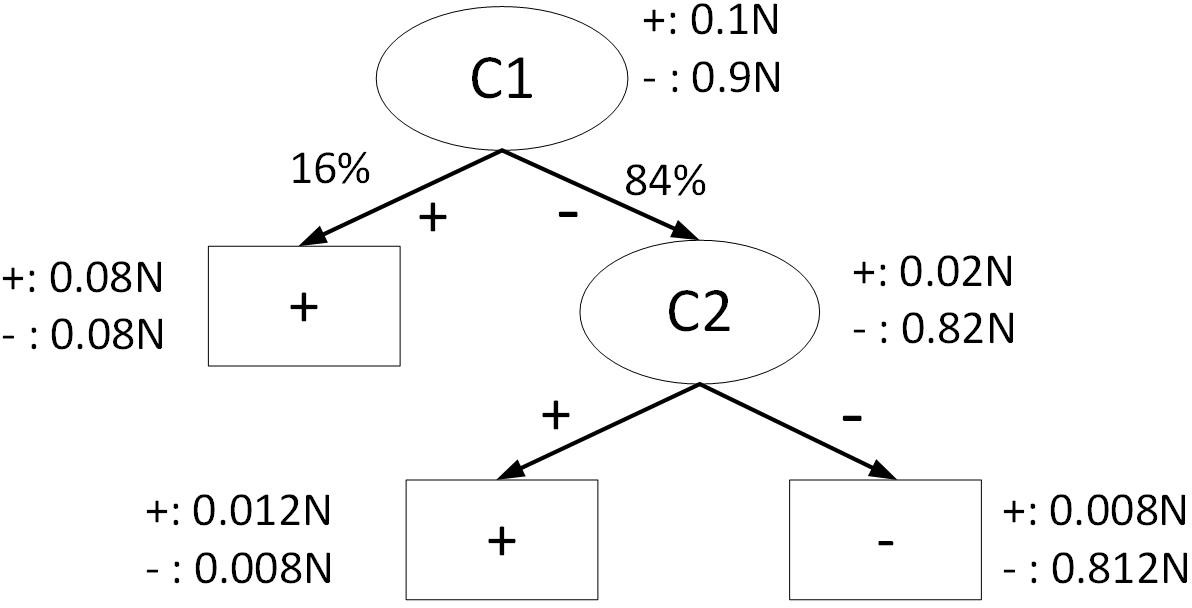
= 0*.*6154

= 0*.*6

For the hybrid classifier, after applying *C*1, 84% of the training exam- ples will be propagated to *C*2, out of which there are 0*.*02*N* positive and 0*.*82*N* negative examples. Based on this information, the confusion ma- trix for *C*2 in the hybrid setting is given below (assuming it maintains the same precision and recall values as before):

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Classifier  *C*2 | | Predicted class | |  |
| + | *−* |
| Actual  class | + | 0.012N | 0.008N | 0.02N |
| *−* | 0.008N | 0.812N | 0.82N |
|  | | 0.02N | 0.82N | 0.84N |

A summary of the class distribution at each leaf node in the hybrid classifier is shown in the Figure below.



The precision, recall, and F-measure for the hybrid classifier are:

0*.*08*N* +0*.*012*N*

Precision = = 0*.*5111

0*.*08*N* +0*.*08*N* +0*.*012*N* +0*.*008*N*

Recall = 0*.*08*N* +0*.*012*N*

0*.*1*N*

= 0.92

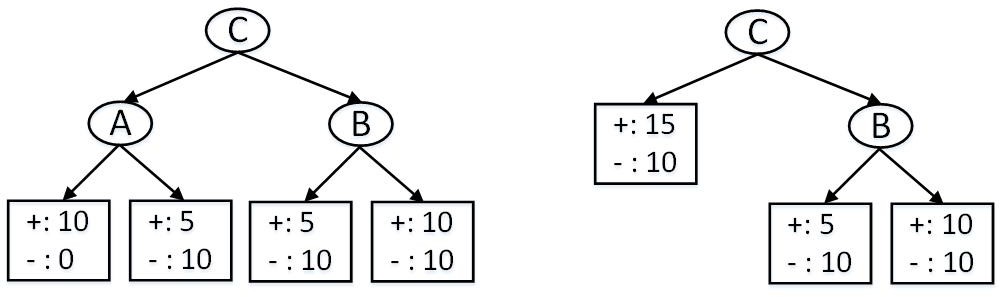
F-measure = 2*×*0*.*5111*×*0*.*92 = 0*.*6571*.*

0*.*5111+0*.*92

This is better than the F-measure for both *C*1 and *C*2.

* + 1. Consider the two decision trees shown in Figure 3.26 for a binary classifi- cation problem. Assume the classes are denoted as + and *−*, respectively. Suppose there are four binary attributes in the data, *A*, *B*, *C*, and *D*.

The counts shown in the leaf nodes of the tree correspond to the number of training examples assigned to the nodes. Assume that the decision tree classifier assigns the majority class of training examples as the class label of each leaf node.



**Figure 3.26.** Two candidate decision trees

* + - 1. Draw the confusion matrix for both trees on the training data. A confusion matrix is a table that summarizes the number of examples correctly or incorrectly predicted by the model. For example:

|  |  |  |  |
| --- | --- | --- | --- |
|  | | Predicted | |
| + | *−* |
| Actual | + | *n*++ | *n*+*—* |
| *−* | *n—*+ | *n——* |

In the above table, *n*+*—* is the number of positive examples incor- rectly predicted as negative class.

**Answer:** For the left tree, we may assign the class labels of the leaf nodes as follows (from left to right): +, -, -, +. In this case, the confusion matrix for the left tree is

|  |  |  |  |
| --- | --- | --- | --- |
| C1 | | Predicted | |
| + | *−* |
| Actual | + | 20 | 10 |
| *−* | 10 | 20 |

If the right-most leaf node was assigned to the negative class, then the confusion matrix for the left tree is

|  |  |  |  |
| --- | --- | --- | --- |
| C2 | | Predicted | |
| + | *−* |
| Actual | + | 10 | 20 |
| *−* | 0 | 30 |

For the right tree, we may assign the class labels of the leaf nodes as follows (from left to right): +, -, +. In this case, the confusion matrix for the left tree is

|  |  |  |  |
| --- | --- | --- | --- |
| C3 | | Predicted | |
| + | *−* |
| Actual | + | 25 | 5 |
| *−* | 20 | 10 |

If the right-most leaf node was assigned to the negative class, then the confusion matrix for the right tree is

|  |  |  |  |
| --- | --- | --- | --- |
| C4 | | Predicted | |
| + | *−* |
| Actual | + | 15 | 15 |
| *−* | 10 | 20 |

* + - 1. Calculate the training error rate of both decision trees. Which tree has a lower training error?

**Answer:** The training error rate for the left tree is: Training error = 20 = 0*.*33.

60

The training error rate for the right tree is:

Training error = 25 = 0*.*42.

60

Thus, the left tree has a lower error rate.

* + - 1. Apply the minimum description length principle to determine which tree should be preferred.

**Answer:** To apply MDL, we first compute the following:

The cost for encoding each internal node = log2 *d* = log2 4 = 2 bits. The cost for encoding each leaf node = log2 *c* = log2 2 = 1 bit.

The cost for encoding each error = log2 *N* = *[*log2 60*|* = 6 bits. Since the left tree has 3 internal nodes, 4 leaf nodes, and misclassifies

20 examples, its total description length is

3 *×* 2 + 4 *×* 1 + 20 *×* 6 = 130 bits*.*

Since the right tree has 2 internal nodes, 3 leaf nodes, and misclas- sifies 25 examples, its total description length is

2 *×* 2 + 3 *×* 1 + 25 *×* 6 = 157 bits*.*

Thus, according to the MDL principle, the left tree should be pre- ferred.