- 18.3 (Irfandi) Suppose we generate a training set from a decision tree and then apply decision tree learning to that training set. Is it the case that the algorithm will eventually return the correct tree as the training set size goes to infinity? Why or why not?

18.3 The algorithm may not return the "correct" tree, but it will return a tree that is logically equivalent, assuming that the method for generating examples eventually generates all possible combinations of input attributes. This is true because any two decision tree defined on the same set of attributes that agree on all possible examples are, by definition, logically equivalent. The actually form of the tree may differ because there are many different ways to represent the same function. (For example, with two attributes A and B we can have one tree with A at the root and another with B at the root.) The root attribute of the original tree may not in fact be the one that will be chosen by the information gain geuristic when applied to the training examples.

- 18.4 (Gavin) In the recursive construction of decision trees, it sometimes happens that a mixed set of positive and negative examples remains at a leaf node, even when all of the attributes have been used. Suppose that we have *p* positive examples and *n* negative examples.

    - Show that the solution used by Decision-Tree-Learning, which picks the majority classification, minimizes the absolute error over the set of examples in the leaf.

    - Show that the class probability *p/(p+n)* minimizes the sum of squared errors *(x-y)^2* where *x* and *y* are the predicted probability and true class (0 or 1) respectively.

18.4 This question brings a little bit of mathematics to bear on the analysis of the learning problem, preparing the ground for Chapter 20. Error minimization is a basic technique in both statistics and neural nets. The main thing is to see that the error on a given training set can be written as a mathematical expression and viewed as a function of the hypothesis chosen. Here, the hypothesis in question is a single number $\alpha \in [0, 1]$ returned at the leaf.

a. If α is returned, the absolute error is

$$\begin{aligned} E = p(1-\alpha) + n\alpha = \alpha(n-p) + p &= n \text{ when } \alpha = 1 \\ &= p \text{ when } \alpha = 0 \end{aligned}$$

This is minimized by setting

$$\alpha = 1 \text{ if } p > n$$
$$\alpha = 0 \text{ if } p < n$$

That is, α is the majority value.

b. First calculate the sum of squared errors, and its derivative:

$$E = p(1-\alpha)^2 + n\alpha^2$$
$$\frac{dE}{d\alpha} = 2\alpha n - 2p(1-\alpha) = 2\alpha(p+n) - 2p$$

The fact that the second derivative, $\frac{d^2E}{d\alpha^2} = 2(p+n)$, is greater than zero means that E is minimized (not maximized) where $\frac{dE}{d\alpha} = 0$, i.e., when $\alpha = \frac{p}{p+n}$.

- 18.7 (Fatemeh) A decision *graph* is a variant of a decision tree that allows nodes (i.e. attributes used for splits) to have multiple parents, rather than a single parent. The resulting graph must still be acyclic. Now consider the XOR function over three binary attributes, which produces value 1 if and only if an odd number of input attributes has value 1.

    - Draw a minimum-sized decision tree for the three-input XOR function.

    - Draw a minimum-sized decision graph for the three-input XOR function.

18.7 See Figure S18.1, where nodes on successive rows measure attributes A_1, A_2, and A_3. (Any fixed ordering works.)

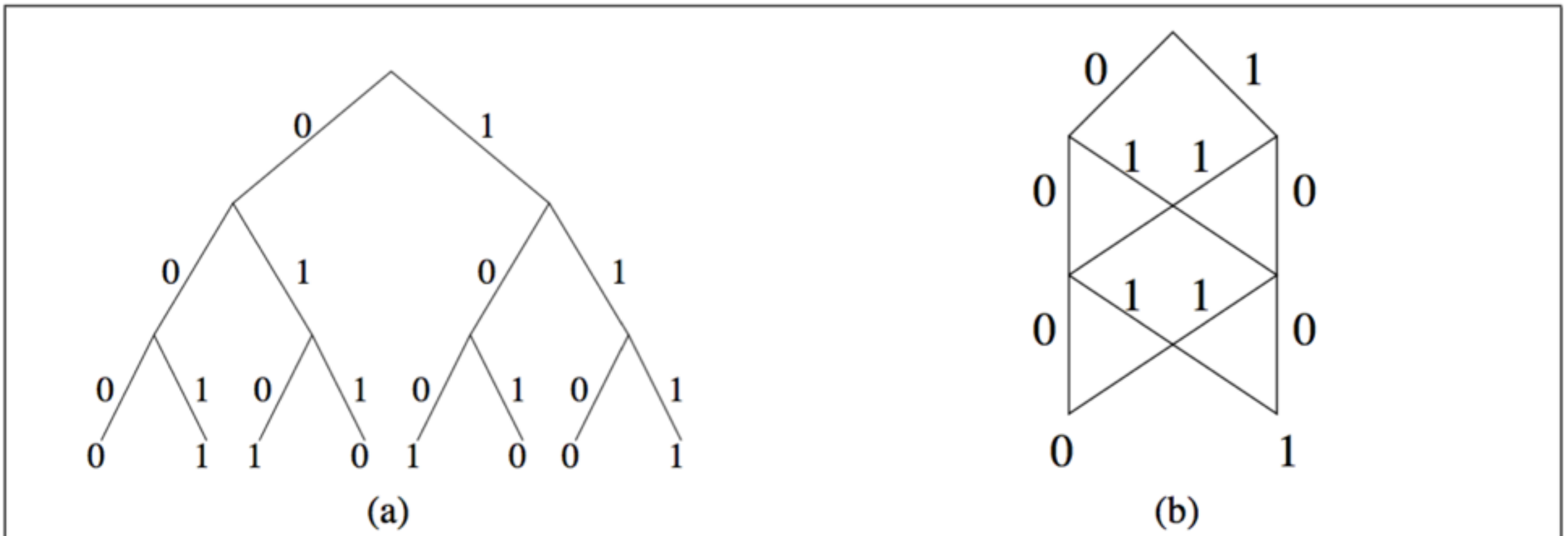

Figure S18.1 XOR function representations: (a) decision tree, and (b) decision graph.

- 18.12 (Qingcan) Consider the WillWait data we saw in lecture. Apply the Decision Tree Learning algorithm to learn a decision tree for this data set. Show all computations you performed. You may skip considering some attributes when finding a split, but compare at least two attributes for each split.

Example	Attributes										Target
	Alt	*Bar*	*Fri*	*Hun*	*Pat*	*Price*	*Rain*	*Res*	*Type*	*Est*	*WillWait*
X_1	*T*	*F*	*F*	*T*	*Some*	*$$$*	*F*	*T*	*French*	*0–10*	*T*
X_2	*T*	*F*	*F*	*T*	*Full*	*$*	*F*	*F*	*Thai*	*30–60*	*F*
X_3	*F*	*T*	*F*	*F*	*Some*	*$*	*F*	*F*	*Burger*	*0–10*	*T*
X_4	*T*	*F*	*T*	*T*	*Full*	*$*	*F*	*F*	*Thai*	*10–30*	*T*
X_5	*T*	*F*	*T*	*F*	*Full*	*$$$*	*F*	*T*	*French*	*>60*	*F*
X_6	*F*	*T*	*F*	*T*	*Some*	*$$*	*T*	*T*	*Italian*	*0–10*	*T*
X_7	*F*	*T*	*F*	*F*	*None*	*$*	*T*	*F*	*Burger*	*0–10*	*F*
X_8	*F*	*F*	*F*	*T*	*Some*	*$$*	*T*	*T*	*Thai*	*0–10*	*T*
X_9	*F*	*T*	*T*	*F*	*Full*	*$*	*T*	*F*	*Burger*	*>60*	*F*
X_{10}	*T*	*T*	*T*	*T*	*Full*	*$$$*	*F*	*T*	*Italian*	*10–30*	*F*
X_{11}	*F*	*F*	*F*	*F*	*None*	*$*	*F*	*F*	*Thai*	*0–10*	*F*
X_{12}	*T*	*T*	*T*	*T*	*Full*	*$*	*F*	*F*	*Burger*	*30–60*	*T*

18.12

Test	If yes	If no
$A_1 = 1$	1	next test
$A_3 = 1 \wedge A_4 = 0$	0	next test
$A_2 = 0$	0	1