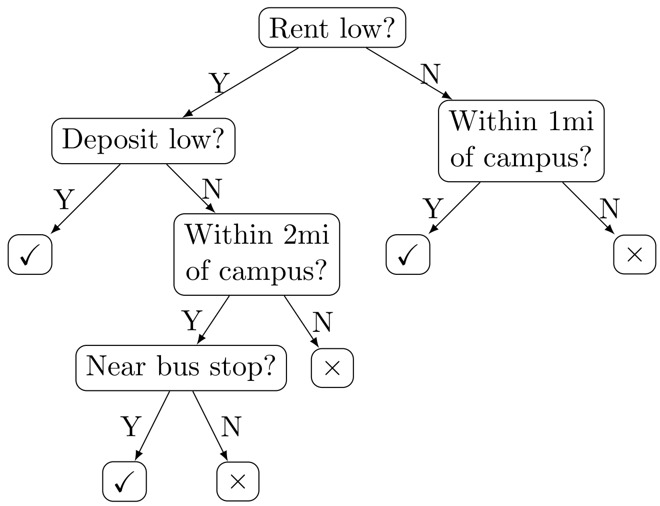
07-280 Decision Trees

Nihar B. Shah

Consider the following setting. You want to rent an apartment. Among the many options available, you want to decide which apartments to check out in person. How do you decide?



- Internal nodes split on some attribute
- Leaves have output labels (✓ or ×)

This is called a **decision tree**.

Major Benefit: Human interpretable.

We created this using a rule-based approach. In many situations, you don't have access to such rules or the setting is just too complex.

ML approach: Given examples, how do you construct a decision tree?

What do the examples comprise?

In this problem, each example is one prospective apartment:

- "x" is answers to "Rent low?", "Deposit low?", "1 mi?", "2 mi?", "bus stop?"
- "y" is answer to whether to go check the apt or not.

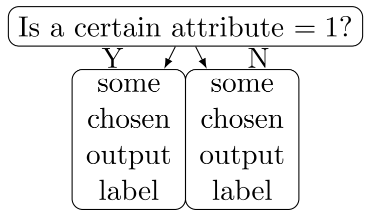
Formally:

$$N = \#\text{ prospective apts}$$
$$\mathcal{X} = \{\text{Yes}, \text{No}\}^5 = \{0,1\}^5$$
$$\mathcal{Y} = \{\text{Y}, \text{N}\} = \{0,1\}$$

Once you construct a decision tree, given any new apt $x^{(\text{new})} \in \mathcal{X}$, you run it through the tree to get an output as to whether you should check it out or not.

More generally, consider $\mathcal{X} = \{0,1\}^d$ (d attributes), and $\mathcal{Y} = \{0,1\}$.

Key building block: Decision Stump Suppose you can split just once...



Constructing the Decision Tree

Algorithm 1 Decision Tree Construction

function DT(dataset S, set of attributes $A \subseteq \{1, 2, \ldots, d\}$)
  if $A = \emptyset$ **or** all examples in S have same label **then**
    return a leaf with value = majority vote of labels in S
  end if
  Let $b = \textsc{BestAttribute}(S, A) \in A$
  Let LeftNode = DT($\{(x,y) \in S \mid x_b = 1\}, A \setminus \{b\}$)
  Let RightNode = DT($\{(x,y) \in S \mid x_b = 0\}, A \setminus \{b\}$)
  return tree "Is b^{th} attribute equal to 1?" (Y: LeftNode, N: RightNode)
end function

This function is called with $S = \{(x_i, y_i)\}_{i=1}^N$ and $A = \{1, 2, \ldots, d\}$.

Consider one kind of BestAttribute selection: Assume you can split only once (i.e., create a decision stump). Then what choice of attribute to split (and appropriate choice of labels for the two resulting leaves) will lead to lowest training error?

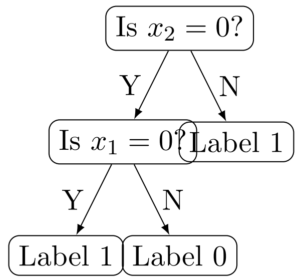
Example (Poll): Training data

y	x_1	x_2	x_3
0	1	0	0
0	1	0	1
0	1	0	0
1	0	0	1
1	1	1	0
1	1	1	1
1	1	1	0
1	1	1	1

Answer: In poll.

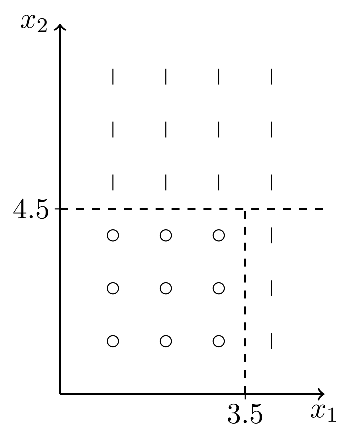
The decision tree construction algorithm initially picks the attribute which would result in the lowest training error if allowed only one split. (Note that this is a greedy approach towards ERM, where greedyness is suboptimal but helps in computational efficiency.) It then passes all training samples which have that attribute equal to 1 to the left node, and remaining samples to the right node. It now proceeds recursively until a stopping criterion is met. At this point, we are at a leaf and the algorithm outputs a label.

In the aforementioned example, the tree is:



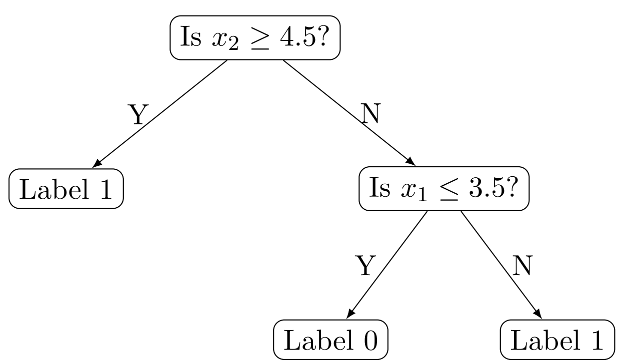
Various Extensions

- **Attributes having more than $\{0, 1\}$ values**



E.g., $\mathcal{X} = \mathbb{R}^2$, $\mathcal{Y} = \{0, 1\}$

Tree we get:



"Decision boundaries" the set of lines that separate output '0' from output '1' regions.

- Need not remove chsen attribute when calling function again in recursion

- **Different stopping criteria** (e.g., stop if the split doesn't reduce training error by much)
- **BestAttribute():**
  - Minimize training error (earlier)
  - Minimize Gini impurity (won't discuss here)
  - Maximize mutual information / information gain (below)

Information Theory

We will discuss a few concepts on quantifying uncertainty and information that are widely used in machine learning and in many other disciplines.

Entropy

First consider a "20 questions" like game. Suppose I choose one of the students in the class uniformly at random. There are 128 students in the class. You can ask me Yes/No questions and have to guess who I chose. How many questions would you need on average?

Answer $= 7 = \log_2 128 = \log_2 \frac{1}{1/128}$

Notice that the probability with which I chose any given student was 1/128. Thus the term $\log_2 \frac{1}{\text{Probability}}$ is fundamental! If $k = \#\text{students}$ and $p_1, \ldots, p_k$ are probabilities of choosing students $1, \ldots, k$ respectively, then here $p_1 = p_2 = \cdots = p_k = \frac{1}{128}$. We can rewrite:

$$\text{Answer} = \sum_{i=1}^{k} p_i \log \frac{1}{p_i}$$

Now suppose I choose a student not uniformly at random, but with probabilities $p_1 = \frac{1}{4}$, $p_2 = \frac{1}{8}, \ldots, p_7 = \frac{1}{8}$, $p_8, \ldots, p_{128} = 0$. Then average number of questions you need to identify the chosen student is still $\sum_{i=1}^{k} p_i \log \frac{1}{p_i}$ (where $0 \cdot \log \frac{1}{0} := 0$). This equals 11/4 for this problem.

The quantity $\sum_{i=1}^{k} p_i \log \frac{1}{p_i}$ is called the **entropy** of the distribution $(p_1, \ldots, p_k)$.

There are many cool properties, uses, and results about entropy. Measures amount of uncertainty: more uncertainty $\Longleftrightarrow$ more questions needed.

Recall that we wanted to choose attributes that reduce amount of uncertainty of the labels. Wen will now work towards that.

Let $\{P(Y = y)\}_{y \in \mathcal{Y}}$ be the distribution of the labels in S. E.g., if 40% of labels in S are "1" and 60% are "0" then $P(Y = 1) = 0.4$ and $P(Y = 0) = 0.6$.

Then the entropy of these labels is:

$$\sum_{y \in \mathcal{Y}} P(Y = y) \log \frac{1}{P(Y = y)},$$

denoted as H(Y).

We want to choose an attribute so that splitting on this attribute will reduce the entropy as much as possible.

Consider the setting of binary valued coordinates of each x. Consider dataset S, and we want to choose the attribute to split.

As discussed in class, let $H(Y)$ denote the entropy of the labels for set S (i.e., at the current node). Consider any attribute a. Define the following four quantities:

- Let $H(Y|a=0)$ denote the entropy when the dataset is restricted to the subset of S where the a^{th} coordinate of x is 0. This is the amount of uncertainty (entropy) in the right branch after the split.
- Let $H(Y|a=1)$ denote the entropy when the dataset is restricted to the subset of S where the a^{th} coordinate of x is 1. This is the amount of uncertainty (entropy) in the left branch after the split.
- Let $P(a=0)$ denote the fraction of examples in S where the a^{th} coordinate of x is 0
- Let $P(a=1)$ denote the fraction of examples in S where the a^{th} coordinate of x is 1

Then the conditional entropy, denoted as $H(Y|a)$, is given by $P(a=0)H(Y|a=0)+P(a=1)H(Y|a=1)$. This is just the averaged uncertainty after a single split, where the average is weighted based on the fraction of samples that go into the left and the right branches.

The mutual information between attribute a and the labels Y is given by $H(Y)-H(Y|a)$ (denoted as $I(Y;a)$). This is just the amount of reduction in the entropy after splitting once.

The BestAttribute function will choose the attribute a which maximizes the mutual information. This is called the mutual information or the information gain method.