

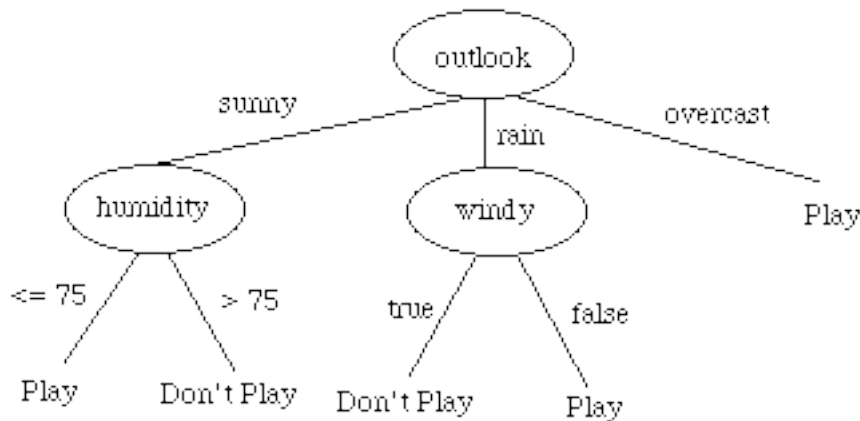
Top Down Induction of Decision Trees

What is a decision tree?

The following is a simple example of a tree for determining whether today is a good golf day.

Non-leaf nodes of the tree specify attributes. Edges specify possible values of those attributes. Leaves specify class names.

Given an example, you would determine the class to which it belonged by tracing through the decision tree from the root to a leaf.



Decision trees are propositional in nature:

$$\text{SunnyDay} \wedge \text{HighHumidity} \Rightarrow \text{Don't Play}$$

Objective: learn a tree that is

Correct (both on training examples and on unseen examples)

Compact (for efficiency, comprehensibility, generality)

Learning a Decision Tree: a Recursive Algorithm

If all examples are from the same class

The tree is a leaf with that class name

Else

Pick an attribute for the decision node (i.e., pick a test to make)

Construct one edge for each possible value of that attribute (i.e., each possible test outcome)

Partition the examples by attribute value (test outcome)

Build subtrees recursively

Note that this is a **greedy** algorithm.

Characteristics of Tests

Let P be the set of examples with class “Play.”

Let D be the set of examples with class “Don’t Play.”

Say that $|P| = 10, |D| = 10$

A Boolean test splits the data into two subsets, T_1 and T_2 .

The best test is one such that $T_1 = P$ and $T_2 = D$.

The worst test is one such that $T_1 = 1/2 P + 1/2 D$ and $T_2 = 1/2 P + 1/2 D$

Selecting Tests: Using ideas from Information Theory

Suppose that we have a set of training examples. We’ll call the set **T**.

Let C_i be the name of a class, where C_1, C_2, \dots, C_n are the class labels assigned to examples in T.

Let **freq**(C_i, T) be the number of examples in the training set that belong to class C_i .

Now let $|T|$ be the number of examples in the training set.

Suppose you were to select one example at random from T and announce that the example belonged to class C_i . This announcement would have probability

$$\frac{\mathbf{freq}(C_i, T)}{|T|}$$

and the information it conveys is

$$-\log_2 (\mathbf{freq}(C_i, T)/|T|) \text{ bits}$$

As the probability goes up, the information conveyed goes to 0. It’s highest for low probabilities.

Here’s one way you might think about this: Say that I have a bag of 100 marbles. 99 of them are blue and one is red. If I pull out a blue one and announce to you that it’s blue, that’s not terribly interesting. On the other hand, if I tell you it’s red, then that’s more interesting.

We can think of the information term as representing the amount of information we need to identify an example as being a member of a particular class.

Entropy

The following measures the average amount of information needed to identify the class of an example in T:

$$\mathbf{info(T)} = -\sum (\text{freq}(C_i, T) / |T|) * \log_2 (\text{freq}(C_i, T) / |T|) ,$$

where the weighted sum is computed over the number of classes in T.

This is called the entropy of T. You can think of it as a measure of the disorder (or impurity) of the set of examples.

[Think about what happens when there are multiple classes and all of the examples in the training set belong to just one of those classes.]

So if we compute $\mathbf{info(T)}$, we get a measure of the disorder of the examples.

Now, how does this relate to the selection of a test? (i.e., the selection of the attribute on which we will branch in the decision tree).

We will select a split of the examples that lessens the disorder.

Information Gain

Let X be an attribute of the examples. Now say that X has n possible values.

If X were selected as a test, we would create a decision tree node with n branches.

Now say that j is a possible value of X. Let T_j be the examples that have the value j for attribute X.

We can compute the average entropy (disorder) that results from making this split:

$$\mathbf{info_x(T)} = \sum (|T_j| / |T|) * \mathbf{info}(T_j),$$

Where the sum is taken over the n possible values of the attribute X.

We can compute this for every attribute.

Once we have done so, we select the attribute that maximizes the value of

$$\text{info}(T) - \text{info}_X(T)$$

Information Gain measures the expected reduction in entropy caused by partitioning the examples according to attribute X.

$$\text{Gain}(T, X) = \text{info}(T) - \text{info}_X(T)$$

An Example. Building the Golf Tree

Say that we would like to build a decision tree from data. The training examples describe days on which it is either a good idea to play golf – or not. The days are described by four attributes: outlook, temperature, humidity, and wind.

The available training examples are as follows:

	Outlook	Temperature	Humidity	Wind	Class
Example 1	Sunny	85	85	False	Don't Play
Example 2	Sunny	80	90	True	Don't Play
Example 3	Overcast	83	88	False	Play
Example 4	Rainy	70	96	False	Play
Example 5	Rainy	68	80	False	Play
Example 6	Rainy	65	70	True	Don't Play
Example7	Overcast	64	65	True	Play
Example8	Sunny	72	95	False	Don't Play
Example9	Sunny	69	70	False	Play
Example10	Rainy	75	80	False	Play
Example11	Sunny	75	70	True	Play
Example12	Overcast	72	90	True	Play
Example13	Overcast	81	75	False	Play
Example14	Rainy	71	96	True	Don't Play

So $|T|$, the number of examples in the training set, = 14.

Now, $\text{freq}(\text{Play}, T) = 9$ and $\text{freq}(\text{Don'tPlay}, T) = 5$.

Then $\text{info}(T) = -((9/14)(\log_2 9/14) + (5/14)(\log_2 5/14)) = 0.94$

Now let's consider splitting this set of examples on the *outlook* attribute.

First, we divide the data into three sets. One contains all the examples for which the outlook is sunny; the next contains all the examples for which it is raining; the third contains all the examples for which it is overcast.

The 14 original examples divide as follows:

Sunny: 5 examples, Play = 2, Don't Play = 3
Rainy: 5 examples, Play = 3, Don't Play = 2
Overcast: 4 examples, Play = 4, Don't Play = 0.

$$\text{info}(\text{Sunny Set}) = -((2/5 \log_2 2/5) + (3/5 \log_2 3/5)) = 0.97$$

$$\text{info}(\text{Rainy Set}) = 0.97$$

$$\text{info}(\text{Overcast Set}) = 0$$

Now let's use these to compute the information needed to classify examples that have been divided on the outlook attribute:

$$(5/14)(0.97) + (5/14)(0.97) + (4/14)(0) = 0.69$$

This gives us an information gain of $0.94 - 0.69 = 0.25$

Now we do the same for the wind attribute. The process of handling the continuous-valued attributes of temperature and humidity is slightly different (but we'll look at that later).

We find that these three yield lower information gain, so we select outlook for our test.

Then we recursively run the algorithm at each child node. That is, we treat the sunny group as if it is an entire set of examples. We determine the attribute that best splits those. We then do the same for the rainy group. We might find that different tests are selected at each of these nodes!