DECISION TREES – LEARNING DECISION TREES

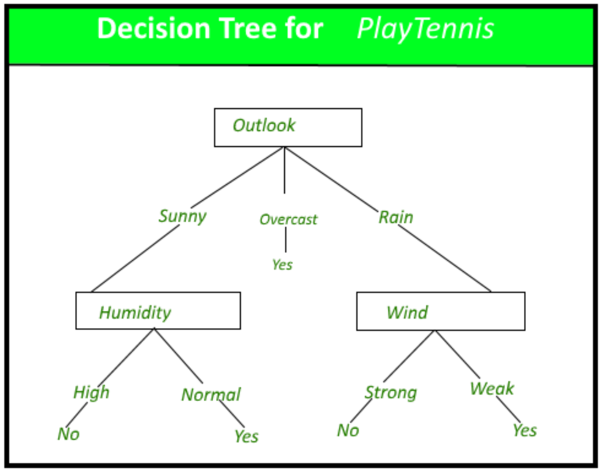
Machine learning

(**ML**) is a field of inquiry devoted to understanding and building methods that 'learn', that is, methods that leverage data to improve performance on some set of tasks.[1]

It is seen as a part of artificial intelligence. Machine learning algorithms build a model based on sample data, known as training data, in order to make predictions or decisions without being explicitly programmed to do so.

Decision trees

Decision tree is the most powerful and popular tool for classification and prediction. A Decision tree is a flowchart like tree structure, where each internal node denotes a test on an attribute, each branch represents an outcome of the test, and each leaf node (terminal node) holds a class label.



As already indicated, for a classification task we can simply define a set of instances *D* to be homogenous if they are all from the same class, and the function Label(*D*) will then obviously return that class.

Notice that in line 5 of Algorithm 5.1we may be calling Label(*D*) with a non-homogeneous set of instances in case one of the *Di* is empty, so the general definition of Label(*D*) is that it returns the majority class of the instances in *D*.2 This leaves us to decide how to define the function BestSplit(*D*,*F*).

Let's assume for the moment that we are dealing with Boolean features, so ***D*** is split into ***D***1 and ***D***2. Let's also assume we have two classes, and denote by $D^{\oplus}$ and $D^{\ominus}$ the positives and negatives in ***D***

negatives. Clearly, the best situation is where $D_1^{\oplus} = D^{\oplus}$ and $D_1^{\ominus} = \emptyset$, or where $D_1^{\oplus} = \emptyset$ and $D_1^{\ominus} = D^{\ominus}$. In that case, the two children of the split are said to be *pure*. So we

One important principle that we will adhere to is that the impurity should only depend on the relative magnitude of $n^{\oplus}$ and $n^{\ominus}$, and should not change if we multiply both with the same amount. This in turn means that impurity can be defined in terms of the proportion $\dot{p} = n^{\oplus}/(n^{\oplus} + n^{\ominus})$, which we remember from Section 2.2 as the ☞*empirical probability* of the positive class.

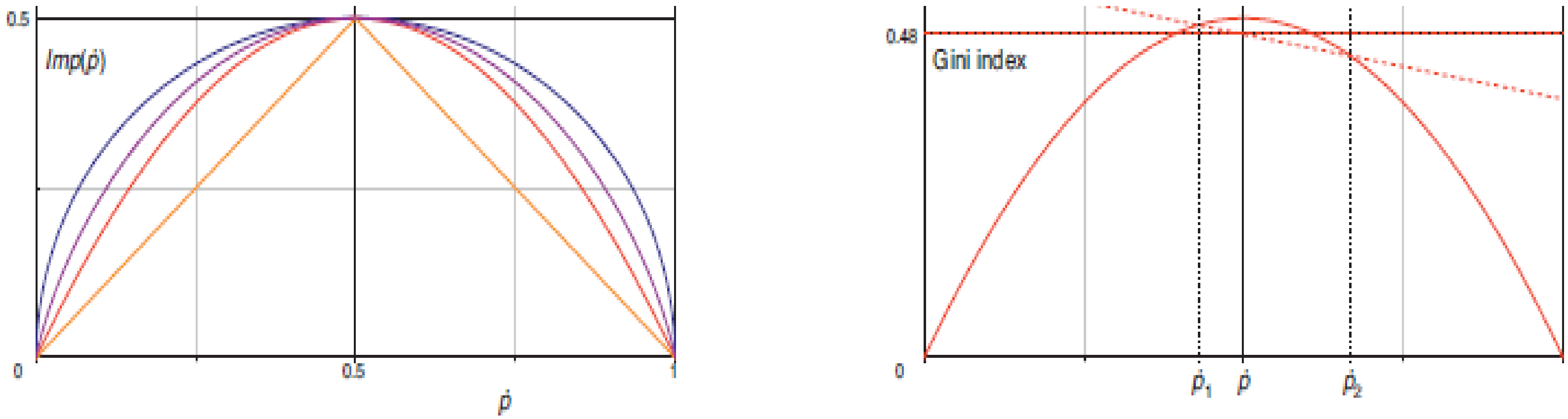


Figure 5.2. **(left)** Impurity functions plotted against the empirical probability of the positive class. From the bottom: the relative size of the minority class, $\min(\dot{p}, 1-\dot{p})$; the Gini index, $2\dot{p}(1-\dot{p})$; entropy, $-\dot{p}\log_2\dot{p} - (1-\dot{p})\log_2(1-\dot{p})$ (divided by 2 so that it reaches its maximum in the same point as the others); and the (rescaled) square root of the Gini index, $\sqrt{\dot{p}(1-\dot{p})}$ – notice that this last function describes a semi-circle. **(right)** Geometric construction to determine the impurity of a split (Teeth = [many, few] from Example 5.1): $\dot{p}$ is the empirical probability of the parent, and $\dot{p}_1$ and $\dot{p}_2$ are the empirical probabilities of the children.

Minority class $\min(\dot{p}, 1-\dot{p})$ – this is sometimes referred to as the error rate, as it measures the proportion of misclassified examples if the leaf was labelled with the majority class; the purer the set of examples, the fewer errors this will make. This impurity measure can equivalently be written as $1/2 - |\dot{p} - 1/2|$.

Gini index $2\dot{p}(1-\dot{p})$ – this is the expected error if we label examples in the leaf randomly: positive with probability $\dot{p}$ and negative with probability $1-\dot{p}$. The probability of a false positive is then $\dot{p}(1-\dot{p})$ and the probability of a false negative $(1-\dot{p})\dot{p}$.[3]

entropy $-\dot{p}\log_2 \dot{p} - (1-\dot{p})\log_2(1-\dot{p})$ – this is the expected information, in bits, conveyed by somebody telling you the class of a randomly drawn example; the purer the set of examples, the more predictable this message becomes and the smaller the expected information.

Example 5.1 (Calculating impurity). Consider again the data in Example 4.4 on p.115. We want to find the best feature to put at the root of the decision tree. The four features available result in the following splits:

Length = [3,4,5]	[2+,0−][1+,3−][2+,2−]
Gills = [yes, no]	[0+,4−][5+,1−]
Beak = [yes, no]	[5+,3−][0+,2−]
Teeth = [many, few]	[3+,4−][2+,1−]

Let's calculate the impurity of the first split. We have three segments: the first one is pure and so has entropy 0; the second one has entropy $-(1/4)\log_2(1/4) - (3/4)\log_2(3/4) = 0.5 + 0.31 = 0.81$; the third one has entropy 1. The total entropy is then the weighted average of these, which is $2/10 \cdot 0 + 4/10 \cdot 0.81 + 4/10 \cdot 1 = 0.72$.

Example 4.4 (Data that is not conjunctively separable). Suppose we have the following five positive examples (the first three are the same as in Example 4.1):

p1: Length = 3 $\wedge$ Gills = no $\wedge$ Beak = yes $\wedge$ Teeth = many
p2: Length = 4 $\wedge$ Gills = no $\wedge$ Beak = yes $\wedge$ Teeth = many
p3: Length = 3 $\wedge$ Gills = no $\wedge$ Beak = yes $\wedge$ Teeth = few
p4: Length = 5 $\wedge$ Gills = no $\wedge$ Beak = yes $\wedge$ Teeth = many
p5: Length = 5 $\wedge$ Gills = no $\wedge$ Beak = yes $\wedge$ Teeth = few

and the following negatives (the first one is the same as in Example 4.2):

n1: Length = 5 $\wedge$ Gills = yes $\wedge$ Beak = yes $\wedge$ Teeth = many
n2: Length = 4 $\wedge$ Gills = yes $\wedge$ Beak = yes $\wedge$ Teeth = many
n3: Length = 5 $\wedge$ Gills = yes $\wedge$ Beak = no $\wedge$ Teeth = many
n4: Length = 4 $\wedge$ Gills = yes $\wedge$ Beak = no $\wedge$ Teeth = many
n5: Length = 4 $\wedge$ Gills = no $\wedge$ Beak = yes $\wedge$ Teeth = few

The least general complete hypothesis is Gills = no $\wedge$ Beak = yes as before, but this covers n5 and hence is inconsistent. There are seven most general consistent hypotheses, none of which are complete:

Length = 3	(covers p1 and p3)
Length = [3,5] $\wedge$ Gills = no	(covers all positives except p2)
Length = [3,5] $\wedge$ Teeth = few	(covers p3 and p5)
Gills = no $\wedge$ Teeth = many	(covers p1, p2 and p4)
Gills = no $\wedge$ Beak = no	
Gills = yes $\wedge$ Teeth = few	
Beak = no $\wedge$ Teeth = few	

The last three of these do not cover any positive examples.

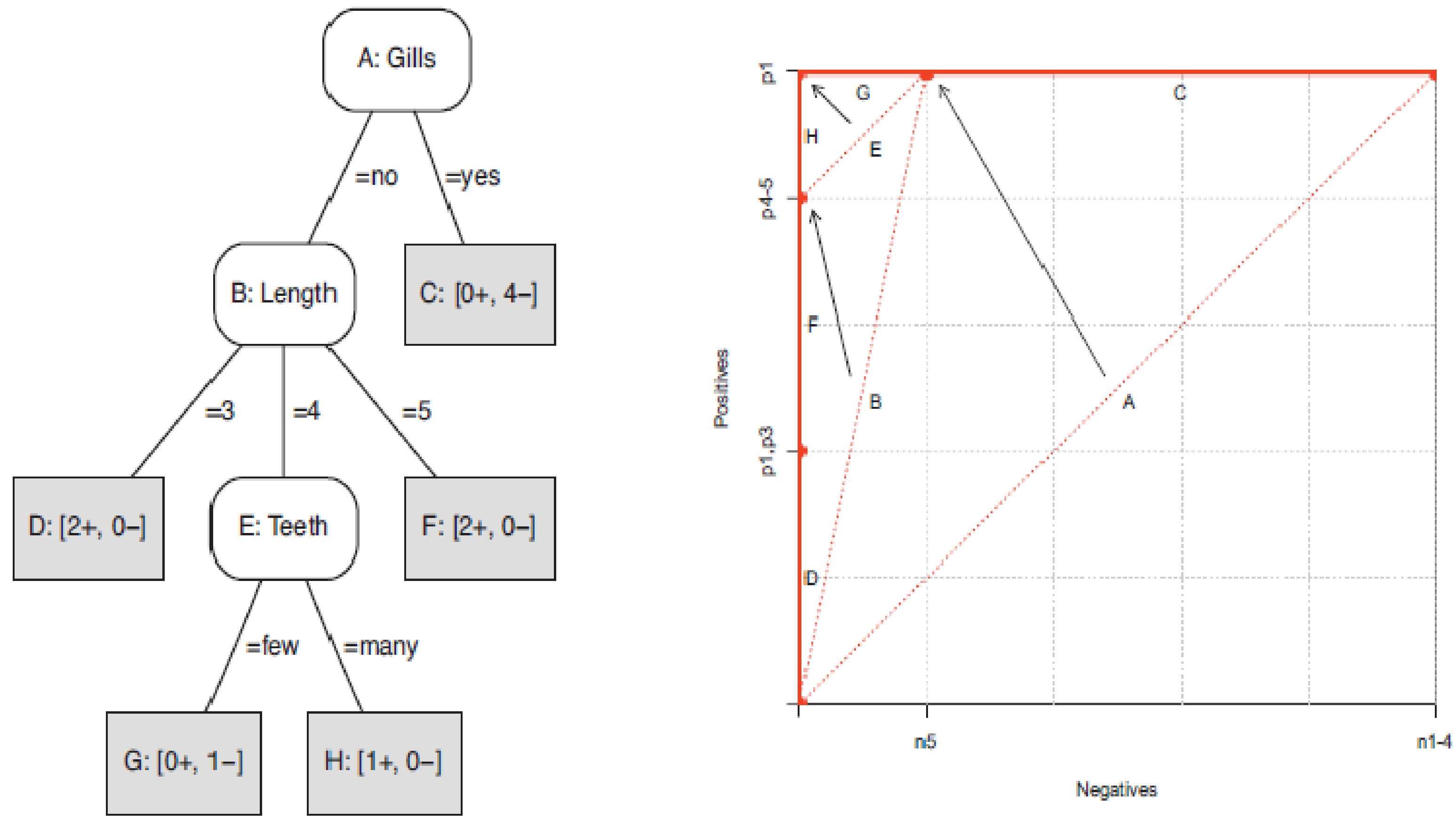


Figure 5.3. **(left)** Decision tree learned from the data in Example 4.4 on p.115. **(right)** Each internal and leaf node of the tree corresponds to a line segment in coverage space: vertical segments for pure positive nodes, horizontal segments for pure negative nodes, and diagonal segments for impure nodes.

Algorithm 5.2: BestSplit-Class(D, F) – find the best split for a decision tree.

Input : data D; set of features F.

Output : feature f to split on.

```
I_min ← 1;
for each f ∈ F do
    split D into subsets D_1,...,D_l according to the values v_j of f;
    if Imp({D_1,...,D_l}) < I_min then
        I_min ← Imp({D_1,...,D_l});
        f_best ← f;
    end
end
return f_best
```