Module 12

Machine Learning

Version 2 CSE IIT, Kharagpur

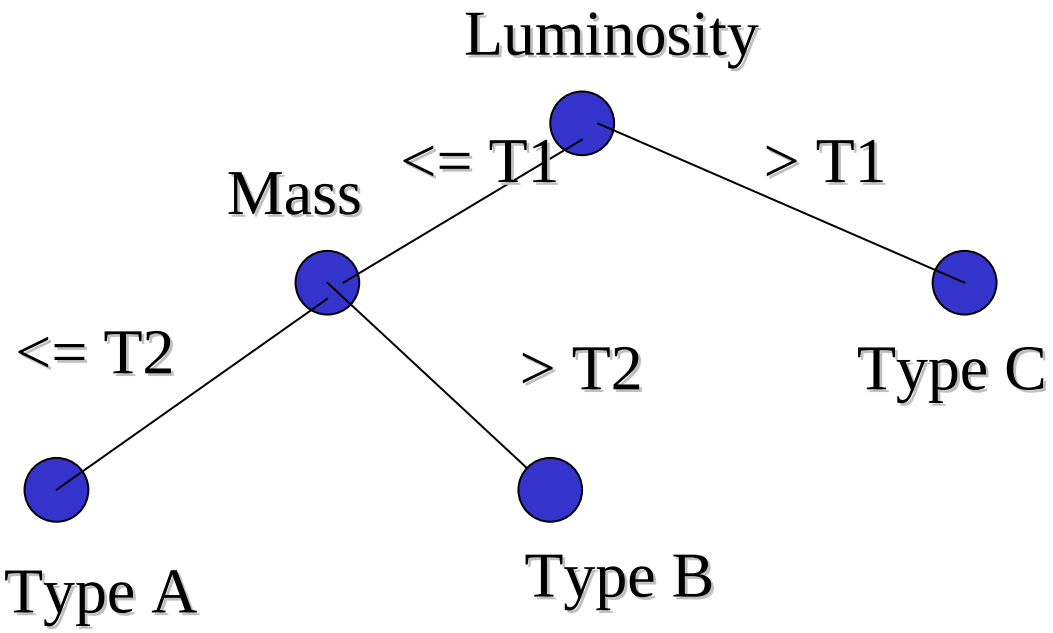
Lesson 35

Rule Induction and Decision Tree - I

Version 2 CSE IIT, Kharagpur

12.3 Decision Trees

Decision trees are a class of learning models that are more robust to noise as well as more powerful as compared to concept learning. Consider the problem of classifying a star based on some astronomical measurements. It can naturally be represented by the following set of decisions on each measurement arranged in a tree like fashion.



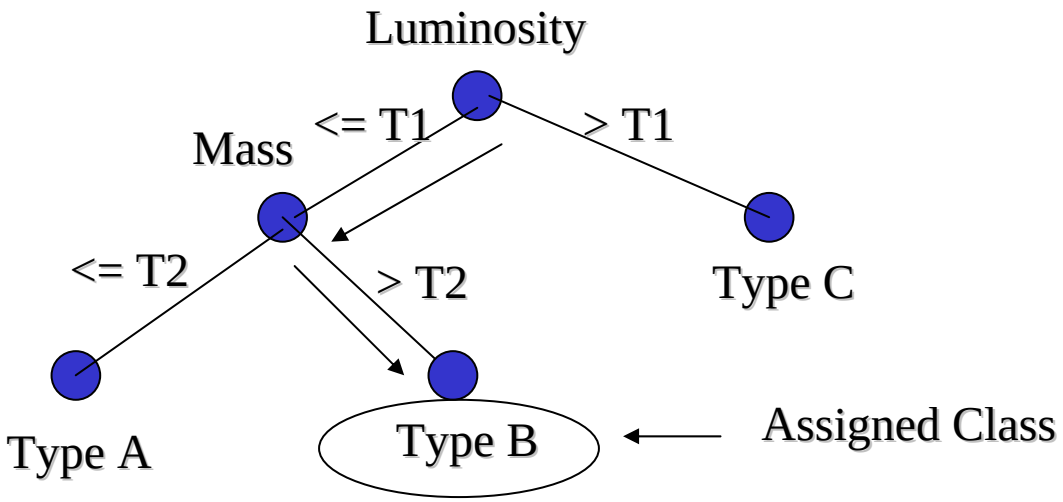
12.3.1 Decision Tree: Definition

- A decision-tree learning algorithm approximates a target concept using a tree representation, where each internal node corresponds to an attribute, and every terminal node corresponds to a class.
- There are two types of nodes:
    - Internal node.- Splits into different branches according to the different values the corresponding attribute can take. Example: luminosity <= T1 or luminosity > T1.
    - Terminal Node.- Decides the class assigned to the example.

12.3.2 Classifying Examples Using Decision Tree

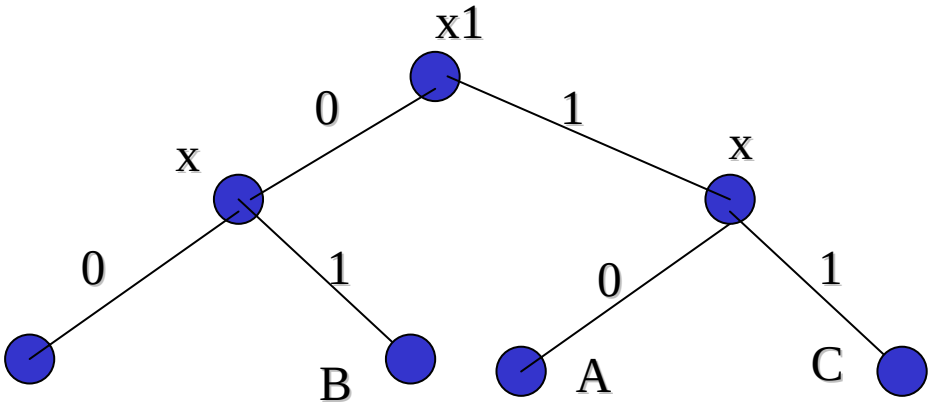
To classify an example X we start at the root of the tree, and check the value of that attribute on X. We follow the branch corresponding to that value and jump to the next node. We continue until we reach a terminal node and take that class as our best prediction.

X = (Luminosity <= T1, Mass > T2)



Decision trees adopt a DNF (Disjunctive Normal Form) representation. For a fixed class, every branch from the root of the tree to a terminal node with that class is a conjunction of attribute values; different branches ending in that class form a disjunction.

In the following example, the rules for class A are: (~X1 & ~x2) OR (X1 & ~x3)



12.3.3 Decision Tree Construction

There are different ways to construct trees from data. We will concentrate on the top-down, greedy search approach:

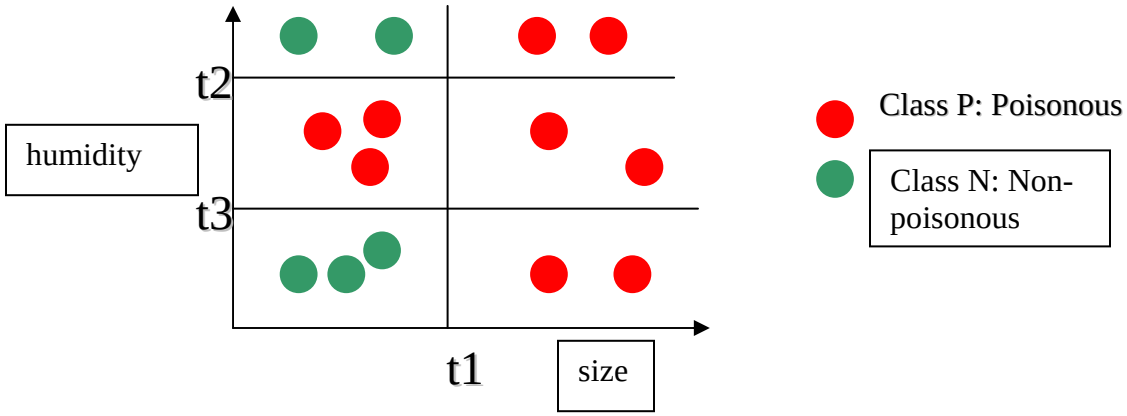
Basic idea:

1. Choose the best attribute a* to place at the root of the tree.

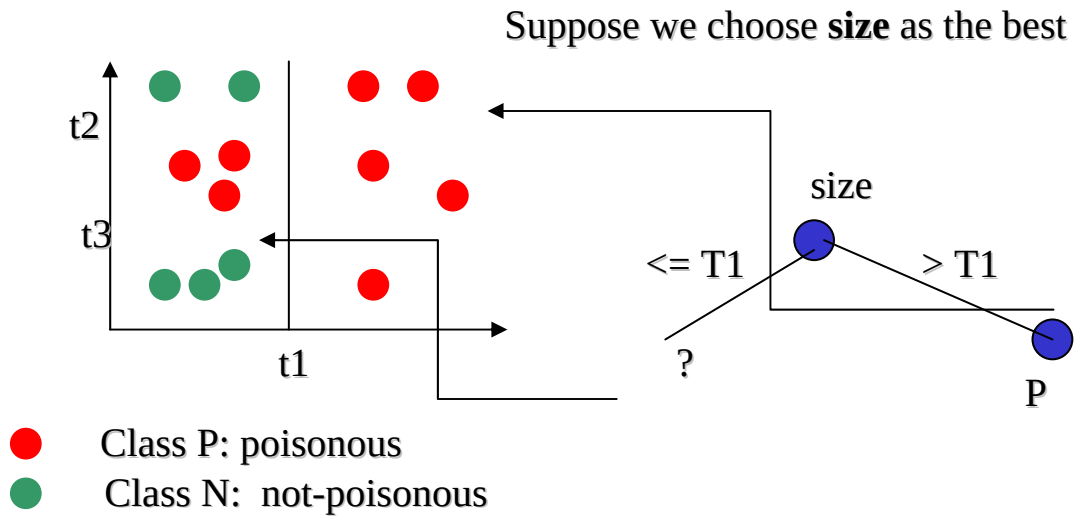
2. Separate training set D into subsets {D1, D2, .., Dk} where each subset Di contains examples having the same value for a*

3. Recursively apply the algorithm on each new subset until examples have the same class or there are few of them.

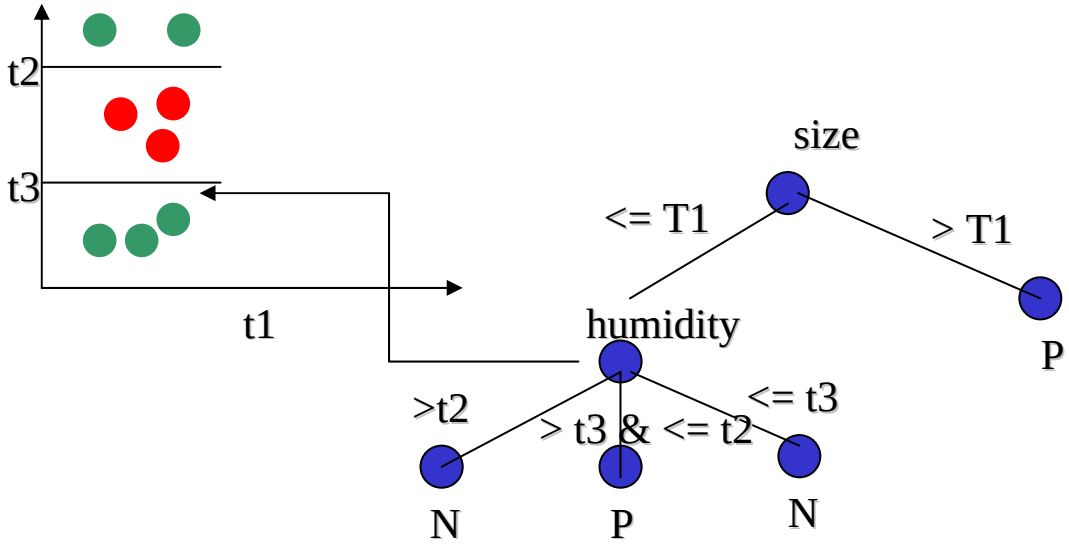
Illustration:



Attributes: size and humidity.
Size has two values: >t1 or <= t1
Humidity has three values: >t2, (>t3 and <=t2), <= t3



Suppose we choose **humidity** as the next best



Steps:

- Create a root for the tree
- If all examples are of the same class or the number of examples is below a threshold return that class
- If no attributes available return majority class
- Let a* be the best attribute
- For each possible value **v** of **a***
    - Add a branch below **a*** labeled "a = v"
    - Let Sv be the subsets of example where attribute a*=v
    - Recursively apply the algorithm to Sv