Notes on Training Decision Trees

Figure 18.5 in Russell & Norvig presents an outline of an algorithm for training a decision tree from examples. This algorithm is designed for the fully-discrete case: a discrete set of classes, with attributes having discrete values. The assignment asks you to train a decision tree with continuous-valued attributes; this document describes changes to the algorithm outline to handle this.

Choosing Attributes
Line 5 calls a CHOOSE-ATTRIBUTE function that uses information gain to pick which attribute to split the tree on. The given formulae for information gain assume discrete-valued attributes. In this case it is acceptable for the assignment to treat the attributes as discrete. You will just calculate information gain over all the values that appear in the training set.

The correct way to do this for a continuous-valued attribute is shown below, but you are not required to use it (although it may actually be easier!).

Splitting the Tree
Line 8 iterates over values of the best attribute chosen by CHOOSE-ATTRIBUTE to train subtrees for each value. In this case, treating a continuous attribute as discrete will not work well, as many values will have just one or two associated examples. Instead, you may discretize the attribute by finding a threshold \( t \) that divides the examples into two groups. Then split the tree into two branches: one for examples with \( \text{best} < t \) and one for examples with \( \text{best} \geq t \).

To choose the threshold for attribute best, I suggest implementing the following procedure:

function CHOOSE-THRESHOLD(best, examples) returns a threshold \( t \)

\[
n \leftarrow \text{total number of remaining examples.}
\]

\[
\text{for each value } v_i \text{ of } \text{best} \text{ do}
\]

\[
A_i \leftarrow \text{the number of remaining examples with } \text{best} < v_i \text{ and } y = \text{setosa}.
\]

\[
B_i \leftarrow \text{the number of remaining examples with } \text{best} < v_i \text{ and } y = \text{versicolor}.
\]

\[
C_i \leftarrow \text{the number of remaining examples with } \text{best} \geq v_i \text{ and } y = \text{setosa}.
\]

\[
D_i \leftarrow \text{the number of remaining examples with } \text{best} \geq v_i \text{ and } y = \text{versicolor}.
\]

\[
acc_i \leftarrow \max\{\frac{A_i + D_i}{n}, \frac{B_i + C_i}{n}\}.
\]

\[
\text{end for}
\]

\[
t \leftarrow \text{value } v_i \text{ that gives maximum } acc_i.
\]

return \( t \)

This finds the value of best that does the best job of splitting examples into two groups that represent the two classes, and uses that value as the threshold. It’s essentially equivalent to the “split point” method briefly described at the end of Section 18.3.

Then modify Figure 18.5 by calling CHOOSE-THRESHOLD before the for loop. The for loop itself is no longer needed; instead, you will execute DECISION-TREE-LEARNING once with the examples that have \( \text{best} < t \) and once with the examples that have \( \text{best} \geq t \).
Continuous Information Gain

For this problem, we want to determine how much information a continuous random variable (one of the attributes) contains about a discrete random variable (the class). Let \( C \) be the class random variable (with domain \{setosa, versicolor\}) and \( A_i \) be the random variable representing attribute \( i \) (with domain \((-\infty, +\infty)\)). We want to calculate information gain \( I(C; A_i) \) for each attribute and take the one with maximum information gain.

The notation I learned and use for entropy and information is somewhat different from the book’s, so I apologize for any resulting confusion (hopefully everything will be clear from context). Information gain is calculated as:

\[
I(C; A_i) = H(A_i) - H(A_i|C)
\]

where \( H(A_i) \) is the entropy of \( A_i \) and \( H(A_i|C) \) is the conditional entropy of \( A_i \) given \( C \). Then:

\[
I(C; A_i) = H(A_i) - H(A_i|C) = H(A_i) - H(A_i|C = setosa)P(C = setosa) - H(A_i|C = versicolor)P(C = versicolor)
\]

To proceed further we have to assume some distributions \( P(A_i), P(A_i|C = setosa), P(A_i|C = versicolor) \). We will assume they are Gaussians. Then \( P(A_i) \) has parameters \( \mu_i, \sigma_i \) (mean and standard deviation) that can be estimated using all examples. \( P(A_i|C = setosa) \) has parameters \( \mu_{i,s}, \sigma_{i,s} \) that can be estimated using all examples of class setosa, and \( P(A_i|C = versicolor) \) has parameters \( \mu_{i,v}, \sigma_{i,v} \) that can be estimated using all examples of class versicolor.

Then, by the definition of entropy of Gaussian distributions, the equation above reduces to:

\[
I(C; A_i) = \ln(\sigma_i\sqrt{2\pi e}) - P(C = setosa)\ln(\sigma_{i,s}\sqrt{2\pi e}) - P(C = versicolor)\ln(\sigma_{i,v}\sqrt{2\pi e})
\]

where \( e \) is the base of the natural log.

The distribution \( P(C) \) can easily be estimated from the remaining examples:

\[
P(C = setosa) = \frac{\text{# of examples with } y_i = setosa}{n}
\]

\[
P(C = versicolor) = \frac{\text{# of examples with } y_i = versicolor}{n}
\]

where \( n \) is the number of remaining examples.

Therefore, implementing \texttt{CHOOSE-ATTRIBUTE} for continuous-valued attributes is just a matter of calculating three standard deviation parameters \( \sigma_i, \sigma_{i,s}, \sigma_{i,v} \) and plugging them into the equation for \( I(C; A_i) \) above along with the distribution \( P(C) \).