Data Mining: Exercises on Classification Trees

Exercise 1: Computing Splits

We want to determine the optimal split in a node that contains the following data:

x_1	a	b	b	b	c	c	d	d	d	e
x_2	28	31	35	40	40	45	45	52	52	60
y	B	B	B	G	B	G	B	G	G	G

Here x_1 is a categorical attribute with possible values {a,b,c,d,e}, x_2 is a numerical attribute, and y is the binary class label. We use the gini index as splitting criterion.

(a) The number of splits for a categorical variable with L distinct values is $2^{L-1} - 1$. So: $2^4 - 1 = 15$.

(b) Sort the values of x_1 on probability of B (or G) and consider all splits between adjacent values in the sorted list. We find $P(B|a) = 1, P(B|b) = \frac{2}{3}, P(B|c) = \frac{1}{2}, P(B|d) = \frac{1}{3}, P(B|e) = 0$. So the splits are: {a},{a,b},{a,b,c},{a,b,c,d}. Note: in our notation, the split {a,b} is the split that sends all cases with $x_1 \in \{a, b\}$ to one child node, and all cases with $x_1 \in \{c, d, e\}$ (the complement of $\{a, b\}$) to the other child node. Note also that it doesn't matter whether we sort on $P(B|x_1)$ or on $P(G|x_1)$, nor whether we sort ascending or descending. In all cases we get the same collection of splits.

(c) The sorted distinct values of x_2 are : 28,31,35,40,45,52,60. So the number of splits is 6 (all splits halfway between 2 adjacent distinct values in the sorted list)

(d) Optimal splits can only occur on segment borders. To determine the segments we merge all values of the split attribute that are adjacent in the sorted list and have the same class distribution (i.e., the same relative frequencies for all classes). This gives the three segments: (28,31,35),(40,45), and (52,60). So we have to evaluate just 2 splits: $x_2 \leq 37.5$ and $x_2 \leq 48.5$.

Exercise 2: Splitting and Pruning

(a) $i(t_1) = \frac{6}{10} \cdot \frac{4}{10} = \frac{24}{100}$, $i(t_2) = \frac{2}{5} \cdot \frac{3}{5} = \frac{24}{100}$, $i(t_3) = \frac{1}{5} \cdot \frac{4}{5} = \frac{16}{100}$

(b) $\Delta i = \frac{24}{100} - \left(\frac{1}{2} \cdot \frac{24}{100} + \frac{1}{2} \cdot \frac{16}{100}\right) = \frac{4}{100}$.

(c) Since we continued splitting until all nodes were pure, we have $T_1 = T_{\max}$.

(d) Rows in the table represent the iterations of the pruning algorithm:

	t_1	t_2	t_3	t_4	t_5	t_7
$g_1(\cdot)$	$\frac{1}{15}$	$\frac{1}{15}$	$\frac{1}{20}^*$	$\frac{1}{20}^*$	$\frac{1}{10}$	$\frac{1}{20}^*$
$g_2(\cdot)$	$\frac{1}{12}$	$\frac{15}{200}^*$	—	—	$\frac{1}{10}$	—
$g_3(\cdot)$	$\frac{1}{10}^*$	—	—	—	—	—

The subscript of g indicates the iteration number. In each iteration the nodes with minimum g values are pruned (indicated with a star in the table). Note that if the pruning hasn't changed T_t, then we don't have to recompute $g(t)$. So we don't need to recompute $g(t_5)$ in the second iteration, because T_{t_5} wasn't changed by the pruning in the first iteration.

Summarizing: T_1 is the smallest minimizing subtree for $\alpha \in [0, \frac{1}{20})$. T_2 is obtained by pruning T_1 in t_4, t_7 and t_3, and it is the best tree for $\alpha \in [\frac{1}{20}, \frac{15}{200})$. T_3 is obtained by pruning T_2 in t_2, and it is the best tree for $\alpha \in [\frac{15}{200}, \frac{1}{10})$. The root node wins for $\alpha \in [\frac{1}{10}, \infty)$.

Note that in the first pruning step, if we prune in t_3 before we prune in t_7, then t_7 will already be pruned away. To avoid this situation, the pruning algorithm specifies that the nodes should be visited in top-down (depth-first) order. It depends on how you implement the pruning whether this really is an issue.

Exercise 3: Splitting on a numeric attribute

In the best case, all examples of one class precede all examples of the other class in the sorted order. In that case we only have to evaluate one split, and it would be a very good one!

In the worst case the class label changes every time in the sorted order, so that we still have to evaluate all possible splits. The resulting split will not be very good either.

In general, if we have k classes we have to evaluate just $k - 1$ splits in the best case. In the worst case, we have to evaluate $n - 1$ splits, where n is the number of observations in the node concerned.

Exercise 4: Gini index

We have defined the gini index for binary classification as

$$i(t) = p(0|t)p(1|t) = p(0|t)(1 - p(0|t)), \tag{1}$$

where the class values are coded as 0 and 1, and $p(j|t)$ denotes the relative frequency of class j in node t. The generalization to an arbitrary number of classes is given by:

$$i(t) = \sum_{j=1}^{C} p(j|t)(1 - p(j|t)), \tag{2}$$

where C denotes the number of classes. If we apply equation (2) to the binary case, we should get the same results as when we apply equation (1). Is this indeed the case?

Call impurity according to equation (1) i_1, and according to equation (2) i_2. For the binary case, we have

$$i_2(t) = p(0|t)(1 - p(0|t)) + p(1|t)(1 - p(1|t)) = 2\, p(0|t)(1 - p(0|t)) = 2\, i_1(t)$$

So impurity according to equation (2) is twice as large as according to equation (1). Therefore, also $\Delta i_2 = 2\, \Delta i_1$. This makes no difference in determining the optimal split because it is only the order of the values that matters. The same split will win.

Show that equation (2) can alternatively be written as

$$i(t) = 1 - \sum_{j=1}^{C} p(j|t)^2.$$

Starting with (2):

$$\begin{aligned} i(t) &= \sum_{j=1}^{C} p(j|t)(1 - p(j|t)) \\ &= \sum_{j=1}^{C} p(j|t) - p(j|t)^2 \\ &= \sum_{j=1}^{C} p(j|t) - \sum_{j=1}^{C} p(j|t)^2 \\ &= 1 - \sum_{j=1}^{C} p(j|t)^2. \end{aligned}$$