DS102 - Discussion 11

Wednesday, 20th November, 2019

1. **Gini Impurity.** The goal in building a decision tree is to create the smallest possible tree in which each leaf node contains training data from only one class. In evaluating possible splits, it is useful to have a way of measuring the "purity" of a node. Purity describes how close the node is to containing data from only one class, and there are different ways of measuring it. Intuitively, we want to make nodes as pure as possible after only a few splits. One standard way of measuring purity is *Gini purity*, defined as:

$$\phi(\mathbf{p}) = \sum_{i=1}^{n} p_i(1 - p_i),$$

where $\mathbf{p} = (p_1, \ldots, p_n)$ and each p_i is the fraction of elements from class i. This expresses the fractions of incorrect predictions in the node if the class of each element was predicted by randomly selecting a label according to the distribution of classes in the node. This value will be 0 if all elements are from the same class, and it increases as the mix becomes more uniform. Calculate the Gini impurity of the following binary data set:

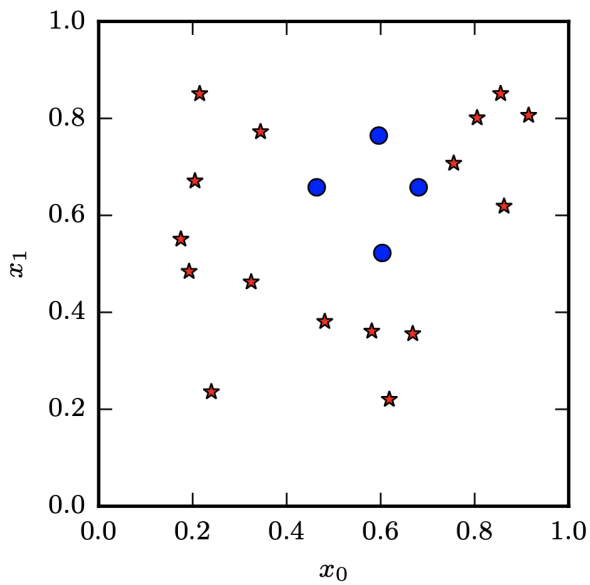

Figure 1: A binary data set.

2. **Tree Construction.** The decision tree construction algorithm proceeds by recursively splitting the training data into increasingly smaller subsets. When splitting a node in the tree we search across all dimensions and all split points to select the split that results in the greatest decrease in impurity. This goodness-of-split value can be expressed as:

$$\theta(s, N) = \phi(\mathbf{p}) - P_L\phi(\mathbf{p}_L) - P_R\phi(\mathbf{p}_R),$$

where $s = \{L, R\}$ is a possible split of data points into two subsets L and R, N is the current node and P_L and P_R represent the fraction of elements that would end up in the left child and right child, respectively. Higher values of $\theta(s, N)$ represent better splits. Execute the recursive tree-construction algorithm on the data above and draw the resulting tree. Calculate the impurity of each node and the goodness-of-split for each split. For simplicity, at each node we consider only a binary split according to one feature; for example, a valid split would be $L = \{\text{points with } x_0 < 0.1\}$ and $R = \{\text{points with } x_0 \geq 0.1\}$, but *not* $L = \{\text{points with } x_0 < 0.1, x_1 > 0.5\}$ and $R = \{\text{points with } x_0 \geq 0.1, x_1 \leq 0.5\}$.

3. **Tree Diagram.** Draw the resulting decision tree.

4. **Classifying Fresh Samples.** Classify the following three points using your decision tree:

$$(x_0, x_1) = (0.3, 1.0), \quad (x_0, x_1) = (0.6, 1.0), \quad (x_0, x_1) = (0.6, 0).$$