Decision Trees: Part 2

Nikhil Sardana

September 2017

1 Introduction

Part 1 of this lecture covered the mathematics of decision trees. This lecture will transition you from graphs and mathematics to writing the actual code for the competition.

2 Review

Decision trees are created by splitting on a threshold of a feature which results in the most information gain, and recursively continuing this process for each of the children.

Let us review Information Gain with an example. Remember, the equation is:

$$IG(D_p, f) = I(D_p) - \frac{N_{left}}{N_p} I(D_{left}) - \frac{N_{right}}{N_p} I(D_{right})$$

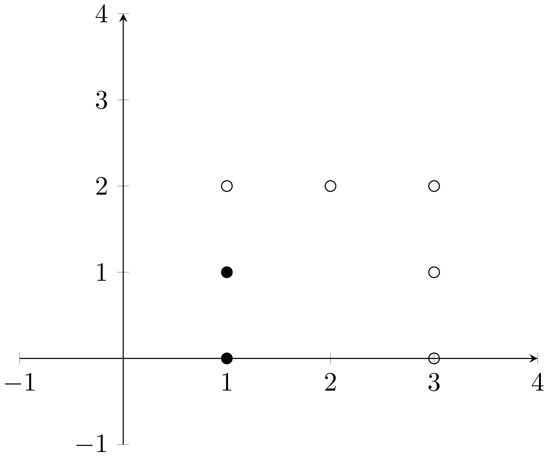
We will use the Gini Impurity measure:

$$I(i) = 1 - \sum_{k=1}^{c} p(k|i)^2$$

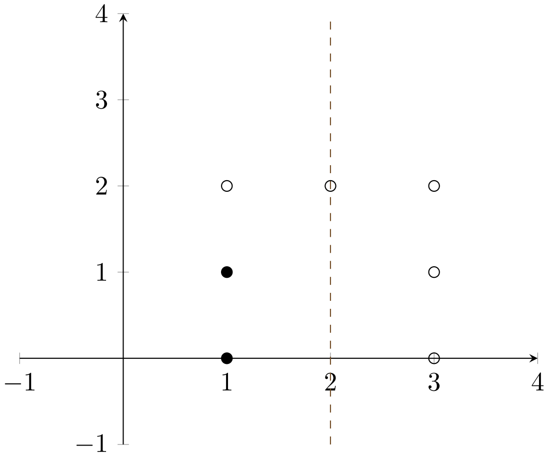
Consider the following data. We have two features, $x1$ and $x2$. Our label is y.

x1	x2	y
1	0	0
1	1	0
1	2	1
2	2	1
3	0	1
3	1	1
3	2	1

Let's graph this data. Let the ● = 0 and ○ = 1.

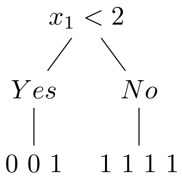


We want to draw a vertical or horizontal line that best separates the data (ie has the greatest information gain). Since we are restricted to the coordinates of data points, a human will look at the graph and choose:



Since we are classifying things based on whether they are less than the threshold ($x < 2$), the point in the middle goes with the right side.

The decision tree that goes with this is:



We can calculate the information gain.

$$IG(D_p, f) = I(D_p) - \frac{N_{left}}{N_p} I(D_{left}) - \frac{N_{right}}{N_p} I(D_{right})$$

$$N_{left} = 3$$

$$N_{right} = 4$$

$$N_p = 7$$

Now we need the impurities.

$$I(i) = 1 - \sum_{k=1}^{c} p(k|i)^2$$

For the parent, we calculate the impurity of the entire dataset, which is:

$$I(D_p) = 1 - \left(\frac{2}{7}\right)^2 - \left(\frac{5}{7}\right)^2 = 1 - \frac{4}{49} - \frac{25}{49} = \frac{49 - 4 - 25}{49} = \frac{20}{49}$$

And for the children:

$$I(D_{left}) = 1 - \left(\frac{2}{3}\right)^2 - \left(\frac{1}{3}\right)^2 = 1 - \frac{4}{9} - \frac{1}{9} = \frac{4}{9}$$

$$I(D_{right}) = 1 - \left(\frac{4}{4}\right)^2 - \left(\frac{0}{7}\right)^2 = 1 - 1 = 0$$

Note that the most pure data $I(i) = 0$ is all one type.
Finally, we get the information gain:

$$IG(D_p, f) = I(D_p) - \frac{N_{left}}{N_p} I(D_{left}) - \frac{N_{right}}{N_p} I(D_{right})$$

$$IG(D_p, f) = \frac{20}{49} - \frac{3}{7} * \frac{4}{9} - \frac{4}{7} * 0 = \frac{20}{49} - \frac{12}{63} = \frac{1}{7} * \left(\frac{20}{7} - \frac{12}{9}\right) = \frac{1}{7} * \left(\frac{180}{63} - \frac{84}{63}\right)$$

$$IG(D_p, f) = \frac{1}{7} * \frac{96}{63} = \frac{96}{441}$$

3 Data Structure

Let's continue looking at this example.

x1	x2	y
1	0	0
1	1	0
1	2	1
2	2	1
3	0	1
3	1	1
3	2	1

How will we represent this data in our code? It's simple! We'll use a list of lists, with each column a feature.

$$\begin{bmatrix}[1,0,0]\\ [1,1,0]\\ [1,2,1]\\ [2,2,1]\\ [3,0,1]\\ [3,1,1]\\ [3,2,1]\end{bmatrix}$$

To find the best information gain, we loop through each of the features (columns), and each of the thresholds, splitting the matrix and calculating the information gain. For example, in the previous example, we split on $x1 < 2$, so our left child matrix would be:

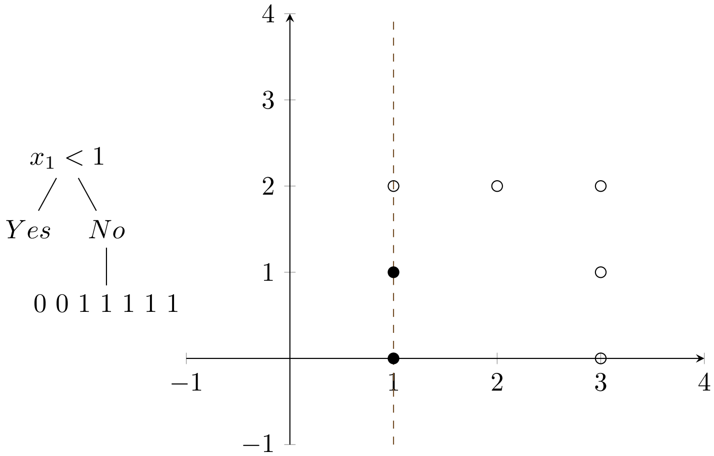
$$\begin{bmatrix}[1,0,0]\\ [1,1,0]\\ [1,2,1]\end{bmatrix}$$

and the right child matrix:

$$\begin{bmatrix}[2,2,1]\\ [3,0,1]\\ [3,1,1]\\ [3,2,1]\end{bmatrix}$$

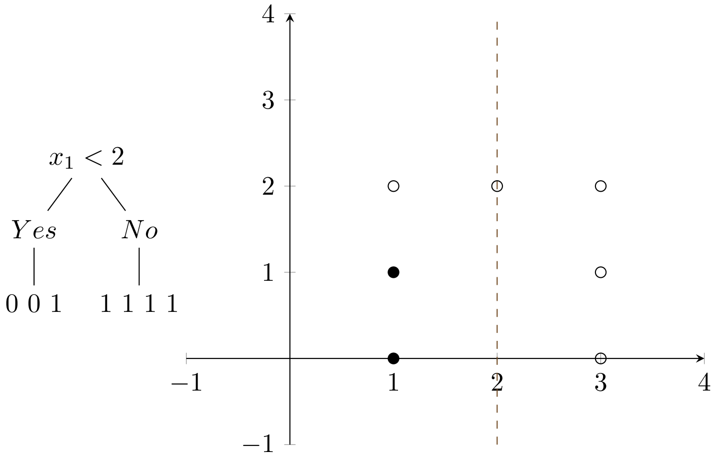
4 Summary

To ensure complete understanding and summarize the topics we've covered, lets loop through all the features and each of the possible thresholds, splitting the matrix for each one.



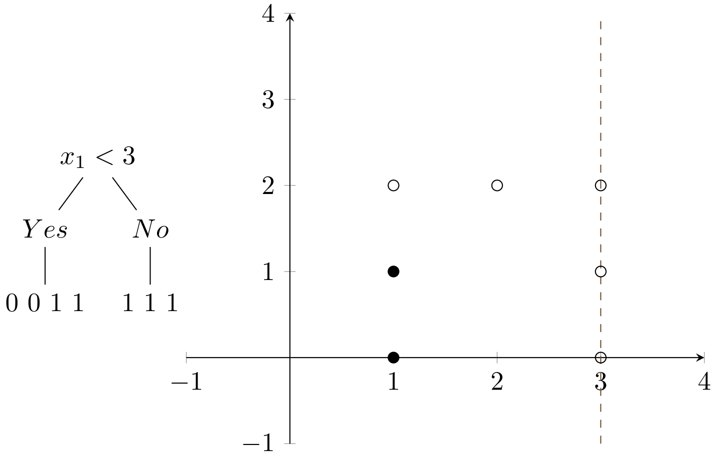
$$D_{left} = []$$

$$D_{right} = \begin{bmatrix} [1,0,0] \\ [1,1,0] \\ [1,2,1] \\ [2,2,1] \\ [3,0,1] \\ [3,1,1] \\ [3,2,1] \end{bmatrix}$$



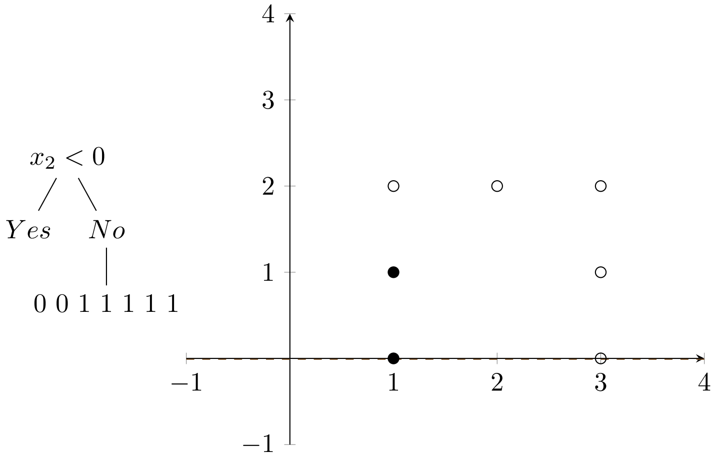
$$D_{left} = \begin{bmatrix} [1,0,0] \\ [1,1,0] \\ [1,2,1] \end{bmatrix}$$

$$D_{right} = \begin{bmatrix} [2,2,1] \\ [3,0,1] \\ [3,1,1] \\ [3,2,1] \end{bmatrix}$$



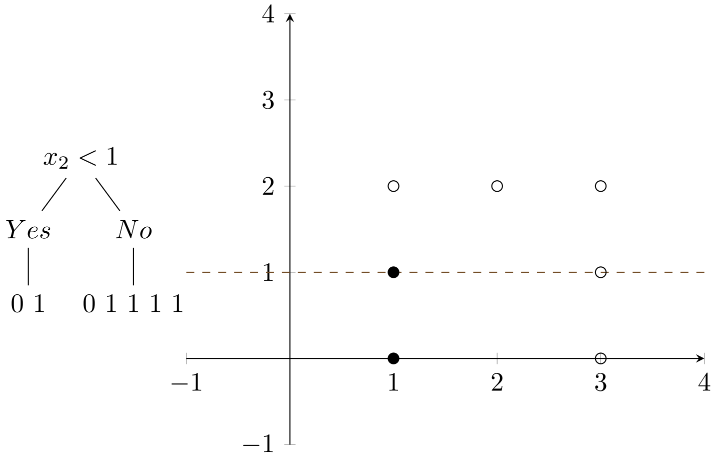
$$D_{left} = \begin{bmatrix} [1,0,0] \\ [1,1,0] \\ [1,2,1] \\ [2,2,1] \end{bmatrix}$$

$$D_{right} = \begin{bmatrix} [3,0,1] \\ [3,1,1] \\ [3,2,1] \end{bmatrix}$$



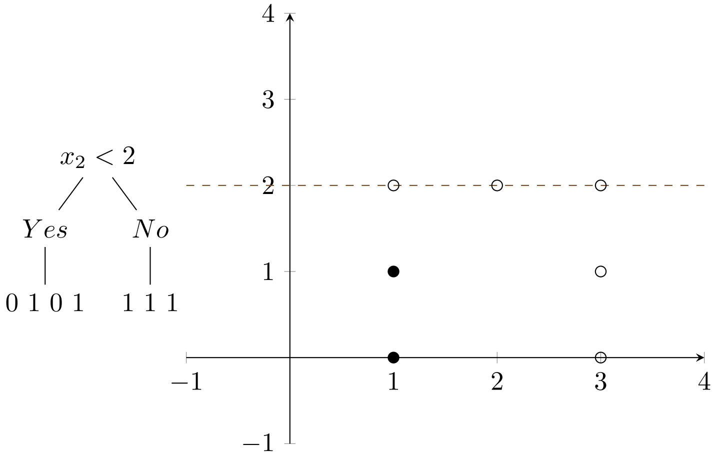
$$D_{left} = []$$

$$D_{right} = \begin{bmatrix} [1,0,0] \\ [1,1,0] \\ [1,2,1] \\ [2,2,1] \\ [3,0,1] \\ [3,1,1] \\ [3,2,1] \end{bmatrix}$$



$$D_{left} = \begin{bmatrix} [1,0,0] \\ [3,0,1] \end{bmatrix}$$

$$D_{right} = \begin{bmatrix} [1,1,0] \\ [1,2,1] \\ [2,2,1] \\ [3,1,1] \\ [3,2,1] \end{bmatrix}$$



$$D_{left} = \begin{bmatrix} [1,0,0] \\ [3,0,1] \\ [1,1,0] \\ [3,1,1] \end{bmatrix}$$

$$D_{right} = \begin{bmatrix} [1,2,1] \\ [2,2,1] \\ [3,2,1] \end{bmatrix}$$

5 Code

Here's how we recommend you structure your decision tree code for the competition. It should be noted: do not split recursively until all data is pure. You will overfit the testing data. Instead, set a baseline for your information gain. If the maximum information gain is less than, say, 0.1, then stop the recursion and make it a leaf of the tree. Play with this baseline to see how your training and testing accuracy change. Remember: your goal is the greatest testing accuracy.

```
def calculateImpurity(matrix):
    #do calculations
    return impurity

def splitmatrix(matrix, feature, threshold):

    #split matrix into leftchild, rightchild
    return leftchild, rightchild

def informationGain(parent, leftchild, rightchild):
    #do calculations
    return infoGain

def bestsplit(parent, depth):
    for each feature in parent:
        for each threshold in feature:
            lchild, rchild = splitmatrix(parent, feature, threshold)
            igain = informationGain(parent, lchild, rchild)

            #if igain is the greatest:
            #    save the leftchild, rightchild, feature, threshold

    #print this node (feature, threshold) of the decision tree
    # hint: use depth to indent
    #recur on leftchild
    #recur on rightchild
```

6 Practice Problems

1. Calculate the information gain for each of the 7 decision trees in Section 4. The second decision tree has already been done for you in the example in Section 2.

2. Do the competition.