INFX574 Lab 6: Branching a classification tree

Deadline: Wed, May 16th, 11:59pm

Introduction

Please submit the completed lab by end of the day. You should submit a) your code (notebooks or whatever you are using) and b) the lab in a final output form (html or pdf).

Please do not just provide computer output. Always comment your main findings. Include any substantial comments as a separate text blocks. Also limit your output: do not submit pages and pages of whatever your code spits out.

Note: you may want to do some of it on paper instead of computer. You are welcome to do this but please include the result as an image in your final file.

Working together is fun and useful but you have to submit your own work. Discussing the solutions and problems with your classmates is all right but do not copy-paste their solution! Please list all your collaborators below:

1.

2. ...

How to branch a decision tree

Your task is to implement a little bit of classification tree algorithm. You have to find the variable that is the best choice for the initial branch.

1 Data

The data *house-votes-84* (see canvas/files/data) contains votes for 16 bills by 435 Representatives in 1984 (see the included readme file). The first variable is the party memebership (*republican* or *democrat*), and the following 16 features are votes (*y*, *n*, or *?* if there was neither yea nor nay vote).

1. Load data. Note: the file does not contain the header line.
2. Explore the data: What is the number of yeas, nays and others by the column.
3. Compute the percentage of democrats and republicans in your data.

2 Which variable gives the best branch

Our aim is to classify the party membership (D or R) using the voting data. Let's ignore overfitting and related issues and use the whole dataset for training.

There are votes for 16 bills you can use. Which one gives the best split? Let's find it out.

1. Pick the first attribute (which happens to be voting for handicapped infants bill, see the readme file). Split your data according to yea or nay vote on that bill. You can just ignore the other here. You get two subsets of data: yea-sayers and nay-sayers.

2. Compute the percentage of republicans and democrats in both subsets.

Intuitively, the more clearly the feature distinguishes between republicans and democrats, the better. But it splits the data into two groups, and it may well be the case that while one group is almost exclusively from a single party, the other group is a 50-50 mix. How can we compare such branches and decide which one is better?

One of the most popular measures to address this problem is *entropy*. For a single branch the entropy can be calculated as

$$\mathsf{H} = -\mathsf{p}^{\mathsf{D}} \log_2 \mathsf{p}^{\mathsf{D}} - \mathsf{p}^{\mathsf{R}} \log_2 \mathsf{p}^{\mathsf{R}} \tag{0.1}$$

where p^{D} is the percentage of democrats and p^{R} the percentage of republicans in that branch. For instance, if we have 40% republicans and 60% democrats, the entropy will by

$$\mathsf{H} = -0.6 \log_2 0.6 - 0.4 \log_2 0.4 = 0.971.$$

3. Compute entropy for both subsets (yea-voters and nay-voters).

But we still have two groups and hence two entropies for each branch. How to deal with this? Fortunately it is intuitive: just compute the weighted average entropy, where the weights are the corresponding group sizes (number of yea-voters and nay-voters).

4. Compute the final entropy of this split.

So far we have struggled to come up with a measure of split purity (entropy) for the first feature. But we have 16 features!

5. Compute the weighted average entropy for all the features.

6. Which feature will give the smallest entropy? This is the best split!

If you have more time and iterest, you can continue the same algorithm for both subsets, and subsets-of-subsets and so on to build a complete classification tree, or alternatively use sklearn *DecisionTreeClassifier* to compare it's suggestion with your finding. Does it start splitting at the same feature?