Decision Trees

Introduction

Decision trees are a method of classifying data. Computers can develop these classifiers by exhaustively searching for the point at which to divide a certain variable by comparing the resulting information gain. For this project, we implemented a function to create 1-rule decision trees, and also used WEKA to generate more complicated trees.

1-R Decision Trees

One-rule trees consider only one independent variable in order to predict a dependent variable. Stars are classified in stages based on three variables: LogMDOT, LogMDISK, and LogMASSC. Ratios of these values are used by experts, but we were interested to see how well the computer could approximate this relation using 1-R stumps. There are three stages of stars represented in the data, so the algorithm finds the pair of numbers on which to divide the data into three parts. The possible break points are values for which the point directly less of a different class from the point directly greater. To find these points, I created a list of pairs of the independent value and the class, and sorted it by the independent value. Any values with the same class or value as the one directly before it is removed from the list of splitting points. If the independent variable can correctly estimate the class, then this process significantly lowers the possible number of split points since most instances of a particular class will neighbor each other.
Once these points are determined, a double-nested for-loop evaluates the information gained by splitting on each possible pair. Information gain is defined as the difference between the entropy before the split and the sum of the entropies of each of the branches, weighted by the proportion of values that went down the branch.
For one dataset, we found the following information gains and error rates (the spread of classes for the entire set is 19, 1284, 1540):
LogMDOT: x<-8.5379 ==> stage 3
x<-6.2443 ==> stage 2
x>=-6.2443==> stage 1
Info gain: -.6411
Error rate: .4295
Confusion Matrix: 1 2 3
[0, .4422, .5578] (0,1221,1540)
[0, 1 , 0] (0,63,0)
[1, 0 , 0] (19,0,0)

LogMDISK: x<-6.4132 ==> stage 3
x<-6.0130 ==> stage 2
x>=-6.0130==> stage 3
Info gain: -.6802
Error rate: .0341
Confusion Matrix: 1 2 3
[0 , 0 , 1] (0,0,1447)
[0 , .4545, .5454] (0,75,90)
[.0154, .9821, .0024] (19,1209,3)

LogMASSC: x<-.6326 ==> stage 2
x<-.1011 ==> stage 3
x>=-.1011==> stage 2
Info gain: -.6965
Error rate: .4077
Confusion Matrix: 1 2 3
[.1097, .8065, .0839] (17,125,13)
[.0006, .3794, .6200] (1,528,496)
[.0010, .5151, .4839] (1,631,1031)

After each stump was built, the program had all of the previous stumps vote for a class for each data point. The voting was simply adding together the confusion vectors for the branch of each tree the point took.
After LogMDOT and LogMDISK, there was a classification error rate of .02743.
When LogMASSC was added, the error rate rose to .03412.

Looking at the error rates, it is clear that MDISK performs the best of these three. The combination of MDOT and MDISK, however, is better than either on its own. Adding MASSC does not help, which makes some sense because the expert classification does not use it for stage 1, so its presence likely adversely affects those stars.

WEKA Decision Trees

We also wanted to classify sources using a more sophisticated decision tree. The actual data that is obtained from observations is the radiant flux at a particular wavelength. By comparing the ratios of fluxes at different wavelengths, sources can be classified into their different stages of evolution: stage I, II and III. When analyzing the data, fluxes were converted into magnitudes. Magnitudes are based on the log of the flux so in order to compare ratios of fluxes we actually compared the difference of magnitudes.

The training set that we used was from a library of SED models. These models simulate young stars and return a spectral energy distribution (SED; a plot of intensity vs. wavelength). The SED can be used to determine the magnitudes at specific wavelengths. The input to these models is a list of physical parameters of the source that it is modelling. Because we already know what the values are we also know what stage of evolution the source is at. So what we did to create the training set find the magnitude of the source at four wavelengths in which we have observations of real data in. We then took all the combinations of differences between these wavelengths. The size of the training set is around 50,000. The resulting decision tree is very complicated and too lengthy to follow all the way through.

We used two different test sets to evaluate how well the classification worked. The first test set was just a different subset of the models like the ones we used for training. The results from this were quite good. The decision tree was able to classify almost 80% correctly. The second test set was based on real data which had been classified previously by a different method. The results of this were not as good at around 50% classified correctly. However, this may reflect the quality of the previous classifications more than the quality of the decision tree. Additionally, there are a few subtleties involved such as interstellar extinction which will likely change the results slightly. Regardless, a 50% overlap between these classification techniques is still exciting. This is much better than what was found from previous comparisons.