ASTR4260: Problem Set #8 - OPTIONAL

Due: Wednesday, April 15 (if you choose to do it)

Both problems below refer to a data set composed of observations of a set of stars, in particular for each star, four "colors" are provided – these are the $\mathbf{x}_i$ data points. In astronomy, color refers to the ratio of the brightness of a star observed in light with two different wavelengths (the name comes from the fact that this is similar to what the eye does in detecting color). Some of these stars are known to be in a class of stars known as RR Lyrae stars, which are variable stars (i.e. their luminosity varies regularly over time) that can be used to estimate the distance to an object. It can be challenging to identify RR Lyrae stars, as some of their properties overlap with regular stars. The idea of this project is to use two machine learning techniques to see how well we can identify the RR Lyrae stars out of a larger population of regular (non-RR Lyrae) stars based on a single set of observed colors. The label indicating the type of star are the y_i values that we are trying to predict with the model; here one of two classes, so this is a *supervised classification* problem.

In the Github classroom repository for this problem set, I provide two data files, one of which ("RRLyrae_features.txt") contains four sets of color measurements for each of 93,141 stars, and the second ("RRLyrae_labels.txt") contains a label indicating if the star is (1) or is not (0) a RRLyra. I also include a starter notebook that reads these datafiles into a set of numpy arrays. Thanks to Viviana Acquiviva and Jake van der Plas for these data. Note that they are already in form which is appropriate for use in machine learning (their means and standard deviations are sufficiently similar that they can be sensibly compared).

Problem 1

Use a decision tree classifier to classify this data using the colors as a predictor for the label. In other words, given $\mathbf{x}_i$, predict y_i using a tree. This could be done, for example, using scikit learn (e.g. one could use `sklearn.tree.DecisonTreeClassifier()`. Using 5-fold validation, compute some measure of the precision (e.g. validation score) in computing the accuracy of the prediction. Note that most stars are *not* RR Lyrae, so a simple accuracy estimator may be misleading. More concretely, since less than 1% of the stars are RR Lyrae, a simple predictor is to say that none of the stars are RR Lyrae, which is better than 99% accurate in predicting the label for all stars. However, obviously this is not a very useful way to look at it – a 2x2 table showing True Positive, True Negative, False Positive, and False Negative may be useful here.

I also encourage you to plot the data in some way – for example, you could plot x,y as two of the four "colors" and color-code the points in the plot using the label. That will show if there are obvious decision tree cuts that split the data set into parts. You may also want to plot the decision tree itself (or the cuts it arrived at to make those decisions), although that will require more work.

Problem 2

Repeat problem 1 with an SVM classifier (for example, scikit's `sklearn.svm.SVC()` is a reasonable choice). Repeat the 5-fold training and validation and compare accuracy. Which is better (and include some thoughts on why)?