ASTR4260: Problem Set #8

Due: Wednesday, November 22, 2023

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.

I provided two data files on Canvas: `RRLyrae_features.txt` and `RRLyrae_labels.txt`. The first one contains four sets of color measurements for each of 93,141 stars, and the second contains a label indicating if the star is (1) or is not (0) a RRLyra. Thanks to Viviana Acquaviva (CUNY) and Jake van der Plas (Google) for these data. Note that they are already in a form that is appropriate for use in machine learning (their means and standard deviations are sufficiently similar that they can be sensibly compared).

To read these files, you may use the following code:

```
import numpy as np

def read_data(name):
    X = np.genfromtxt(name+'_features.txt', delimiter=',')
    Y = np.genfromtxt(name+'_labels.txt')
    return X, Y

name = "RRLyrae"
X, Y = read_data(name)
```

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 (`cross_val_score` with `cv = 5`), compute the accuracy of the prediction (e.g. the validation score). Note that most stars are *not* RR Lyrae, so this 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 truth table showing what stars are identified as what will be useful here.

Two parameters that you should try out in order to improve your result are the maximum depth of the decision tree, and the weighting of the classes. These are both keyword parameters that can be set. Try ``balanced'' for the class weight. Do either of these improve the result? What does an improved result look like for an observer trying to find RR Lyrae stars by taking as few spectra as possible of stars in this data set?

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, as well as examining truth tables. Does changing the kernel change your result? Again use balanced weighting to take into account the rarity of RR Lyrae in the data set if the results are unsatisfactory. Which methods are better (include some thoughts on why)?