Decision Tree Mini Project

In this project, we will again try to classify emails, this time using a decision tree. The starter code is in `decision_tree/dt_author_id.py`.

Part 1: Get the Decision Tree Running
Get the decision tree up and running as a classifier, setting `min_samples_split=40`. It will probably take a while to train. What's the accuracy?

Part 2: Speed It Up
You found in the SVM mini-project that the parameter tune can significantly speed up the training time of a machine learning algorithm. A general rule is that the parameters can tune the complexity of the algorithm, with more complex algorithms generally running more slowly.

Another way to control the complexity of an algorithm is via the number of features that you use in training/testing. The more features the algorithm has available, the more potential there is for a complex fit. We will explore this in detail in the "Feature Selection" lesson, but you'll get a sneak preview now.

- find the number of features in your data. The data is organized into a numpy array where the number of rows is the number of data points and the number of columns is the number of features; so to extract this number, use a line of code like `len(features_train[0])`
- go into `tools/email_preprocess.py`, and find the line of code that looks like this: `selector = SelectPercentile(f_classif, percentile=1)` Change `percentile` from 10 to 1.
- What's the number of features now?
- What do you think SelectPercentile is doing? Would a large value for percentile lead to a more complex or less complex decision tree, all other things being equal?
- Note the difference in training time depending on the number of features.
- What's the accuracy when percentile = 1?