DSC 140A - "Super Homework"

Due: Wednesday, March 22

Problem 1. (Competition)

This problem is a chance to review the methods learned since the second midterm, including the probabilistic methods and decision trees. It is also a chance to earn extra credit through a competition, similar to Homework 06.

Data. At the link below is a data set of 427 data points in $\mathbb{R}^{19}$.

`https://f000.backblazeb2.com/file/jeldridge-data/005-competition/X-train.csv`

The file has 19 columns: one for each feature. The labels can be found in the separate file:

`https://f000.backblazeb2.com/file/jeldridge-data/005-competition/y-train.csv`

Note that neither file contains column headers, and that the labels in `y-train.csv` range from 0 to 4; that is, there are *five* different classes.

At the next link is a *test* data set of over 143 data points in $\mathbb{R}^{19}$:

`https://f000.backblazeb2.com/file/jeldridge-data/005-competition/X-test.csv`

Task. Implement a machine learning algorithm – any algorithm – that has been discussed in DSC 140A *from Lecture 10 on*. This includes all of the probabilistic methods (LDA, QDA, Naïve Bayes, etc.) as well as decision trees and boosting. Train the algorithm on the training set, and make predictions on the test set.

You can use whatever programming language you like, but you should not use any machine learning packages, such as `sklearn`. Instead, you should implement the method "from scratch". You may use numerical packages, such as `numpy` and `scipy`, however.

Submitting your results. There are two Gradescope submissions for this homework. First, there is "Super Homework - Predictions". Upload your predictions to this submission's autograder as a file named `predictions.csv` containing one prediction (an integer between 0 and 4, inclusive) per line. The autograder will check the format of your submission for correctness, so be sure to wait to see its output.

Second, there is the submission named "Super Homework". Upload your code for this problem (as well as your answer to the second problem below) to this submission as a PDF; we will look at your code, but not run it.

Grading. Recall that the test set does not have labels – we have kept them a secret. At grading time, we will test your predictions against the true labels. To get full credit, at least 40% of your predictions should be correct.

Note: although the training and test sets are slightly unbalanced (there are more instances of some classes than others), we will compute the overall (unweighted) accuracy in order to evaluate your predictions. To get full credit you should get 40% of the labels on the test set correct. You can assume that the label distribution in the test set is essentially the same as in the training set.

Competition. For fun, we'll give out extra credit to people who achieve the top prediction performance on the unseen data. If your prediction performance is in one of the categories below, you'll earn the stated amount of extra credit:

- **Greater than 44% accuracy on the test set**: 1 point
- **Greater than 48% accuracy on the test set**: 2 points
- **Top 10%** 3 points

- **Top accuracy:** 6 points

Hints. Some methods are easier to implement than others. You should be able to achieve full credit by implementing simpler methods.

Also, this is a multiclass classification problem (there are more than two classes). Probabilistic methods based on the Bayes classifier naturally generalize to multiclass classification. As always, they predict the class y which maximizes $p(x|Y = y)\mathbb{P}(Y = y)$, whether there are two possibilities for y (binary classification) or many.

Problem 2.

Suppose you are training a decision tree classifier to predict whether it will rain in the next hour or not. To do so, you will use two features: the current temperature (in Fahrenheit) and the current pressure (in millibars). Here is some training data that you have collected:

Temperature	**Pressure**	**Rain?**
65	1001	Yes
72	1003	Yes
79	1030	No
55	1022	Yes
62	1025	No
71	1010	Yes
73	1011	No

Train the decision tree to a depth of two. In other words, decide on a root question, splitting the data into two groups, then decide on another question for each group (if necessary). Your resulting decision tree should have two interior (question) nodes and three leaf nodes and should classify all of the training data correctly. Use the Gini coefficient to measure uncertainty.

You may use code to compute your answer, but you can also solve this problem by hand. If you do use code, provide it with your solution. Your answer should also include a tree which shows the question asked in each interior node.

Hint: Because all of the data are integers, the set of possible questions consists of "Is temp. < 55?", "Is temp. < 65?", ..., "Is pressure < 1030?".