Introduction to Machine Learning

K-Nearest Neighbors Classification

March 20, 2018

Today's activity will be done in groups of 2. The coding activity will be less structured than usual, so you'll have to rely a little more heavily on the pseudocode you write here!

1: Getting the k nearest neighbors

An "instance" of a star looks like this: `[ageOfStarAtDeath, tempOfStarAtDeath, isSuperNova?]`

You're given a list of training instances (the "training set") and one test instance, as well as a value k. Write pseudocode for the function `getNeighbors` that takes these inputs and returns a list of the k nearest training instances to the test instance. Some remarks that will be useful:

1. You can use a function `distance(a,b)` that takes $a = (x_0, y_0)$ and $b = (x_1, y_1)$ and returns $dist(a, b)$.
2. We treat each star as a point on the Cartesian plane (i.e. x = age of star at death, y = temp of star at death).
3. If we sort a 2D list in Python, it will sort by the first element of each inner list. For example: `sorted`$\big([[1,0],[0,1]]\big) == \big[[0,1],[1,0]\big]$

2: Getting the prediction

Now that we have the k nearest neighbors from above, write pseudocode to for the function `getLabel` that gets the model's prediction on a test instance (i.e. the majority "vote" of the nearest neighbors to the instance). In this case, remember that there are only two possible classifications.

3: Test set performance

Using the functions `getNeighbors` and `getLabel`, get the accuracy of a kNN model on a list of test instances (the "test set"). The model's predictions depend on a training set and a given value of k.

4: Write the code!

Open `knnActivity.py` and do challenges 0-4! Make sure to refer to your pseudocode.