Homework 2 | GA Data Science LA 8

Out: Monday, September 14, 2015.
Due (in student repository): Wednesday, September 23, 2015.

Starter File: An optional starter file is provided, **hw1-starter.py**. This file breaks the project into smaller functions. At the bottom of the file is the intended use and ordering of the functions. It can be run immediately. (See the bottom of this file for additional tips.)

Dataset: London 2012 Athletes, **hw2-athletes.csv**.
From: https://docs.google.com/spreadsheets/d/1Y4xQqDuOVbn4mkEFUhW_xE5bgAy51raR1vlNZeXyl3Y/edit#gid=0

K-Nearest Neighbors (k-NN)

In this homework, we will be making a quiz app! Given someone's age, weight, and height, our app will:

1. Find the most similar Olympic athlete(s), and
2. Recommend an Olympic sport.

Exercises

Using native Python (without NumPy and Pandas):

1. Export the data to your hard drive (or use **hw2-athletes.csv**). Open the data file and store each row as a **dictionary**. HINT: Some rows have commas inside quotes. Use the csv.DictReader(…) method in the csv module to easily convert these.

2. Using k-NN, find the k most similar athletes by Euclidean distance. Then, recommend the sport the majority of them participate in. Experiment with various values of k, to ensure they return the most common sport. (A test_point is given to you as an example, in addition to a function which prompts the user for an athlete.)

3. Let's objectively measure our classifier's accuracy! For each athlete in the dataset, predict the athlete's event using k-NN, by excluding that athlete from the nearest neighbors. Find the overall accuracy rate for different values of k. Using this objective method, what is the accuracy for k = 1? k = 50? (This validation technique is called **leave-one-out cross-validation**.)
   Questions to consider:
   - You may only get 25–35% accuracy! Does this make sense? Why?
   - Performing cross-validation will be slow. What makes it slow?

4. We now wonder if this accuracy score is good or not. So, let's compare our model's result to the simplest possible predictor – always predicting the same sport. What sport should you always predict? What accuracy does this predictor yield? Does k-NN do better or worse?

BONUS: Can you improve the accuracy score even more? For example, by selecting additional features, by normalizing each column using **feature scaling**, etc.?

BONUS2: Interestingly, Python includes a special data structure, the heap queue, optimized for finding a small set of minimum values! See if you can integrate it into your k-NN algorithm. See the documentation for heapq.