Assignment 7 - The Circles of Life

Robin Dawes

November 18, 2021

THE local zoo has enlisted your help to separate their collection of animals into related groups.

.

The Problem

THE K-TOWN ZOOLOGICAL INSTITUTE has 100 types of animals on exhibit (including humans - the zookeepers!) The Board of Directors has decided that the animals should be grouped together on the basis of shared characteristics. The zoo has 7 buildings, so you are tasked with separating the species into **7** groups.

Fortunately the zoo is dedicated to the ethical treatment of animals and participates in many international programs of conservation and preservation, so we need have no worries about unintentionally becoming characters in the next Tiger King series.

There are 15 characteristics, each representing an aspect of the animal. Most (such as "airborne" and "milk") are either 0 or 1. But some (such as "legs") can take higher values. All the characteristic values are integers.

The method you will use to group the animals is called K-MEANS CLUSTERING. It works like this:

Choose the number of clusters (groups) you want. In this case we are told the number of groups must be 7.

Choose 7 animals at random. These are the initial "cluster centres". Then repeat the following steps:

For each animal, determine which of the cluster centres it is closest to.[1] This separates the animals into 7 clusters.

[1] See below under **Manhattan and Euclid** for a discussion of "closest".

For each cluster, compute a new centre by averaging the values of each characteristic for the animals currently in the cluster. This new centre will probably have non-integer values for its characteristics and will almost certainly not correspond exactly to any of the animals. That's ok!

This gives us 7 new cluster centres, so we can repeat the process of assigning the animals to clusters by choosing the nearest cluster

centre.

We repeat this process until the clusters stabilize (i.e. they stop changing or almost stop changing). I'll describe ways of measuring stability later in this assignment.

EXAMPLE:

Suppose we have just 5 animals, and they only have 3 characteristics A, B and C. We can represent the data like this (I picked random numbers for the characteristic values - in the actual data set the values are not randomly assigned):

Animal	A	B	C
Horse	1	2	4
Cow	8	2	3
Bear	2	6	1
Moose	2	4	4
Duck	1	6	3

Suppose that we decide to separate these animals into 2 groups based on the 3 characteristics. We randomly choose 2 initial centres (I'll pick Horse and Moose), and compute which of these is closer for each of the other animals. I'm using a simple distance measure called the Manhattan Metric[2]. Using that metric, Cow is closer to Horse, and Bear and Duck are both closer to Moose. This gives us two clusters: [Horse, Cow] and [Moose, Bear, Duck]. We compute the new centre for each cluster by averaging the characteristic values for the animals in the cluster. So the updated centre for Cluster 1 is the set of characteristic values [9/2, 2, 7/2] and the updated centre for Cluster 2 is [5/3, 16/3, 8/3]

[2] See below

Now for all the animals we compute which of the new cluster centres they are closest to. None of them switches to the other cluster, so the clusters do not change. We are done - but if any of the animals had switched to the other cluster we would have iterated again, calculating new centres and seeing if the clusters changed again.

Suppose we choose Horse and Cow as the two initial centres. This gives the two initial clusters as [Horse, Bear, Moose, Duck] and [Cow]. This solution is also stable. Computing the new centres and computing which new centre is closest for each animal does not result in any changes.

This example illustrates that the algorithm can settle on different solutions, depending on the starting points. We typically run the algo-

rithm several times with different randomly chosen starting points to see if one solution comes up more frequently than others.

The Assignment

Part 1

Use NumPy to create an array with one row for each animal and one column for each characteristic. The data is in the file "zoo_2.txt". This dataset was downloaded with permission from `https://www.kaggle.com/uciml/zoo-animal-classification`

The datafile includes a header row containing names for the characteristics. It also has all the species names in the first column. You can ignore the names of the characteristics but you will need the species names to compare your results to the "right answer" (see Part 4.)

You can use features of Python to ignore the first row and first column of the data when building your NumPy array, OR you can edit the data file to remove the first row and first column (and then put the species names in a separate file and read them in from there). Either approach is acceptable.

Part 2

Use the k-means clustering algorithm to divide the animals into 7 clusters. Use the **Manhattan Metric** to compute distances. Run the clustering algorithm at least 10 times so that you are confident about the results. If there is no clear "most likely" result, that is the conclusion for this part of the experiment.

Part 3

Repeat Part 2 but use **Euclidean Distance** to measure distance instead of **Manhattan Metric**.

Part 4

This dataset comes with a "right answer". This is posted on the assignment page in the file "class_2.txt"

Compare your two solutions (the solution from the Manhattan Metric and the solution from Euclidean Distance) to the "right answer". How well or poorly did they do? Did either perform significantly better than the other? (Your results are not required to match the "right answer" perfectly.)

Part ∞

Optional (i.e. not required) follow-on activities:

Look up alternative distance measures and experiment with them in the context of k-means clustering on this dataset.

Look up other clustering algorithms (k-means is one of the most popular - and simplest - but there are many others) and try them out on this dataset.

Locate more datasets and test your clustering algorithm on them. Try `https://data.world/datasets/clustering`

Manhattan and Euclid

Let X and Y be two lists of values, with n values in each list.

$X = x_1, x_2, \ldots, x_n$ and $Y = y_1, y_2, \ldots y_n$

The MANHATTAN METRIC distance between X and Y is computed by

$$MM(X,Y) = \sum_{i=1}^{i=n} |x_i - y_i|$$

The EUCLIDEAN distance between X and Y is computed by

$$ED(X,Y) = \sqrt{\sum_{i=1}^{i=n} (x_i - y_i)^2}$$

Stability

We accept that the k-means clustering algorithm has reached stability when any one of three conditions is met:

1. no items switch to a different cluster after the distances are computed
2. none of the cluster centres move by more than a small amount. For example, we might decide that if the distance between the previous cluster centre and the new one is less than some small value ϵ (eg. $\epsilon = 0.1$), that signifies stability
3. we reach some pre-determined number of iterations. For this dataset - which is quite small - 100 iterations should be more than enough.

How You Will Be Graded

The assignment will be marked out of 100. 90% of the grade will be for correctness and 10% of the grade will be for programming style.

The grader will read your code and will run your program to test correctness.

What to Submit

For this assignment, you are required to upload to onQ:

- A Python program containing
    (a) your implementation of the k-means clustering algorithm
    (b) your use of NumPy to import the data from a text file
    (c) your code to conduct the clustering of the data using both the Manhattan Metric and the Euclidean Distance
- A text file or pdf containing your answers to the questions stated in the Assignment section
- You are NOT required to upload the html page generated by pydoc, because I know some students in the class have not been able to get this to work. However, your code must be properly documented with docstrings.

Remember to put your name and student number at the top of your program file, as well as the statement regarding academic integrity (as specified in Assignment 1). Also, your program must contain appropriate docstring documentation at the beginning of the program and in each defined function.

Due Date

The due date for this assignment is 20211126 (November 26), 11:59 PM.