

CISC-124
Winter 2018

Assignment 2

Classifiers are hugely important software tools in our society. Recommendation systems such as those used by Netflix and Spotify are based in part on classifying movies and songs into specified categories. In this assignment you will implement a very simple form of classifier for your employer's music collection.

Your employer has measured 6 aspects of each song (these would be such things as rhythm, tempo, dynamics, etc.) on a scale of 0 to 10. Thus each song is described by an ID string (the song title) and 6 integer values. An example would be
Maple Sidewalk,3,0,8,1,1,7

She has also defined a number of "categories" - each category is defined by an ID number and specific values for the 6 musical aspects. An example would be
1487,2,6,7,2,10,4

File **categories.txt** consists of several lines that represent "categories" as explained above. Each line consists of

- an ID number for the category (an integer)
- 6 integers giving the values of the aspects

All these fields are separated by commas.

You have been told that there will not be more than 10 categories.

File **songs.txt** consists of many lines that represent songs, each line consisting of

- a song ID (a string) that does not contain a comma
- 6 integers giving the values of the aspects for the song

All these fields are separated by commas. You have not been told how many lines are in this file. The file may contain bad lines that you should detect and report in an error trace file.

Examples of these lines are:

my song, my poem,3,0,7,4,1,1	(there is a comma in the song title)
bad song, 2,3,OK,5,2,0	(one of the aspects is not an integer)
good song,1,2	(not enough aspect values)
best song, 5,1,0,4,7,9,3	(too many aspect values)

We can measure the *distance* of a song to a category by summing the squares of the differences between the aspect values of the category and the aspect values of the song.

For example, if the category has aspect values 3, 8, 2, 2, 9, 6
and the song has aspect values 5, 8, 4, 1, 7, 0

then the distance of the song to the category is given by

$$\begin{aligned} & (3 - 5)^2 + (8 - 8)^2 + (2 - 4)^2 + (2 - 1)^2 + (9 - 7)^2 + (6 - 0)^2 \\ &= (-2)^2 + 0^2 + (-2)^2 + 1^2 + 2^2 + 6^2 \\ &= 4 + 0 + 4 + 1 + 4 + 36 \\ &= 49 \end{aligned}$$

Clearly if the song matches the category exactly the distance will be 0, but if the song has aspect values that are very different from the category then the distance will be large.

For each song, your program must compute the distance from the song to each of the categories.

Produce an output file named “song_category.txt”, with one line for each song, where the line contains the following information:

- song ID string
- ID number of the category that is closest to the song. (If two categories are tied for closeness to the song, list the category with the lower ID number.)

Produce an output file named “category_stats.txt”, with one line for each category, where the line contains the following information:

- category ID number
- the number of songs that chose this category as closest
- the song ID string of a song that is closest to this category

Produce an output file named “error.txt” which contains a “dump” of any bad lines detected in the song.txt file. Each dumped line has an attached message describing the reason for its rejection. Examples of these messages are: "line contains invalid aspect", "line contains insufficient aspects", "line's song ID string contains commas", etc.

Marking:

code completeness and style (12 points),
testing and error file (6 points)
documentation (2 points)