Sheffield Hallam University

College of Business, Technology and Engineering

MSc Computer Science & Software Engineering/Big Data Analytics (Year 1)

Module 55-706555: Programming Concepts and Practice

Coursework 1(60%)

Assignment 1 (2020/2021)

1. Introduction

With the overwhelming volume of online content and increasing ubiquity of Internet-enabled devices, pervasive use of the Web for content sharing and consumption has become our everyday routines. However, people seeking online access to content or items of interest are becoming more and more frustrated due to information overload. Deciding the content to consume from deluge of available alternatives becomes increasingly difficult. In this regard, an online content streaming company provides movies, TV shows, music and other services to millions of online users. The company makes profits from selling interesting content/ services of all genres to their customers. In the last couple of years, the fortune of the company has suffered due to dwindling sales. The management decided to improve their sales by **developing an intelligent service recommendation engine for their online platform**. To achieve this, the company has hired you as their data engineer to design and develop an intelligent recommendation engine. One of the key contents provided by the company is an online music streaming service.

The company has provided you with a dataset containing music tracks of various artists as well as the music features. Your task is to deliver the recommendation engine, working with the datasets to analyse, design, and implement the system.

In the first stage of the development, specifically, you will analyse, design and implement a solution based on the concepts and principles taught in the module.

2. Getting Started and General Specifications

The following tasks are to be performed in this assignment:

a) Loading and parsing music dataset file

In this task, you will develop a suitable module for retrieving data from the provided dataset. The data to be retrieved will be stored in the memory using two dictionaries, one for the artists (with their music plus features), and the other for the music tracks (music tracks and their features). You are provided with a sample dataset. Information regarding the dataset can be found in section 3.

b) Computing similarity between artists

In this task, you will design and implement 5 functions that will compute the similarity scores between two artists using features retrieved from task(a). You will also evaluate these functions by explaining and justifying which of the similarity measures you consider the best in your mini-report.

c) Computing similarity between music tracks

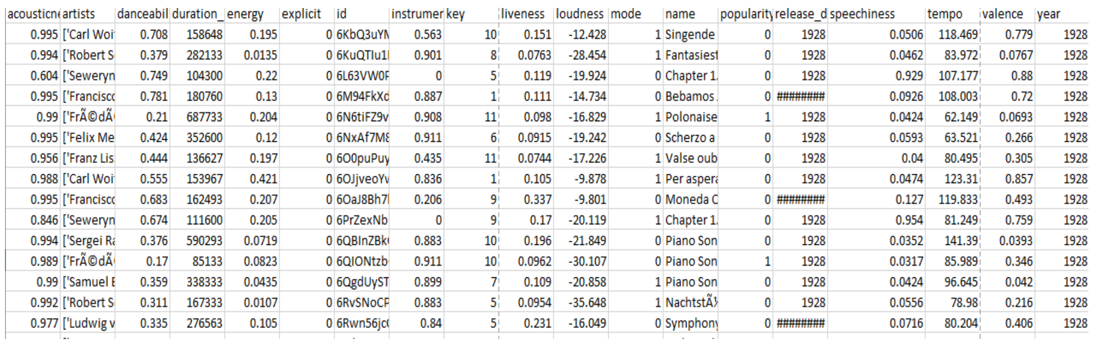
In this task, you will design a function that computes similarity scores between two music tracks. You have the flexibility to design your own metric for computing the similarity scores, potentially using the metrics implemented in (b).

d) Mini-Report

You will write a report of your implementation, summarising your design, implementation decisions, justifications, and the pseducode/algorithms for your functions/system. Provide an explanation in your mini-report on how to execute your application. Also, your mini-report should contain the structure of your program.

3. Sample Dataset

In the folder (music), you will find the dataset, “data.csv.zip” containing features as shown below. You are required to extract the following and other relevant features: “acousticness”, “artists”, “danceability”, “energy”, “id”, “liveness”, “loudness”, “name”, “popularity”,”speechiness”, “tempo”, “popularity”, “energy”, and “valence”.



acousticne	artists	danceabil	duration_	energy	explicit	id	instrumer	key	liveness	loudness	mode	name	popularity	release_d	speechiness	tempo	valence	year
0.995	['Carl Woi	0.708	158648	0.195	0	6KbQ3uYN	0.563	10	0.151	-12.428	1	Singende	0	1928	0.0506	118.469	0.779	1928
0.994	['Robert S	0.379	282133	0.0135	0	6KuQTIu1I	0.901	8	0.0763	-28.454	1	Fantasiest	0	1928	0.0462	83.972	0.0767	1928
0.604	['Seweryn	0.749	104300	0.22	0	6L63VW0F	0	5	0.119	-19.924	0	Chapter 1.	0	1928	0.929	107.177	0.88	1928
0.995	['Francisco	0.781	180760	0.13	0	6M94FkXd	0.887	1	0.111	-14.734	0	Bebamos .	0	########	0.0926	108.003	0.72	1928
0.99	['FrÃ©dÃ	0.21	687733	0.204	0	6N6tiFZ9v	0.908	11	0.098	-16.829	1	Polonaise	1	1928	0.0424	62.149	0.0693	1928
0.995	['Felix Me	0.424	352600	0.12	0	6NxAf7M8	0.911	6	0.0915	-19.242	0	Scherzo a	0	1928	0.0593	63.521	0.266	1928
0.956	['Franz Lis	0.444	136627	0.197	0	6O0puPuy	0.435	11	0.0744	-17.226	1	Valse oub	0	1928	0.04	80.495	0.305	1928
0.988	['Carl Woi	0.555	153967	0.421	0	6OJjveoYv	0.836	1	0.105	-9.878	1	Per asper	0	1928	0.0474	123.31	0.857	1928
0.995	['Francisco	0.683	162493	0.207	0	6OaJ8Bh7I	0.206	9	0.337	-9.801	0	Moneda C	0	########	0.127	119.833	0.493	1928
0.846	['Seweryn	0.674	111600	0.205	0	6PrZexNb	0	9	0.17	-20.119	1	Chapter 1.	0	1928	0.954	81.249	0.759	1928
0.994	['Sergei Ra	0.376	590293	0.0719	0	6QBInZBk	0.883	10	0.196	-21.849	0	Piano Son	0	1928	0.0352	141.39	0.0393	1928
0.989	['FrÃ©dÃ	0.17	85133	0.0823	0	6QIONtzb	0.911	10	0.0962	-30.107	0	Piano Son	1	1928	0.0317	85.989	0.346	1928
0.99	['Samuel E	0.359	338333	0.0435	0	6QgdUySTI	0.899	7	0.109	-20.858	1	Piano Son	0	1928	0.0424	96.645	0.042	1928
0.992	['Robert S	0.311	167333	0.0107	0	6RvSNoCP	0.883	5	0.0954	-35.648	1	NachtstÃ¼	0	1928	0.0556	78.98	0.216	1928
0.977	['Ludwig v	0.335	276563	0.105	0	6Rwn56jc	0.84	5	0.231	-16.049	0	Symphony	0	########	0.0716	80.204	0.406	1928

4. Implementations

Your task is to design and develop two modules namely ***load_dataset_module***, ***similarity_module***, and a main function that uses these modules.

a) The first module (task a) (***load_dataset_module***) retrieves “accoustiness”, “artists”, “danceability”, “energy”, “id”, “liveness”, “loudness”, “name”, “popularity”,”speechness”, “tempo”, and “valence” from the data.csv dataset.

b) This module should implement two functions that returns (respectively) two dictionaries (***artist_music*** and ***music_features***). The ***artist_music*** dictionary contains artists, music name, and corresponding features. The ***music_features*** dictionary contains music id, and their respective features.

c) The second module (task b) (***similarity_module***) implements 5 functions, each providing functionality for computing similarity between either between music or between artists. These functions are: **euclidean_similarity** (Euclidean distance), **cosine_similarity** (Cosine distance), **pearson_similarity** (Pearson correlation), **jaccard_similarity**(Jaccard distance) and **manhattan_similarity**(Manhattan distance). You may need to research the mathematical formulas for these similarity metrics. Each of these functions will accept 3 parameters namely: the ***artist_music*** *or* ***music_features*** dictionary and two ids (music id or artist id). The functions should be able to use features e.g. loudness, liveness, valence, etc. to compute the similarity.

d) Apart from computing similarity between artists, you should also implement a function for computing similarity between two music tracks. The function should accept appropriate parameters such as the music ids, and one of the similarity functions in (c). This function is also a part of the ***similarity_module*** in (b). Hint: your function should be able to use any of the features retrieved from the dataset to compute the similarity.

e) A main function that implements a simple user interface for accepting user inputs such as music ids, artist ids, what similarity metric to use, etc. It will provide the functionality for displaying similarity between two artist or two music tracks. It should allow users to quit when they are done with the application.

5. Pay attention to the following requirements

a) In-module retrieval is available for this assessment. If you do not achieve the pass mark in the first attempt, you have an opportunity to rework your assignment and resubmit for reassessment.

b) This assignment is an individual piece of work, and your submission must be in the form of **two module files (.py) and a Jupyter Notebook file (.ipynb)**. We should be able to open and run your modules on a standard campus computer.

c) You will submit a mini-report. The report should provide analysis of the problem being solved, justification for your design decisions and pseudocodes for the similarity metrics and other functions you have developed. It should explain the relationships between the modules. A good report should provide evidence of critical analysis of the implemented system. Even if your application does not work correctly, you should still submit the mini-report explaining what you have done, what works and what has not worked.

d) Any evidence of collusion/plagiarism will be penalised if appropriate! If there is some doubt about the authenticity of a particular piece of work, then the person submitting it will be expected to defend such work, including reasons for the programming decisions taken. You must document with references any use of libraries or existing code in your mini-report.

e) This assignment is linked to assignment 2. This means that assignment 2 is a continuation of assignment 1.

f) Appropriate use of variable names for clearer understating is desirable. Adequate commenting of your code for easier understanding is also desirable.

6. Submission Process

Your assignment should be submitted electronically through the module's Blackboard site as a single ZIP file that contains all your source codes and mini-report, not more than 5 pages. Check your upload to ensure you have submitted the correct files successfully as any issues will not be considered after the deadline.

Submission Deadline

Friday, 4th December, 2020, 2:59 pm

7. Assessment Criteria

This first assignment is linked with the second assignment. This means you cannot execute the tasks in the second assignment without those in the first assignment. Both assignments are individual pieces of work, and your submission must be in the form of implemented modules and a mini-report.

This assignment assesses the module's learning outcomes (LO) in the following way:

- This assignment focuses on designing and implementing python module for loading and extracting data from real-world datasets and a module for computing user or item similarity, using appropriate data structures, string manipulation, iteration, selection, etc.(LO1) and programming design strategies(LO2).

This assignment will be assessed mainly by **code testing/inspection, and through a video demonstration** of the submitted codes using the data files accompanying this brief. **You should submit a video demo of maximum of ten minute, demonstrating how your solution meets the assessment criteria.** In addition, the coursework will be assessed against the Learning Outcomes (LOs) using a set of assessment criteria. This set of assessment criteria allows assessing how successful you have met the LOs. In order to ensure consistent use of the relevant criteria, the assessment criteria are summarised in the following assessment matrix and grid. This is an indicator of how the marks will scale across each category of the learning outcomes it covers.

Assessment Matrix

Assignment	**Assessment Criteria**	**Marks**	**Learning Outcomes Covered**		
			LO1	LO2	LO3
Assignment 1	Elucidation of the problem and elicitation of solution requirements	10%	X	X	X
	Program development using design strategies	15%	X		X
	Implementation of software solution using relevant programming concepts	25%	X	X	X
	Quality and usefulness of mini-report and deliverable	10%	X	X	

Sheffield Hallam University

Assessment Marking Grid

Fail (<50%)	Pass(50-59)	Merit(60-69)	Distinction (70% +)
Elucidation of the problem and elicitation of solution requirements (/10)			
No evidence of understanding of problem being solved. No solution requirements.	Good evidence of understanding of the problem, its definition and analysis. Good understanding of important solution requirements, may lack some clarity, missing requirements, etc.	Very good demonstration of knowledge and understanding of the problem. Very good understanding of the system requirements and its analysis.	Excellent demonstration of understanding of the problem being solved. Excellent details of solution requirements and analysis.
Program development using design strategies (/15)			
No application of design strategies. Limited use of design strategies	Good use of appropriate design strategies in the program development process. Minor issues, such as lack of use of some design strategies such as missing such as pseudocodes or architectural figures/flowcharts	Very good use of appropriate design strategies in the program development process. Use of pseudocodes, algorithms, flowcharts, use of figures e.g. high-level architectural diagram showing key aspects of your solution, etc.	Excellent use of appropriate design strategies in the program development process.
Implementation of solution using relevant programming concepts (/25)			
Lack/poor use of programming concepts.	Good understanding and use of appropriate programming concepts such as selection, repetition, sequences, functions and parameters, etc. Use of appropriate data types, data structures, etc.	Very good understanding and use of appropriate programming concepts such as selection, repetition, sequences, functions and parameters, exception handling, etc. Software does not crash when being run. Entering wrong input does not crash the program but gracefully handles such exceptions, etc.	Excellent, professional level use of programming concepts in the implementation, robust application delivered.
Quality and usefulness of mini-report and deliverable (/10)			
Report lacking good structure, no personal reflection, no	Good structure, evidence of personal reflection on what went well or not. Good justification for design and implementation	Very good structure, evidence of personal reflection on what went well or not. Very good justification for design and	Excellent structure, excellent personal reflection on what went well or not.

description of the deliverable or explanation and justification of decisions. Poor use of language.	decisions. Good use of language.	implementation decisions. Good use of language.	Good justification for design and implementation decisions. Excellent use of language. Evidence of innovation in the deliverable, e.g. excellent user interaction through UI, etc.

****

All work must be your own. *If evidence of collusion/copying is found, then* ***such collusion will be penalised, severely if appropriate!*** *If there is some doubt about the authenticity of a particular piece of work, then the person submitting it will be expected to give a detailed explanation of such work, including reasons for the programming decisions taken.*

****

References

1. Spotify music dataset: https://www.kaggle.com/yamaerenay/spotify-dataset-19212020-160k-tracks
2. A.S Shirkhorshidi et al(2015) A Comparison Study on Similarity and Dissimilarity Measures in Clustering Continuous Data