Sheffield Hallam University

Intelligent Service Recommendation Engine Implementation Report

SUBMITTED BY	VAIBHAV RANJAN
STUDENT ID	C0035906

Contents

LIST OF FIGURES

1. INTRODUCTION

Twenty first century has seen a remarkable rise in volume of online content and devices capable of processing them. Information overload is making selection of appropriate content very cumbersome. There is a great potential of business improvement and expansion in the field of intelligent service recommendation engine, this will not only provide recommendations to users and enhance their experience, it will also provide insights for organisation to bring the type of content which will be beneficial for business growth and market expansion.

2. PROBLEM EXPLORATION

The problem deals with content overloading to the front-end user and looks into potential solutions to make the better use of data by recommending the content based on usage analytics of the user. A dataset with 169910 rows and 19 columns is provided which contains features of songs and artists based on various properties measured as continuous data. The snapshot of the data is shown below:

acousticness	artists	danceability	duration_	energy	explicit	id	instrumen	key	liveness	loudness	mode	name	popularity	release_d	speechine	tempo	valence	year
0.995	['Carl Woit	0.708	158648	0.195	0	6KbQ3uYM	0.563	10	0.151	-12.428	1	Singende B	0	1928	0.0506	118.469	0.779	1928
0.994	['Robert Sc	0.379	282133	0.0135	0	6KuQTIu1K	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	6L63VW0P	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 J	0	########	0.0926	108.003	0.72	1928

Figure 1 DATASET SNAPSHOT

Dataset consists of ‘artist name’, ‘song name’, ‘ID’ ‘acousticness’, ‘artists’, ‘danceability’, ‘energy’, ‘id’, ‘liveness’, ‘loudness’, ‘name’, ’speechiness’, ‘tempo’, ‘popularity’, ‘energy’, and ‘valence’. The intelligent recommendation system will utilize the data available in the dataset and use five similarity functions to provide user with the similarity score. The metrices to be used are Euclidean distance, cosine distance, Pearson Correlation Coefficient, Jaccard Distance Coefficient and Manhattan distance.

3. SOLUTION SPECIFICATION

The solution will consist of four main aspects.

3.1. Loading and Retrieving Data

This module will load the data and retrieves all the relevant details needed for calculating the scores using different functions. This step is very significant as it provides base for the next step of formulating dictionaries.

3.2. Dictionary Creation

This section should generate two dictionaries namely 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.

3.3. Similarity Function Module

This section will contain methods for calculating similarity using data available from both artist_music and music_features dictionaries by using metrices mentioned in problem exploration section.

3.4. User Interface

Interaction with user is very import aspect of any application and this section will fulfil UI requirements. This section will require user input about the specification to be compared i.e., artist or music track and similarity index to be used.

4. DESIGN DESCRIPTION INFORMATION CONTENT

4.1. Data Source Identification

The software will use data from ‘Data.csv’ stored by administrator in the same location as the module files.

4.2. Stakeholders

The administrator and the user are two primary stakeholders.

4.3. Design Views

This project will be implemented in modular structure. It is useful for stakeholders, because they can add new features or remove the modules they do not need from the project.

4.4. Design Rationale

By taking important features such as performance, maintainability, generalizability of the system, we agreed on the method of design. The coding will be done in well commented way for easy understanding.

4.5. Design Languages

Python 3+ language will be used and it is recommended to use spyder or Jupiter notebook for running the program.

5. DESIGN VIEWPOINTS

5.1. General Use Case

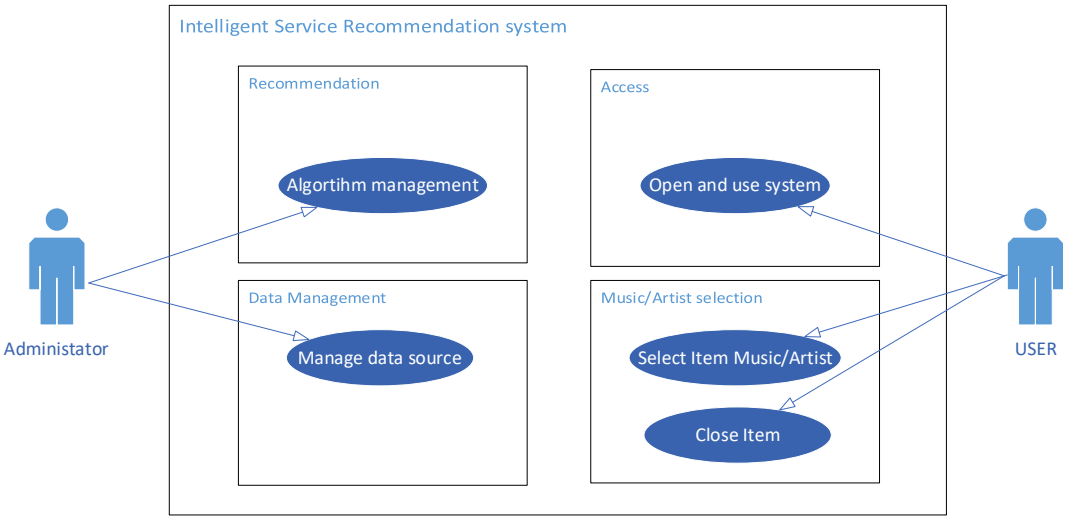

Figure 2 GENERAL USE CASE

The operations in the used case are described below

5.1.1. Algorithm management

An administrator can manage and update algorithm in case of requirement modification.

5.1.2. Manage data source

The location of data is administrated by administrator.

5.1.3. Open and use system

User can open the system and start using it.

5.1.4. Select Item
User selects item for comparison i.e., artist or music.

5.1.5. Close Item
User closes the program when needed.

5.2. Software Architecture and Design

The system architecture describes system as component of its subsystems. There project is formed by three submodules: data pre-processing, similarity algorithms and user interface. Detailed explanation about relation between models are given below:

5.2.1. Data Pre-Processing
This module reads the data and separates the file using coma (,) and then divides the data into two dictionaries namely: artist_dictionaries and music_dictionaries and then extracts numerical features like danceability, valence etc. into artist_features_matrix where artist section is subdivided based on count of artists in particular song and music_features_matrix containing music ID and relevant features. In case of artist_features_matrix a matrix is returned when there are multiple songs by same artist.

5.2.2. Similarity Algorithms
This section of the program evaluates various similarity metrices based on the inputs provided by the user. Numerical Data is extracted based on inputs provided by the user and it is input to the algorithm module which calculates various metrices shown below:

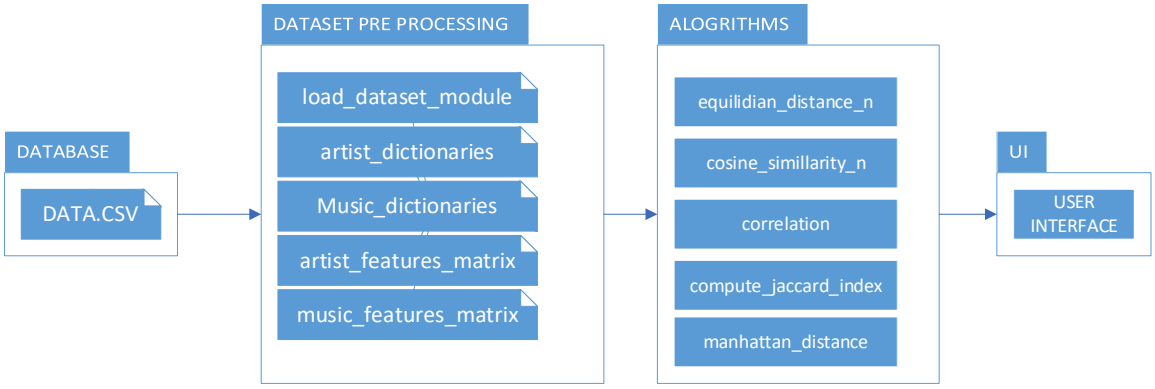

Figure 3 SYSTEM SEQUENCE DIAGRAM

5.2.3. User Interface Steps
Please save "**load_dataset_module**", "**similarity_module**", "**main_ui**" and "**data.csv**" in the same location.

5.2.3.1. Open and Run the" **main_ui**" module.
5.2.3.2. User enters value of '0' for artist comparison and '1' for music comparison.
5.2.3.3. After selection first artist name is entered followed by second.
5.2.3.4. User is asked to select option from 1-5 for various metrices.
5.2.3.5. Result is displayed.
5.2.3.6. Option to quit application or continue comparison is displayed.

```
Press 0: ARTIST COMPARISON and 1 : MUSIC COMPARISON  : 0

Please enter the name of FIRST ARTIST: Vladimir Horowitz

Please enter the name of SECOND ARTIST: Vladimir Horowitz

WHICH SIMILARITY MATRIX YOU WANT TO USE, PLEASE ENTER NUMBERS FOLLOWED BY
SPACE IN CASE OF MULTIPLE
1: EQUILIDIAN DISTANCE
2: COSINE SIMILARITY
3: PEASRSON CORRELATION
4: MANHATTAN DISTANCE
5: JACCARD INDEX

PLEASE ENTER YOUR CHOICE BETWEEN 1 TO 5: 1 2 3 4 5

THE DISTANCE CALCULATIONS ARE SHOWN BELOW
EQUILIDIAN DISTANCE : 0.0
COSINE SIMILARITY : 0.77
PEASRSON CORRELATION : 1.0
MANHATTAN DISTANCE : 0.0
JACCARD INDEX : 1.0

Do you want to quit application? PRESS 1:YES  0:NO :1
```

Figure 4 USER INTERFACE

5.2.4. User Effort Estimation

5.2.4.1. Navigation: In Spyder Press run icon to run program. In Jupyter notebook press the run line icon.

5.2.4.2. Data Entry: A max of 9 entries in once cycle. Calculations are described below

- •Enter 0 or 1
- •Enter Artist name or Music ID (e.g., 6KuQTIu1KoTTkLXKrwlLPV)
- •Enter numbers 1-5, each followed by space to select options.
- •Press 0 or 1 for continuing or exit

5.2.5. Module Linkages

The figure below describes the interaction between different modules in the software. The data pre-processing unit reads the data and parses it into 19 columns which is then read separately into music and artist dictionaries. The user then provides the input for retrieval of relevant dictionary and to select the similarity metrices. Once user selects the metrices an output is displayed with the selected features.

Note: In case of artist with multiple songs a matrix will be returned by data pre-processing unit which will be used for distance calculations. For instance, 'Vladimir Horowitz' have 614 songs in database so the artist dictionary will have 614 rows and 10 columns with all numerical data.

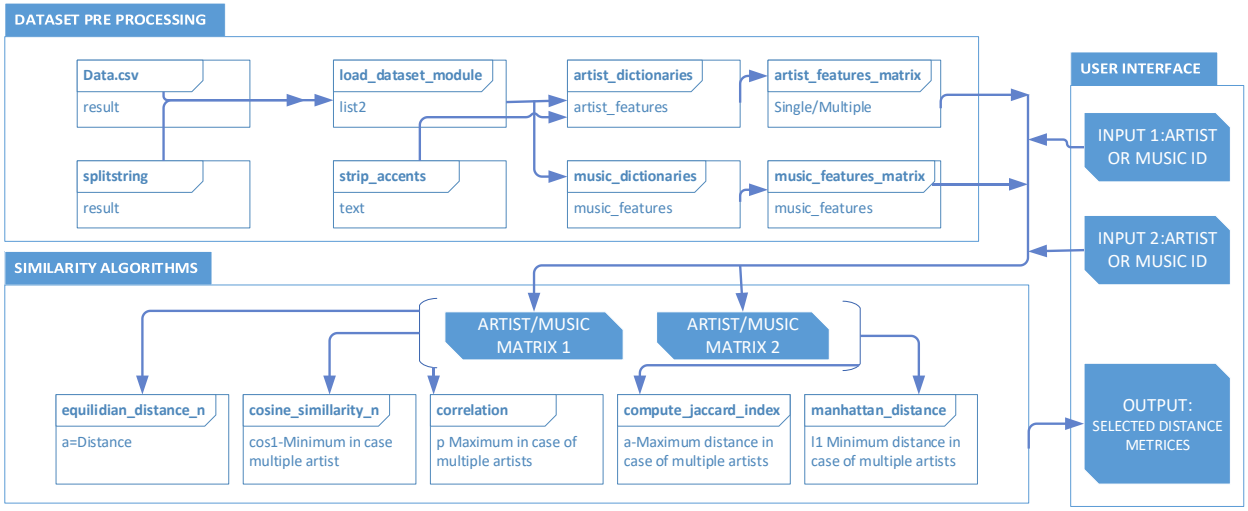

Figure 5 MODULE LINKAGES

5.2.6. System Requirements

5.2.6.1. Anaconda

- •Operating system: Windows 8 or newer, 64-bit macOS 10.13+, or Linux, including Ubuntu, RedHat, CentOS 6+, and others.
- •System architecture: Windows- 64-bit x86, 32-bit x86; MacOS- 64-bit x86; Linux- 64-bit x86, 64-bit Power8/Power9.
- •Minimum 5 GB disk space to download and install.

5.2.6.2. Team Studio

- •Memory and disk space required per user: 1GB RAM + 1GB of disk + .5 CPU core.
- •Server overhead: 2-4GB or 10% system overhead (whatever is larger), .5 CPU cores, 10GB disk space.
- •Port requirements: Port 8000 plus 5 unique, random ports per notebook.

6. VALIDATION AND EVALUATION

The cosine similarity and Pearson correlation yields similar results for same set of artist or songs and the results from Manhattan and Euclidian were showing similar trend. A close look on the dataset shows that the variation in 'Loudness' and 'Tempo' is high which will have large impact on Euclidian and Manhattan distance as they are highly affected by large scale attributes.

Evaluating a proposed music recommendation system finds one of the biggest challenges of designing one. Absence of standard ground truth data for evaluation makes this task very subjective. However, a more accurate method to evaluate a recommendation system is to use humans to evaluate music recommendations.

7. CONCLUSION

The produced results demonstrate the feasibility of this type of intelligent recommendation engine. Based on the results it can be observed that for most of the inputs there is a pattern between Euclidian-Manhattan and Cosine -Pearson. Since, Manhattan Distance is preferred over the Euclidean distance metric as the dimension of the data increases (Aggarwal et al., 2001) and the Pearson correlation is invariant to both scale and location changes of variables ((Egghe & Leydesdorff, 2009), the usage of Manhattan and Pearson as a combination is recommended. Manhattan will specify the distance between the pair of artist or songs and Pearson will represent corelation with 1 as perfect -1 as perfectly negative and 0 as no correlation.

8. REFERENCES

Aggarwal, C. C., Hinneburg, A., & Keim, D. A. (2001). On the surprising behavior of distance metrics in high dimensional space. *International Conference on Database Theory*, 420–434.

Egghe, L., & Leydesdorff, L. (2009). The relation between Pearson's correlation coefficient r and Salton's cosine measure. *Journal of the American Society for Information Science and Technology*, *60*(5), 1027–1036. https://doi.org/10.1002/asi.21009