EVALUATING THE EFFECTIVENESS OF MUSIC RECOMMENDATIONS USING COSINE SIMILARITIES OF VARIOUS COMBINED FEATURE SETS CONSISTING OF METADATA AND USER GENERATED TAGS

A Paper
Submitted to the Graduate Faculty
of the
North Dakota State University
of Agriculture and Applied Science

By

Shubhit Kumar

In Partial Fulfillment of the Requirements
for the Degree of
MASTER OF SCIENCE

Major Department:
Computer Science

April 2022

Fargo, North Dakota

North Dakota State University
Graduate School

Title

EVALUATING THE EFFECTIVENESS OF MUSIC RECOMMENDATIONS USING COSINE SIMILARITIES OF VARIOUS COMBINED FEATURE SETS CONSISTING OF METADATA AND USER GENERATED TAGS

By

Shubhit Kumar

The Supervisory Committee certifies that this ***disquisition*** complies with North Dakota State University's regulations and meets the accepted standards for the degree of

MASTER OF SCIENCE

SUPERVISORY COMMITTEE:

Anne Denton
Chair

Lu Liu

Sigurd Johnson

Approved:

April 26, 2022	Simone Ludwig
Date	Department Chair

ABSTRACT

This paper aims to compare and evaluate the methods for recommending songs using metadata, user generated tags, metadata and tags, and metadata and top five tags. Recommendations are calculated using cosine similarities with relevant exceptions (artist filtering, tag filtering, year to decade conversion) investigated.

ACKNOWLEDGMENTS

I would like to acknowledge my advisor, Dr. Anne Denton. Thank you for all your knowledge, guidance, and enthusiasm throughout this process.

I would also like to acknowledge my committee members, Dr. Lu Liu and Dr. Sigurd Johnson. Thank you for your time and insight.

DEDICATION

I would like to dedicate this paper to my maa, dad, and uncle. Thank you for always being there to guide me and for pushing me academically to achieve my best.

TABLE OF CONTENTS

ABSTRACT iii

ACKNOWLEDGMENTS iv

DEDICATION v

LIST OF TABLES viii

LIST OF FIGURES x

1. INTRODUCTION 1

2. BACKGROUND INFORMATION 2

3. RELATED WORK 3

4. GATHERING DATA 4

4.1. Picking Dataset 4

4.2. Collecting Metadata 5

4.3. Collecting Credited Collaborators 6

4.3.1. Getting Featured Artists Data 7

4.3.2. Getting Producer Data 7

4.4. Collecting User Generated Tags 8

4.5. Additional Sanitization Processing 9

5. CODE AND MATH EXPLANATION 11

5.1. Terminology 11

5.2. Frequency Matrix 12

5.3. Document Creation 12

5.4. Cosine Similarity 15

5.4.1. Cosine Similarity Explanation 15

5.4.2. Cosine Similarity Processing 17

6. RESULTS 19

6.1. Creating Recommendation Request 19

6.2. Artist Filtering 19

6.3. Year to Decade Conversion 23

6.4. Test Results: Electro-pop Recommendations 27

6.4.1. Metadata Recommendations 27

6.4.2. Tag Recommendations 29

6.4.3. Metadata and Tags Recommendations 31

6.4.4. Tag Saturation 31

6.5. Test Results: Hip-Hop/Rap Recommendations 33

6.5.1. Metadata Recommendations 33

6.5.2. Tag Recommendations 34

6.5.3. Metadata and Top Tags Recommendations 35

6.6. Language Exclusion 37

6.7. Results of Zero Vectors 39

6.8. Comparison Summary 41

7. FURTHER RESEARCH 45

7.1. Writer Data 45

7.2. Theme Data 46

7.3. Evaluating Other Datasets 46

7.4. Surveys and Other Recommendation Methods 47

7.5. Reevaluating Tag Ranks 47

8. CONCLUSION 48

REFERENCES 49

LIST OF TABLES

Table Page

1. Preview of aggregated metadata data file. 6
2. Preview of data file including featured artists and producers. 7
3. Recommendations for “LAST WALTZ” by TWICE using metadata 19
4. Recommendations for “LAST WALTZ” by TWICE using tags 20
5. Recommendations for “LAST WALTZ” by TWICE using metadata. 22
6. Recommendations for “LAST WALTZ” by TWICE using tags. 22
7. Recommendations for “LAST WALTZ” by TWICE using metadata and tags combined. 23
8. Recommendations for “Summer Of ‘69” by Bryan Adams using metadata. 25
9. Recommendations for “Summer Of ‘69” by Bryan Adams using metadata. 26
10. Recommendations for “Get Lucky” by Daft Punk using metadata. 28
11. Recommendations for “Get Lucky” by Daft Punk using tags. 30
12. Recommendations for “Get Lucky” by Daft Punk using metadata and tags. 31
13. Recommendations for “Get Lucky” by Daft Punk using metadata and top five tags. 32
14. Recommendations for “Mercy.1” by Kanye West using metadata. 33
15. Recommendations for “Mercy.1” by Kanye West using tags. 35
16. Recommendations for “Mercy.1” by Kanye West using metadata and top five tags. 36
17. Recommendations for “Rush” by Seatbelts using metadata and top five tags with language data included. 37
18. Recommendations for “Rush” by Seatbelts using metadata and top five tags with language data excluded. 38
19. Recommendations for “Go Legend (feat. Metro Boomin & Travis Scott)” by Big Sean using tags. 40
20. Side by side comparison of cosine similarities for “Get Lucky” by Daft Punk 42

21. Advantages and Disadvantages of each combined feature set.. 43

LIST OF FIGURES

Figure Page

1. Code snippet of get_metadata.py 5
2. Example vector. 12
3. Code snippet of combine_features_tags function 14
4. Code snippet of document and frequency matrix creation. 15
5. Vector comparison. 16
6. Code snippet of cosine similarity calculation and recommendation sorting. 18
7. Code snippet for decade conversion. 25
8. Code snippet of original combine_features_meta function. 45

1. INTRODUCTION

Recommender systems are systems that take a user's input and attempt to provide recommendations based on what they infer to be the user's preference. The idea of recommender systems has been around since the 1990s and they are commonly used today: Netflix, YouTube, and other video streaming services use recommender systems to recommend users the next video, show, or movie to watch; Amazon and other online marketplaces use recommender systems to try to determine products that a user might want to purchase; and social media sites like Facebook, Twitter, and Reddit use recommender systems to recommend profiles or communities that are trending and that a user may find interesting.

Another use of recommender systems is to recommend music. Many music recommender systems focus on collaborative filtering methods, which recommend music based on what other people with similar tastes like. While such methods address the social aspects associated with music, and by extension music culture, the metadata of songs are often ignored. In this paper we will look and compare the results of song recommendations of different combined feature sets consisting of metadata, user generated tags, and different combinations of both.

2. BACKGROUND INFORMATION

While there are several approaches to recommender systems, the two most relevant approaches in music are content-based filtering and collaborative filtering. The content-based filtering approach attempts to recommend objects, in this case songs, using characteristics of the objects combined with the knowledge of the user’s preferences. Content filtering is a great approach when a lot of data is known about the objects, but not much information is known about the user. The downside to content-based filtering is a lot of prior knowledge about the dataset is required. When looking at music, the metadata of a track provides the data and potential insight that can be used to develop these recommendations.

Collaborative filtering is another popular approach for recommending music. This approach determines recommendations based on users with similar preferences. Collaborative filtering is founded on the idea that if a user agreed with someone in the past, they are likely to agree with that person again in the future. However, this idea relies on the assumption that the user will continue to like the same things they liked in the past. While this is not always the case, it is generally safe to assume a person’s preferences will stay consistent. The issue with collaborative filtering is it is hard to give recommendations at the start because there is insufficient data on users. However, as more data is collected about users, the approach becomes stronger. The other issue is that collaborative filtering requires that extensive data on past preferences are stored for each user for extensive periods of time. Such data is only available within companies that offer large streaming services. Due to privacy concerns, such data is not available to outside entities, and was not available for the recommender system developed in this paper.

3. RELATED WORK

“An Algebra of Recommendations” by Jussi Karlgren is one of the first published papers regarding recommender systems [5]. The idea behind the paper was originally based on the concept of a bookshelf. A good bookshelf is typically ordered by topics and so books can be recommended through proximity (how close another book is to the user’s original book). Meanwhile, an unorganized bookshelf can also recommend using proximity because of the idea that someone took time to place that book near the user’s original book. Whether the book placement was intentional or not, the user will be able to find recommendations by looking nearby. The issue with this idea is that both systems rely on the idea of proximity and because there is no intrinsic sense of proximity while looking at documents on a computer, Karlgren worked to develop one. By simply asking the question “If I like Book A, do you believe I will like Book B?” and aggregating the answers to that question across multiple users, Karlgren created what can be viewed as a digital bookshelf. Through linear algebra and converting these books into documents (discussed in Section 5.3), Karlgren was then able to create a sense of proximity.

The Million Song Dataset is another project created in 2011 that looks at music recommendation [9]. The Million Song Dataset contains the metadata and audio characteristics of one million contemporary popular tracks. This dataset is widely used in music recommendation research. While it is not used for this paper, it is a project worth noting as it was the primary contending dataset in consideration.

4. GATHERING DATA

4.1. Picking Dataset

Before collecting any data, the first step is to determine which dataset to use. While there are a few datasets available online, like the aforementioned Million Song Dataset, the issue with these datasets is that there is no ground truth available. The presented experiment uses content-based filtering and picking an unknown dataset will make it harder to determine the validity of the results. After looking for other datasets, I decided to use my music library.

The library contains a total of 3,440 tracks (n = 3,440) spanning 388 artists, 20 genres, and 10 languages. The languages include English, Farsi, French, Hindi, Instrumental, Japanese, Korean, Portuguese, Spanish, and Tamil. And the genres include Alternative/Indie, Country, Disco, Electronic, Film, Folk, Funk, Grime, Hip-Hop/Rap, Hyperpop, Instrumental, Jazz, Metal, Pop, Punk, R&B/Soul, Reggae, Rock, Singer-Songwriter, and Trap. Of these genres, Grime and Hyperpop are not widely known. Grime, sometimes referred to as UK Grime, is a subgenre of electronic music that is based out of London. The genre started in the early 2000s and has been growing since. As for Hyperpop, it falls under the broader genre Escape Room. According to Spotify "data alchemist" Glenn McDonald, "I made up the name myself, because I couldn't figure out any existing one to apply. The vibe is kind of an underground-trap/PC-music/indietronic/activist-hip-hop kind of thing, and I thought of "escape room" both for the sense of escaping from trap, and for the ideas of excitement, puzzle-solving and indoorness implied by the actual physical escape-room phenomenon" [2]. With this level of diversity, there should be enough songs in the dataset to provide sufficient opportunities for the recommender system. This also helps better implement a content-based filtering approach since there is a full understanding of every track in the dataset, the metadata, as well as some history behind the artists, genres, etc.

4.2. Collecting Metadata

With the dataset chosen, the first step to start the data collecting process is to gather the basic metadata. This data includes the track title, artist, album, album artist, genre(s), release year, and language(s). Not all of these attributes will be used in the recommender system, but they are required to allow the user to search for songs and interact with the system.

The metadata is collected first by visiting music files using a depth first search through the music folder. Track file paths use the following formatting pattern: Music/<Album Artist>/<Album>/<Track>. In order to distinguish audio files from non-audio files like album cover images (ending in ".png" or ".jpg") and digital album booklets (ending in ".pdf"), files ending in ".mp3", ".m4a", ".flac", and ".wav" are flagged for data collection.

The metadata is then gathered using a Python script, titled "get_metadata.py", that makes use of TinyTag [1]. TinyTag is a Python library designed to read metadata for various music file formats. Figure 1 shows a code snippet of how metadata is collected and entered into a CSV file using TinyTag.

```
file = tinytag.TinyTag.get(file_path)

wr.writerow([index, file.artist, file.album, file.title, file.year, file.genre])
```

Figure 1. Code snippet of get_metadata.py
Note. The code used to extract a track's metadata and write it to the data file.

By declaring the variable "file" and passing the file path of the track to the tinytag.TinyTag.get function, the script is able to access the metadata of the track. Each piece of metadata is accessed using the metadata's respective attribute declared in TinyTag (e.g., file.artist for the artist, file.title for the title, etc.). Those attributes are then written to a CSV file.

Note that an index attribute is added as a primary key to make sure each track is uniquely identifiable and easier to access. The index attribute gives the first song a value of zero, the next song a value of one, and keeps incrementing the value by one for every song added to the data file.

Once the data file is created, the next step is to add the language data. Because the language data is created and stored under a custom tag, the data is collected by hand and added to the data file. Songs that do not have lyrics have their language value stored as "Instrumental".

Table 1 shows a sample of the data file after the metadata is collected. Attributes with multiple values are separated using a semicolon. In this example, the semicolons mean that "Big City" by Kero Kero Bonito is both a Pop and Hyperpop song and has a mix of English and Japanese.

Table 1. Preview of aggregated metadata data file.

index	artist	album	title	year	genre	language
1715	Kenny Loggins	Yesterday, Today, Tomorrow - The Greatest Hits Of Kenny Loggins	Danger Zone	1997	Rock	English
1716	Kep1er	FIRST IMPACT	WA DA DA	2022	Pop	Korean
1717	Kero Kero Bonito	Bonito Generation	Big City	2016	Pop; Hyperpop	English; Japanese

Note. A snippet of the aggregated metadata containing the index, artist, album, title, release year, genre(s), and language(s) values.

4.3. Collecting Credited Collaborators

Many artists have other artists and producers with whom they work alongside to create a song. When people listen to those songs, they may like the song for the featured artist's performance or for the producer's production style. In order to capture this aspect, collecting information on featured artists and producers can provide insight to develop stronger recommendations.

4.3.1. Getting Featured Artists Data

To collect data on featured artists, first a column header is added to the data file called "featured_artists". Afterwards, each track's title is searched for using keywords indicating a featured artist: keywords include, but are not limited to, "ft", "feat", "featuring", "with", etc. Those tracks are then flagged, and the featured artists are extracted and entered into the "featured_artists" column in the data file.

4.3.2. Getting Producer Data

Like the featured artists, a column header is added to the data file called "producers". The first step to getting producer data is by looking for track titles that have keywords indicating a producer like "prod". However, this method returns a very limited number of files. The remaining files are then checked manually against existing music databases and the producer information is added to the data file.

Table 2. Preview of data file including featured artists and producers.

index	artist	album	title	featured_artists	producers
1817	Lady Gaga	Dawn of Chromatica	Sour Candy (with BLACKPINK) – Shygirl & Mura Masa Remix	BLACKPINK; Shygirl	Mura Masa
1818	Lamp	For Lovers	Last Train At 25 O'clock		
1819	Lana Del Rey	Blue Banisters	Arcadia		Lana Del Rey; Drew Erickson

Note. A snippet of the data table including featured artists and producers. Tracks with multiple featured artists and producers have the values separated by a semicolon. All the data originally listed in Table 1 (year, genre, language) is available in the data file but not listed in this table.

Table 2 shows an example of the track data with featured artists and producers. Tracks that did not have any publicly available information on featured artists nor producers have their respective values left blank. Note that the data file contains all the same data as Table 1, but previously listed columns are left out for simplification. With the information on featured artists and producers collected and added to the data file, the next step is to gather the user generated tags.

4.4. Collecting User Generated Tags

User generated tags are collected from Last.FM [7]. Last.FM is a web service that allows users to track their music listening history and provides statistical insight into the user's listening habits. It also serves as a social platform where users can discuss music using the "Shoutbox" feature found on the webpages of artists, albums, tracks, and user profile pages.

A plethora of information can be found within each track's Last.FM webpage. The first section of information is standard information on the track including title, artist, and album. The second section of information is service specific information such as number of unique listeners and total "scrobbles" (number of plays). Finally, the page also includes other information like the music video, links to the track on different streaming services, artist's social media links, recommended artists and songs (based on a collaborative filtering approach), and user comments in the Shoutbox.

Last.FM also has the option for users to add tags to songs to help classify them. Because Last.FM has an internal tag ranking system, the tag weights are not explicitly known, but it should not affect the data collection process. While most tags pertain to track information like genre, release year, etc., any term can be added as a tag. Most often, tags of niche genres (e.g. hyperpop, escape room, beardcore, etc.) are found towards the middle of the list of tags. Otherwise, tags can also relate to moods (happy, sad), ratings (7 out of 10, Pitchfork Top 100 Tracks of 2007, best song ever), or other characteristics (space music, annoying, energetic). These tags provide insight on what users think about a song.

Using a Python script, a list of tracks and their associated artist is created from the CSV file. The script then connects to Last.FM's API and searches for the track. Last.FM's autocorrect featured is enabled when searching for a track to mitigate any typing errors or disputes in the formatting. The getTopTracks function is then used to request a JSON file from Last.FM which contains a list of the track's tags as well as a "count" value. This count value is an internal ranking system created by Last.FM. However, no explicit documentation was found on how the count value is determined.

Once the JSON file is returned, all the tags are then collected and sanitized. To sanitize the tags, the characters are switched to all lower-case letters, spaces are removed, and the character set only allows for a-z and 0-9 (thus removing any special characters). Afterwards, any repeated tags are removed, and the remaining tags are written to the CSV file under the column "tags".

Disclaimer:

- All data collected from Last.FM is using the "royalty-free, non-exclusive licence" listed under Last.FM's API Terms of Service.
- All data collected through Last.FM's API is the property of Last.FM.
- All data collected is under the "Reasonable Usage Cap" of 100 MB. The total data collected amasses less than 2 MB.
- At no point was any information that could potentially identify a Last.FM user accessed, stored, published, distributed, or made available in any form.

4.5. Additional Sanitization Processing

Once all the data is collected and stored in the "data.csv" file, some additional processing needs to be done. The reasoning will be discussed later, but all spaces in the artist's name is replaced with underscores and all semicolons indicating multiple values in one cell are removed and replaced with spaces.

5. CODE AND MATH EXPLANATION

5.1. Terminology

In order to give song recommendations, the recommender system will make use of a frequency matrix and cosine similarities. Before explaining how the frequency matrix and cosine similarities work, the following terms should be noted:

- Attribute – A particular quality or characteristic of a track. In this case, the attributes consist of each piece of metadata as well as the list of tags (not the individual tags themselves). With respect to the data table, attributes are the columns.
- Value – The value(s) of an attribute for a track. This will be the actual genre(s), tag(s), producer(s), etc. With respect to the table, this is the data stored in a specific cell.
- Document – Because the audio file itself cannot be compared using cosine similarities in its current state, the document stands in as an easily comparable, text-based substitute that holds information about the track.
- Vector – An object containing both magnitude and direction. Typically, vectors are displayed as line segments with an arrow indicating the vector's direction. The dimensionality of a vector is directly related to the number of elements in the vector. While two- and three-dimensional vectors can be easy to plot and visualize, higher dimensional vectors require other methods to extract information. Figure 2 shows an example of a two-dimensional vector.
- Vector Space – Given a set of vectors, the vector space is all the points that can be reached in an area by adding the vectors and using scalar multiplication to increase/decrease the magnitude and change direction.

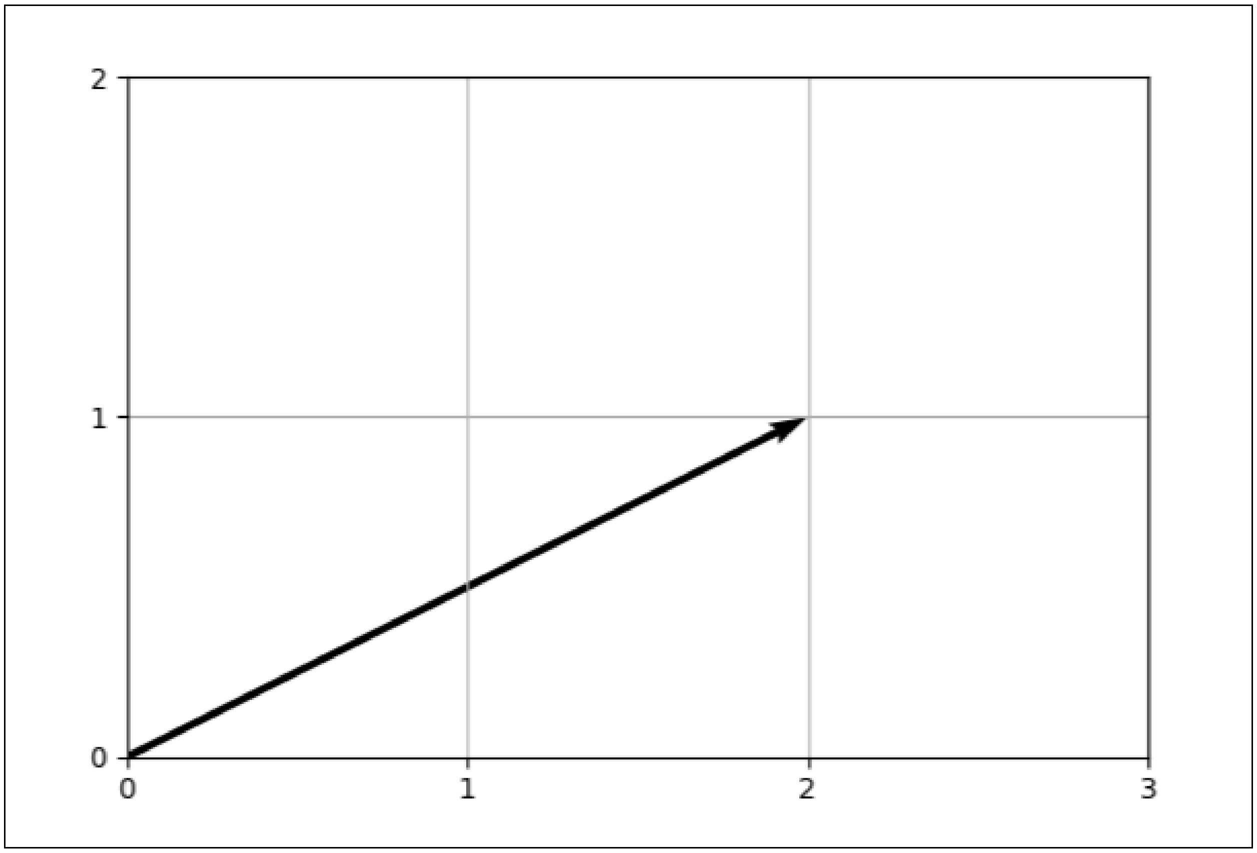

Figure 2. Example vector.
Note. Example vector $\vec{v}$ in which $\vec{v} = \langle 2, 1 \rangle$. $\vec{v}$ is a two-dimensional vector [4].

5.2. Frequency Matrix

A frequency matrix is an m×n matrix containing whole numbers that contains the frequency of a value in various documents. With respect to this experiment, the rows represent each track's document and m is the number of rows in the frequency matrix. Meanwhile, the columns represent each unique value that appears across all attributes and n is the number of columns in the frequency matrix. Attributes that can hold multiple values are now split so that each unique value gets its own column. The elements in the matrix represent the frequency at which the column value appears in the row document.

5.3. Document Creation

In order to create the frequency matrix, the first step is to convert each track in the CSV file into a document. In this case, the document will be a string of values. Depending on how data is being processed, a predetermined list of attributes is decided. This list is known as the combined feature set. So, if a tag-based recommendation is being evaluated, then only the tags will be passed to the document creation function, and if a metadata-based recommendation is being calculated, then only the metadata attributes will be passed to the document creation function. With the attributes selected, the program passes the predetermined attributes, on a per row basis (i.e., per track basis), to a function which then concatenates the string form for each value (adding spaces between attributes) and returns the concatenated string. That returned string is now the track's document. The documentation creation functions and respective descriptions are as so:

- combine_features_meta – combines metadata attributes
- combine_features_tags – combines tags
- combine_features_top_tags – combines metadata with the top five tags
- combine_features_top_tags_no_language – combines metadata (excluding language) with top five tags
- combine_features_all_tags – combines metadata and all tags

As mentioned during Section 4.5, attributes with multiple values are sanitized and stored so that each value has a space in between each other. Individual values that originally had spaces (e.g., artists who simply go by their first and last name) have their spaces replaced with underscores so that the whole value is identified as one word. This will make it so that these values are identified as unique in the frequency matrix. One example is if the value for artist Kanye West was stored as "Kanye West", then the frequency matrix would create a separate column for "Kanye" and another column for "West". After sanitizing, "Kanye West" is converted to "kanye_west" making it so that the frequency matrix only has a column for the value “kanye_west”. Another example is if there are two producers who worked together to produce a track, the existing space character stored between their names in the producer attribute column of the data file will be processed so that that space is included in the document. So, when “Logic 6ix” gets passed as the producer value to the document creation function, it converts it to “logic 6ix ” (note the extra space at the end). This makes it so that there is an individual column for “logic”, one for “6ix”, and one for the value that is added after the producers in the matrix. Figure 3 shows the code for the function combine_features_tags.

```
def combine_features_tags(row):
    combined_features = ""
    if(len(str(row['tags'])) > 0):
        combined_features = combined_features + row['tags'].lower() + " "
    return combined_features
```

Figure 3. Code snippet of combine_features_tags function
Note. The code used for combine_features_tags that gets passed the tag values for a track and converts them into a text based document for the frequency matrix.

Once all the documents are created, the next step is to create the frequency matrix. This is done using SciKit Learn, a Python library containing various tools for data analysis [3, 6]. Using the fit_transform function from the CountVectorizer class, the frequency matrix is created by gathering all unique values across all documents, then looking at how often each value appears in a document and storing that frequency count in its respective element. Figure 4 shows the code used to create the frequency matrix. The first line in the code creates a ‘combined_features’ column to hold the song documents and saves the output from Figure 3 for each track. The parameter “axis = 1” means that this will be applied to each column in the row. After instantiating the CountVectorizer class, the next step is to use the fit_transform function to get the frequency matrix from the “combined_features” row that was created.

```
df['combined_features'] = df.apply(combine_features_tags, axis=1)

cv = CountVectorizer()

count_matrix = cv.fit_transform(df["combined_features"])
```

Figure 4. Code snippet of document and frequency matrix creation.
Note. The code used to apply the document creation function to each track, then converting those documents into a frequency matrix.

5.4. Cosine Similarity

5.4.1. Cosine Similarity Explanation

With the frequency matrix created, the next step is to determine which songs to recommend. One method to determine recommendations is using cosine similarity. Cosine similarity is the cosine of the angle between two vectors. The range of a cosine similarity can span [-1, 1] in which -1 signifies perfectly opposite facing vectors, 0 signifies orthogonal (right angle) vectors, and 1 signifies two vectors with the same direction. Cosine similarity solely looks at the angle between two vectors: the magnitude of the vectors can be different. Equation 1 shows the mathematical formula to determine the cosine similarity. Equation 2 and Equation 3 are supporting equations needed to perform Equation 1.

$$\cos\Theta = \frac{\mathrm{A}\cdot\mathrm{B}}{\|A\|\|B\|} \qquad \text{(Equation 1)}$$

$$\mathrm{A}\cdot\mathrm{B} = \sum_{i=1}^{n} a_i b_i \qquad \text{(Equation 2)}$$

$$\|A\| = \sqrt{\sum_{i=1}^{n} {a_i}^2} = \sqrt{{a_1}^2 + {a_2}^2 + \cdots + {a_n}^2} \qquad \text{(Equation 3)}$$

In these equations, A and B are frequency vectors and cos Θ is the cosine similarity. The numerator of Equation 1 is the dot product of vectors A and B (see Equation 2). Meanwhile, the denominator is the product of the Euclidian norm of A and the Euclidian norm of B (see Equation 3). The Euclidian norm of a vector signifies the distance from the origin to the end point. Equation 1 does not work if either vectors, A or B, are zero vectors (i.e., every element in the vector is zero). This is because the Euclidian norm of a zero vector is zero and if either vector is a zero vector, then the denominator of the Equation 1 is zero, making the answer undefined.

Another perspective is an angle requires two lines to create it; with a zero vector, only one line would exist. Figure 5 demonstrates this issue by showing a simplified frequency matrix that only contains the frequencies for the words “pop” and “english”. By picking any combination of two tracks between Track 1, Track 2, and Track 3, an angle forms that can be used to determine the cosine similarity. However, using any track along with Track 4 (yellow dot at the origin) does not create an angle because Track 4 is a zero vector.

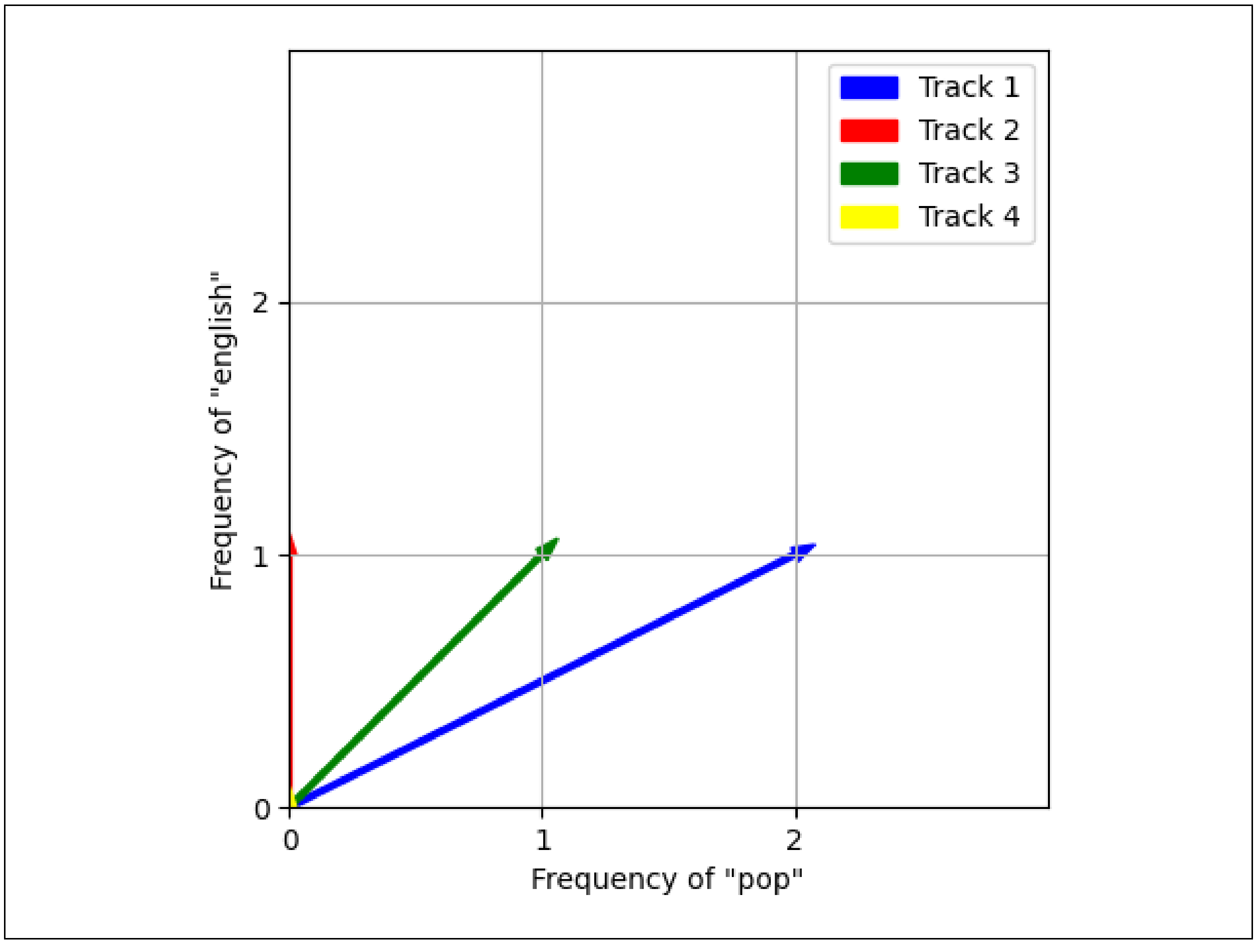

Figure 5. Vector comparison.
Note. Selecting two frequency vectors from Tracks 1, 2, and 3 form an angle. However, using any vector with Track 4’s vector (note the yellow dot at the origin) does not form a vector [4].

Because this formula does not work using zero vectors and the frequency values can be zero, it is possible that a frequency vector could be a zero vector. However, this is not a concern when dealing with any recommendations that include metadata in the document. Because every track in the dataset at least has an artist associated to it, the frequency for the artist's value will be one. Assuming all other attributes have a frequency of zero, at least the artist will have a frequency of one, thus the vector will be non-zero. However, when dealing with only tags in the document, it is possible for the frequency vector to be a zero vector when there are no tags associated with the track. The results of this scenario are discussed in Section 6.7.

Finally, for this experiment, since value frequencies cannot be negative (i.e., a word cannot appear negative times in a document), the angle between the two vectors can at most be 90 degrees. This means that the cosine similarity's range now spans [0, 1] because the angle range decreases from [0, 180] to [0, 90].

5.4.2. Cosine Similarity Processing

Once the frequency matrix is created, the next step is to calculate the cosine similarities and determine the recommendations. Figure 6 shows the code used to determine the cosine similarities, using the consine_similarity function in SciKit Learn, of the user-entered track against every track (including itself). The recommendations are then determined by sorting the tracks in order of cosine similarity values from highest to lowest. A higher cosine similarity indicates more overlap between the songs' metadata and/or tags, thus being a stronger recommendation.

```
cosine_sim = cosine_similarity(count_matrix)

similar_songs = list(enumerate(cosine_sim[song_index]))

sorted_similar_songs = sorted(similar_songs, key=lambda x: x[1], reverse=True)
```

Figure 6. Code snippet of cosine similarity calculation and recommendation sorting.
Note. The code used to determine and sort the cosine similarity values of each track against the track located at index ‘song_index’.

6. RESULTS

6.1. Creating Recommendation Request

In order to retrieve recommendations for similar songs, the user first runs the Python script. The script prompts the user to enter the artist's name, then the track title. Once it collects the necessary information, it locates the song's index and performs the calculations. After the calculations are completed, a list of every song is returned sorted by cosine similarity (highest to lowest), then by artist (alphabetically), then by track title (alphabetically). The top 10 tracks are then returned to the user. Exceptions on determining the top 10 recommendations are discussed in Sections 6.2 and 6.3.

6.2. Artist Filtering

Now looking at the first set of results, the song "LAST WALTZ" by TWICE is entered into the system. Table 3 shows the recommendations for "LAST WALTZ" based on metadata while Table 4 shows the recommendations based on tags.

Table 3. Recommendations for "LAST WALTZ" by TWICE using metadata

Artist	Track	Cosine Similarity
TWICE	LAST WALTZ	1.000
TWICE	1, 3, 2 (JEONGYEON, MINA, TZUYU)	0.9129
TWICE	PUSH & PULL (JIHYO, SANA, DAHYUN)	0.9129
TWICE	CACTUS	0.8333
TWICE	CANDY	0.8333
TWICE	REWIND	0.8333
TWICE	SCIENTIST (R3HAB Remix)	0.8333
TWICE	ESPRESSO	0.7715
TWICE	F.I.L.A (Fall In Love Again)	0.7715
TWICE	ICON	0.7715

Table 4. Recommendations for “LAST WALTZ” by TWICE using tags

Artist	Track	Cosine Similarity
TWICE	BRING IT BACK	1.0000
TWICE	ICON	1.0000
TWICE	LAST WALTZ	1.0000
TWICE	ESPRESSO	0.8944
TWICE	REWIND	0.8944
TWICE	HANDLE IT	0.8660
TWICE	REAL YOU	0.8660
TWICE	SAY YES	0.8660
TWICE	STUCK	0.8660
TWICE	First Time	0.8660

Looking at these unaltered results, the first issue is the model prefers to recommend the same artist as the user-entered track. The results listed above for “LAST WALTZ” by TWICE show that the model recommends only TWICE songs, and in both cases, “LAST WALTZ” itself is recommended with a cosine similarity of one.

The issue of the same artist being recommended can, in part, be explained from a mathematical perspective. When the frequency matrix is created, the value of the matrix element for the artist value will be at least one. With respect to the example above, during the frequency matrix construction, every track made by TWICE will have a value of at least one in the matrix element pertaining to the track and the column for the value “twice”. Because these songs share the same artist and the “twice” vector element for all their songs is at least one, the vectors will be closer together, the angle between the vectors will be smaller, and the cosine similarity will be higher. Extrapolated from there, since the model compares the user-entered track against all songs, including the original song, the user-entered track will have the exact same vector as itself, thus making it the strongest recommendation.

The other part is to look at this situation from a social standpoint. People typically have others that they enjoy working with and musicians are no exception. It is not uncommon to see an artist work with one producer to make an entire album. One example is the album “No Pressure” by Logic in which a significant portion of the production is handled by Logic and 6ix. Another example is the album “Shabrang” by Sevdaliza where Sevdaliza works alongside producer Mucky for most of the tracks. In these cases, it is likely that the artist and producer have developed a relationship and that their collaborative works can span multiple albums. This also further explains the strength of the cosine similarity since not only is the frequency of the artist’s value greater than zero, but the producer’s as well.

Finally, this idea can be extracted to genres and languages as well. Most artists tend to stay within their genre and rarely venture out, and very few artists make songs in more than one language. Therefore, the genre and language values are also the same across most of an artist’s discography. All these factors combined make it so that the vectors of songs pertaining to one artist are closer together than those of other artists.

In order to increase the recommendation diversity, the following steps are taken:

- When a user enters a song, it will be assumed that the user is already familiar with the primary artist’s discography.
- Recommendations containing the same primary artist will be skipped.

Now that the same primary artist filter assumptions have been establish, running the recommender system again and skipping other songs by TWICE yield the following results:

Table 5. Recommendations for “LAST WALTZ” by TWICE using metadata.

Artist	Track	Cosine Similarity
Red Velvet	Better Be	0.7303
Red Velvet	Hello, Sunset	0.7303
Red Velvet	Knock On Wood	0.7303
Red Velvet	Pose	0.7303
Red Velvet	Pushin’ N Pullin’	0.7303
aespa	Next Level	0.6667
LISA	LALISA	0.6667
CHUNG HA	Bicycle	0.6172
Red Velvet	Queendom	0.6172
ITZY	#Twenty	0.6124

Note. Same artist recommendation filter applied.

Table 6. Recommendations for “LAST WALTZ” by TWICE using tags.

Artist	Track	Cosine Similarity
ITZY	Mirror	0.5000
ITZY	TENNIS (0:0)	0.5000
Red Velvet	Pose	0.5000
ITZY	#Twenty	0.3536
ITZY	Chillin' Chillin'	0.3536
ITZY	Gas Me Up	0.3536
ITZY	LOVE is	0.3536
TAEYEON	Four Seasons	0.3536
TAEYEON	What Do I Call You	0.3536
CHUNG HA	Bicycle	0.2887

Note. Same artist recommendation filter applied.

Looking at the results in Table 5, Red Velvet is recommended the most. This is likely due to the genre and language metadata being the same (Pop and Korean respectively). Meanwhile, Table 6 looks at recommendations based on tags. In these results, ITZY is recommended most often. This is because ITZY and TWICE work under the same entertainment and record label, JYP Entertainment.

The cosine similarities in Table 5 are, in general, higher than Table 6. This is because when dealing with metadata, there are fewer unique values, so there are fewer columns in the frequency matrix. This does not necessarily mean that the recommendations in Table 6 are worse than Table 5; it means that given the respective attributes, the associations using tags are generally weaker because there are more unique values to consider when dealing with tags. This idea of tag saturation is discussed further in Section 6.4.4

Table 7 shows the recommendation results for “LAST WALTZ” when combining metadata and tags into the document. When combined, the results show a mix of the results when using metadata and when using tags. However, compared to Table 5 and Table 6, Table 7 has more diversity in recommendations along with cosine similarities generally falling in between the metadata and tags separately.

Table 7. Recommendations for “LAST WALTZ” by TWICE using metadata and tags combined.

Artist	Track	Cosine Similarity
Red Velvet	Pose	0.6455
ITZY	Mirror	0.5657
Red Velvet	Better Be	0.5657
ITZY	Chillin’ Chillin’	0.5590
ITZY	LOVE is	0.5590
CHUNG HA	Bicycle	0.5477
ITZY	Sooo LUCKY	0.5270
Red Velvet	Hello, Sunset	0.5164
Red Velvet	Pushin’ N Pullin’	0.5164
aespa	ICONIC	0.5078

Note. Same artist recommendation filter applied.

6.3. Year to Decade Conversion

The second alteration in the results filtering is to change the release year attribute into the release decade. Because the release year is the only quantitative variable in the dataset, the number of unique year values oversaturates the frequency matrix. Categorizing the year and making it qualitative aligns it with the rest of the attributes in terms of the number of unique values associated with the attribute. Besides, if an artist releases one album one year, and releases another album the next year, does not mean that the albums are starkly different. Reclassifying the release year to the release decade will help smooth out any biases in specific years that contain more songs than its surrounding years.

Ideas in music also take time to develop and sometimes these ideas are revisited and redeveloped in “eras”. For example, while disco was popular in the 1970s with bands like ABBA, Bee Gees, Boney M., and Earth Wind & Fire, the genre has recently been undergoing revitalization. Known as nu-disco, albums like Dua Lipa’s Future Nostalgia, The Weeknd’s Blinding Lights, and Róisín Murphy’s Róisín Machine have been revisiting and expanding on the same ideas that were used nearly fifty years ago. While there can be multiple eras for the same musical ideas, the time within these eras usually span consecutive years and should be taken into consideration. Using the release decade will help by giving each era a span of 10 years to develop.

Besides, categorizing albums and tracks by what decade they were released has been a popular method of classifying music. Music streaming giant Spotify has curated several playlists – often paired with a certain genre – dedicated to specific decades [8]. Examples of these playlists include “All Out 80s”, “00s Rock Anthems”, “2010s Country”, and “I Love My ‘70s Funk”.

In order to lessen the strength that the release year has on the recommendations, the calculations to determine the release decade are made during the document creation process. When the track’s year value is passed to the document creation function, the year is divided by 10, then using the math.trunc function, the digits after the decimal point are removed. Because the digits are simply removed, the number is always rounded down. For example, 1975 will become 197.5 and rounded to 197, 1997 will be rounded to 199, and 2012 will be rounded to 201. The rounded decade value is then added to the document and given a column in the frequency matrix. Figure 7 shows how this code is implemented in the combine_features_meta function.

```
if(len(str(row['year'])) > 0):
    combined_features = combined_features + \
        str(math.trunc(row['year']/10)) + " "
```

Figure 7. Code snippet for decade conversion.
Note. The code used to convert the release year value into the release decade.

To demonstrate the effects of this conversion, we will look at the song “Summer Of ’69” by Bryan Adams from the album “Reckless”. “Reckless” was released on November 5, 1984. Table 8 shows the recommendations for “Summer Of ’69” using metadata with the year value set to “1984”.

Table 8. Recommendations for “Summer Of ‘69” by Bryan Adams using metadata.

Artist	Track	Cosine Similarity
Van Halen	Hot for Teacher - 2015 Remaster	0.3273
Van Halen	Panama - 2015 Remaster	0.3030
Dion	Dream Lover	0.2536
Dion	Kansas City	0.2536
Dion	Life Is But A Dream	0.2536
Dion	Little Star	0.2536
Dion	Lonely World	0.2536
Dion	Runaround Sue	0.2536
Dion	Runaway Girl	0.2536
Dion	Somebody Nobody Wants	0.2536

Note. Same artist recommendation filter applied. 1984 used as year value.

In these results, both Van Halen songs are from the same album. Despite being remastered songs, the album is titled “1984” and was originally released in the year 1984. Therefore, both Van Halen songs and “Summer Of ’69” have a non-zero value in their respective elements under the column for “1984” in the frequency matrix.

Afterwards, the remaining recommendations are all derived from Dion’s 1961 release “Runaround Sue”. Because the release year holds little value compared to the other attributes, two albums from entirely different decades are recommended so closely together. “Summer Of ’69” is a pop rock song that was released during the era of rock, metal, and pop rock in the 1980s while “Runaround Sue” is a doo-wop album released during the 1960s. While doo-wop and pop rock share a few characteristics, they come from different eras.

Table 9 shows the recommendations for “Summer Of ’69” after converting the release year to the release decade. The decade value is now stored as “198”.

Table 9. Recommendations for “Summer Of ‘69” by Bryan Adams using metadata.

Artist	Track	Cosine Similarity
Joan Jett & The Blackhearts	Crimson and Clover	0.4743
Joan Jett & The Blackhearts	You Don't Know What You Got [Live]	0.4743
Van Halen	Hot for Teacher – 2015 Remaster	0.4743
Pixies	Gigantic – Remastered	0.4330
The Bangles	Walk Like an Egyptian	0.4330
Van Halen	Panama – 2015 Remaster	0.4330
a-ha	Take on Me	0.4009
Joan Jett & The Blackhearts	I Love Rock ‘N’ Roll	0.4009
Pixies	Debaser	0.4009
Pylon	Altitude - Remastered	0.3030

Note. Same artist recommendation filter applied. 198 used as decade value.

Once the year was converted to the decade, Joan Jett & The Blackhearts’ album “I Love Rock ‘N’ Roll” takes over the recommendation list. “I Love Rock ‘N’ Roll” came out in 1981 and falls under the punk rock genre. “Surfer Rosa” by Pixies (released in 1988) also falls under the rock adjacent genre of alternative rock or indie rock. Meanwhile, Van Halen appears again with a higher cosine similarity as before. a-ha’s “Hunting High and Low” (released in 1985) is often seen as a poster child when thinking about 1980s pop. Finally, Pylon’s “Chomp” (released in 1983) falls under the post-punk and alternative rock genres too.

Now, with this alteration, the cosine similarities are higher and the spread in recommended song’s album releases narrows from 1961-1984 to 1981-1988. This gives the opportunity to reflect the ideas that were developed during the 1980s. Using the decade over the year allows for a better representation of eras and better usability of the year attribute.

With these two modifications added and the reasoning behind each explained, the next step is to compare the strength that different combined feature sets have on providing recommendations.

6.4. Test Results: Electro-pop Recommendations

6.4.1. Metadata Recommendations

The first song we will look at is “Get Lucky” by Daft Punk. Daft Punk was a French music group consisting of two members, Thomas Bangalter and Guy-Manuel de Homem-Christo. They primarily made electronic, house, dance, and disco music. Despite the group’s disbandment in 2021, their outreach is considered significant amongst the music community.

Daft Punk’s contribution on “Get Lucky” is mainly on production while Pharrell Williams is featured on the song as the main singer. Most commonly known for his #1 US Billboard Hot 100 super hit “Happy” from the Despicable Me 2 soundtrack, Pharrell is a rapper, producer, singer, and songwriter too.

Table 10. Recommendations for “Get Lucky” by Daft Punk using metadata.

Artist	Track	Cosine Similarity
Clean Bandit	Rather Be (feat. Jess Glynne)	0.6172
Flight Facilities	Crave You (feat. Giselle Rosselli)	0.6172
Icona Pop	I Love It (feat. Charli XCX)	0.6172
M.I.A.	BORN FREE	0.6172
M.I.A.	TEQKILLA	0.6172
M.I.A.	aTENTion	0.6172
Mike Posner	I Took a Pill In Ibiza [Seeb Remix]	0.6172
The Knocks	Love Me Like That (feat. Carly Rae Jepsen)	0.6172
Disclosure	Latch (feat. Sam Smith)	0.5714
M.I.A.	Bring The Noize	0.5714

Note. Same artist recommendation filter applied. Release year converted to decade value.

Table 10 shows the recommendation results for “Get Lucky” using metadata. Looking at the results, a pattern emerges; like “Get Lucky”, most of the recommended songs are electronic songs featuring a pop artist. In this case, Clean Bandit, Flight Facilities, Icona Pop, The Knocks, and Disclosure are similar to Daft Punk because they primarily create electronic music and typically do not participate in vocal performances. Likewise, Jess Glynne, Giselle Rosselli, Charli XCX, Carly Rae Jepsen, and Sam Smith perform the vocals in their respective songs like Pharrell Williams.

The recommendations that do not follow the electronic and pop singer formula are the M.I.A. songs and “I Took a Pill In Ibiza [Seeb Remix]” by Mike Posner. M.I.A. is a solo artist whose discography typically consists mostly of hip-hop or pop verses over electronic beats. While it does not fit the previously mentioned formula, it serves as another valid recommendation for its overlap in pop and electronic genres. Meanwhile, Mike Posner’s “I Took a Pill In Ibiza [Seeb Remix]” flips the formula; Instead of an electronic artist reaching out to a pop artist for vocals, Norwegian electronic duo Seeb remixed the original Mike Posner pop song “I Took a Pill In Ibiza”.

The cosine similarities for these recommendations hover between 0.57 and 0.62. Calculating the inverse cosine shows the angles between the frequency vectors for “Get Lucky” and each recommendation span approximately 52 to 55 degrees. These cosine similarities are relatively high meaning the metadata values of these songs have a substantial overlap with “Get Lucky”.

6.4.2. Tag Recommendations

Before looking at the tag-based recommendations, it is worth looking at the tags associated with “Get Lucky”. “Get Lucky” has a total of 98 tags. Some of the top tags associated with “Get Lucky” are “electronic”, “disco”, “funk”, “dance”, “house”, “2013”, “daft punk”, “pop”, “10s”, “French”, “Pharrell Williams”, “summer”, and “nu-disco”. These first few tags generally focus on genres, subgenres, and primary characteristics of the song.

Further down the list appear other tags like “boogie boogie woogie”, “sachen die ich gerne hoeren mag” (translated from German to English means “things I like to hear”), “anthemic”, “danceable”, “male vocalist”, and “space music”. These tags are generally ways Last.FM users categorize songs using their own personal methods. Using the “count” tag ranking system, these tags provide a social aspect to the recommendation. If one Last.FM user is to agree with one of these tags, and by association the user agrees with the user who added that tag to the track, it is likely that the first user would like other songs with that tag. Other songs with the same tag could have also been classified by the same user who added that tag to the original song, thereby increasing the collaborative aspect of using tags. Table 11 shows the recommendation results for “Get Lucky” using only the user-generated tags.

Table 11. Recommendations for “Get Lucky” by Daft Punk using tags.

Artist	Track	Cosine Similarity
Gorillaz	DARE (feat. Shaun Ryder of Black Grape)	0.3219
Justice	D.A.N.C.E.	0.3146
Deee-Lite	Groove Is in the Heart	0.3093
Moloko	Sing It Back	0.3066
Nicki Minaj	Starships	0.2813
Dev	In The Dark	0.2769
Icona Pop	I Love It (feat. Charli XCX)	0.2723
Michael Jackson	Billie Jean	0.2644
Jennifer Lopez	On The Floor (feat. Pitbull)	0.2591
Sublime	Doin’ Time	0.2577

Note. Same artist recommendation filter applied. Release year converted to decade value.

Once again, the cosine similarity values when using tags are generally lower than when using metadata. The angles span approximately 71 to 76 degrees. This is because the tags further down the list do not appear very often in other songs but create an additional column in the frequency matrix. Because the frequencies of those lesser used tags across most tracks is zero, adding them to the frequency matrix increases the angle between the vectors.

Now looking at the recommendations themselves, “DARE (feat. Shaun Ryder of Black Grape)”, “D.A.N.C.E.”, “Groove Is in the Heart”, “Sing It Back”, and “I Love It (feat. Charli XCX)” follow the previously mentioned pattern of electronic artist working with a pop artist for vocals. Meanwhile, Nicki Minaj is a rapper, but sometimes creates pop rap tracks; “Starships” is one of those pop rap tracks. It reflects the 2012 music landscape in which it maintains hints of electropop. Similar artists of the time include Ellie Goulding, Kesha, Sky Ferreira, Marina & The Diamonds, and Owl City. While not as rooted in the electronic genre as “Get Lucky”, it is a valid recommendation based on genre overlap. Further down the list are other pop songs. Finally, “Doin’ Time” by Sublime is a ska song and is the first song on this list that is neither electronic nor pop nor any adjacent genres.

6.4.3. Metadata and Tags Recommendations

Next let us look at the recommendations when combining metadata and tags. Table 12 shows the recommendations for “Get Lucky” with both metadata and tags in the document.

Table 12. Recommendations for “Get Lucky” by Daft Punk using metadata and tags.

Artist	Track	Cosine Similarity
Justice	D.A.N.C.E.	0.3577
Moloko	Sing It Back	0.3462
Deee-Lite	Groove Is in the Heart	0.3436
Gorillaz	DARE (feat. Shaun Ryder of Black Grape)	0.3402
Dev	In The Dark	0.3258
Icona Pop	I Love It (feat. Charli XCX)	0.3243
Rina Sawayama	Comme des Garçons (Like the Boys)	0.3125
Moloko	Cannot Contain This	0.3111
Jessie Ware	Ooh La La	0.3024
Jennifer Lopez	On The Floor (feat. Pitbull)	0.3014

Note. Same artist recommendation filter applied. Release year converted to decade value.

While the recommendations take from both the metadata-based recommendations and the tag-based recommendations, the cosine similarity values for the top recommendations (0.3014 – 0.3577) sit closer to the tag-based recommendations (0.2577 - 0.3219) than the metadata-based recommendations (0.5714 – 0.6172). While both recommendation methods seem to give valid results, combining them seems to increase the diversity of the recommendations but decrease the strength.

6.4.4. Tag Saturation

Next, we investigate why the cosine similarity values when using tags is much lower than when using metadata. The issue with the metadata and tag combination from Table 12 is that towards the bottom of the list of tags, some tags for “Get Lucky” such as “1”, “50”, “uu”, “na”, and “get” seem arbitrary. It could be that these tags were accidentally added or that users added them to dilute the tags associated with the song. While these tags add additional columns to the frequency matrix, it is also possible that these tags appear on unrelated songs that could be classified as bad recommendations. To combat this scenario, the results are calculated again using metadata along with the top five tags for each track. If a track has less than five tags, then all existing tags for that song are used. Table 13 shows the recommendations for “Get Lucky” using metadata along with the top five tags.

Table 13. Recommendations for “Get Lucky” by Daft Punk using metadata and top five tags.

Artist	Track	Cosine Similarity
Icona Pop	I Love It (feat. Charli XCX)	0.5809
Sophie Ellis-Bextor	Murder On the Dance Floor	0.5809
Disclosure	F For You	0.5809
Clean Bandit	Rather Be (feat. Jess Glynne)	0.5809
Mike Posner	I Took a Pill In Ibiza [Seeb Remix]	0.5669
Yerin Baek	“HOMESWEETHOME”	0.5669
Jessie Ware	Adore You	0.5625
Jessie Ware	Ooh La La	0.5625
Jessie Ware	Read My Lips	0.5625
Jessie Ware	Spotlight	0.5265

Note. Same artist recommendation filter applied. Release year converted to decade value.

These results look at the niche genres aligned with “Get Lucky” and so three recommendations fit the electronic and pop artist formula, “I Love It (feat. Charli XCX)”, “Rather Be (feat. Jess Glynne), and “I Took a Pill In Ibiza [Seeb Remix]”. “F For You” by Disclosure is similar to Daft Punk in its metadata, but it is missing the pop aspect like “Latch (feat. Sam Smith)” had with Sam Smith’s feature. Regardless, both “F For You” and “Latch (feat. Sam Smith)” would work as valid recommendations as they both come from the same album and share many characteristics. “Murder On the Dance Floor” and Jessie Ware’s songs are synthpop songs. Synthpop naturally has some electronic elements in it with its use of synthesizers. Finally, ““HOMESWEETHOME”” by Yerin Baek also shares some qualities as its tags are “electronic”, “dance”, and “house”. The cosine similarities for these recommendations sit closer to the metadata-based values than the tags-based and combined values.

6.5. Test Results: Hip-Hop/Rap Recommendations

6.5.1. Metadata Recommendations

In this next example, we look at the song “Mercy.1” by Kanye West. This song features rappers Big Sean, Pusha T, and 2 Chainz. The former two featured artists were signed under Kanye West’s label Getting Out Our Dreams (G.O.O.D.) Music. Table 14 shows the metadata-based recommendations for “Mercy.1”.

Table 14. Recommendations for “Mercy.1” by Kanye West using metadata.

Artist	Track	Cosine Similarity
Pusha T	What Would Meek Do? (feat. Kanye West)	0.8452
2 Chainz	Birthday Song (feat. Kanye West)	0.8081
Pusha T	Come Back Baby	0.8081
Pusha T	If You Know You Know	0.8081
Pusha T	Infrared	0.8081
Pusha T	The Games We Play	0.8081
Big Sean	All Your Fault (feat. Kanye West)	0.7715
J. Cole	Looking For Trouble (feat. Kanye West, Big Sean, Pusha T & CyHi Da Prince)	0.7606
Big Sean	Marvin & Chardonnay (feat. Kanye West & Roscoe Dash)	0.7559
Pusha T	Santeria	0.7559

Note. Same artist recommendation filter applied. Release year converted to decade value.

Looking at the results, Pusha T manages to appear most often in the list. The reason is because all the Pusha T songs listed are from the same album, “DAYTONA”, in which Kanye West was the executive producer. Compared to “Mercy.1”, “DAYTONA” was released in the same decade and shares a lot of characteristics in terms of genre and contributors.

Looking at the cosine similarities, the first recommendation is higher than the rest. With a cosine similarity of 0.8452, the angle between “Mercy.1” and “What Would Meek Do? (feat. Kanye West)” is about 32.3 degrees. This angle is substantially lower than the results from

Section 6.4.1. The reason is because not only is Kanye West the producer of the track, but Kanye also has a verse, making him a featured artist. Because Kanye West is the producer and a featured artist, the element value for “kanye_west” in the frequency matrix is two. Similarly, because Kanye West is the primary artist and producer for “Mercy.1”, the element value for “kanye_west” in the frequency matrix for “Mercy.1” is two as well. This is one aspect which brings the vectors for “What Would Meek Do? (feat. Kanye West)” and “Mercy.1” closer, increasing the cosine similarity value.

The remaining Pusha T recommendations (except for “Santeria”) have a cosine similarity of 0.8081 with the angle equating to approximately 36 degrees. The reason why “Birthday Song” by 2 Chainz has the same cosine similarity as the other Pusha T tracks is because the value for “kanye_west” in the frequency matrix is one because Kanye West produced Pusha T’s tracks and Kanye West had a featured verse on “Birthday Song”.

Meanwhile, the reason behind why “Santeria” has a lower cosine similarity than the other Pusha T recommendations is because while much of the metadata is the same, the song contains both English and Spanish lyrics. Because the value for “spanish” is one for “Santeria” and zero for the other Pusha T songs, the angle between “Mercy.1” and “Santeria” is greater than “Mercy.1” and the other Pusha T songs. Section 6.6 further discusses the effects language has on recommendations.

6.5.2. Tag Recommendations

Next, we will look at the tag-based recommendations for “Mercy.1”. The tags collected by Last.FM are “trap”, “Big Sean”, “Pusha T”, and “2 Chainz”. Table 15 shows the top recommendations for “Mercy.1” using tags.

Table 15. Recommendations for “Mercy.1” by Kanye West using tags.

Artist	Track	Cosine Similarity
Big Sean	Overtime	0.5000
Freddie Gibbs	Palmolive (feat. Pusha T & Killer Mike)	0.5000
Logic	State Of Emergency (feat. 2 Chainz)	0.5000
MadeinTYO	Mr. Tokyo	0.5000
Migos	White Sand (feat. Travis Scott, Ty Dolla $ign & Big Sean)	0.5000
Big Sean	Sacrifices (feat. Migos)	0.3536
Freddie Gibbs	4 Thangs (feat. Big Sean & Hit-Boy)	0.3536
G-Eazy	One Of Them (feat. Big Sean)	0.3536
Logic	Wrist (feat. Pusha T)	0.3536
Logic	Wassup (feat. Big Sean)	0.3536

Note. Same artist recommendation filter applied. Release year converted to decade value.

While most of these results make sense through mutual artists who worked on “Mercy.1’, “Mr. Tokyo” by MadeinTYO has no overlap in artists. Looking at the tags for “Mr. Tokyo”, the only tag associated with the song is “trap”. Since there are so few tags to work with in “Mercy.1”, the cosine similarity values are higher than other results.

6.5.3. Metadata and Top Tags Recommendations

Next is to look at the metadata and the top five tags combined. Since “Mercy.1” only has four tags, all tags for “Mercy.1” will be included. However, other tracks with more than five tags will be substituted. Table 16 shows the recommendations for “Mercy.1” using metadata and the top five tags combined.

Table 16. Recommendations for “Mercy.1” by Kanye West using metadata and top five tags.

Artist	Track	Cosine Similarity
J. Cole	Looking For Trouble (feat. Kanye West, Big Sean, Pusha T & CyHi Da Prince)	0.6708
Pusha T	What Would Meek Do? (feat. Kanye West)	0.6694
2 Chainz	Birthday Song (feat. Kanye West)	0.6299
Logic	State Of Emergency (feat. 2 Chainz)	0.6285
Jay-Z	Who Gon Stop Me	0.6124
Logic	Wassup (feat. Big Sean)	0.6124
Pusha T	The Games We Play	0.6124
Drake	All Me (feat. 2 Chainz & Big Sean)	0.6086
Big Sean	All Your Fault (feat. Kanye West)	0.5948
Drake	Pop Style (feat. Jay-Z & Kanye West)	0.5833

Note. Same artist recommendation filter applied. Release year converted to decade value.

Breaking down the results for the metadata and top five tags combined, again, shows more diversity while maintaining stronger cosine similarity associations than tag based recommendations did alone. The first song recommended here, “Looking For Trouble (feat. Kanye West, Big Sean, Pusha T & CyHi Da Prince)” by J. Cole, looks to be recommended for its overlap in artists: Kanye West, Big Sean, and Pusha T all make an appearance in “Mercy.1” and “Looking For Trouble (feat. Kanye West, Big Sean, Pusha T & CyHi Da Prince)”. Pusha T’s “What Would Meek Do? (feat. Kanye West)” is recommended for the aforementioned reason that Kanye is both featured in this song and is the producer. Meanwhile, the remaining recommendations are all artist adjacent, following the idea that if a user likes a featured artist’s performance, they may like other songs with that artist.

Note that the cosine similarities are overall higher than Daft Punk’s in Table 13. This could potentially be explained by another factor; Out of the 3,440 songs in the dataset, roughly 1,600 are classified under the “hip-hop/rap” genre. With nearly fifty percent of the dataset sharing at least one frequency column with “Mercy.1”, there are more options for the system to choose from. The higher cosine similarities could also be because of the few tags associated with "Mercy.1".

6.6. Language Exclusion

Now we will look at how language affects recommendations. Language has been incorporated into all the previous results shown, but some people do not mind listening to songs in other languages or songs with no lyrics at all. "Rush" by Seatbelts is part of the 1998 show "Cowboy Bebop". Much of the show's soundtrack falls under blues, jazz, and their subgenres (particularly bebop). Table 17 shows the recommendations for "Rush" using metadata and the top five tags while including the language value.

Table 17. Recommendations for "Rush" by Seatbelts using metadata and top five tags with language data included.

Artist	Track	Cosine Similarity
Steve Conte	Rain	0.6934
Candy Dulfer	Lily Was Here	0.4961
Mai Yamane	Want It All Back	0.4615
John Cameron	Liquid Sunshine	0.4193
Limes	Blas	0.4193
Limes	Burnt Shake	0.4193
Limes	Heyo	0.4193
Limes	Hooplah	0.4193
Limes	Marigold	0.4193
Limes	Alright	0.3922

Note. Same artist recommendation filter applied. Release year converted to decade value.

The first song here is by a different artist but is still from the "Cowboy Bebop" soundtrack and has some overlap. Meanwhile, "Lily Was Here" by Candy Dulfer, "Liquid Sunshine" by John Cameron, and all the Limes' songs listed are jazz based. The song that does not fit the trend is Mai Yamane's "Want It All Back". This song is also from the "Cowboy Bebop" soundtrack, but it is on a separate released album. In "Want It All Back", Yamane sings in English. Overall, most songs were of the “Instrumental” language category with one being in the “English” category.

Now removing the language data can be done during the document creation process using the combine_features_top_tags_no_language function. Table 18 shows the recommendation results after excluding language.

Table 18. Recommendations for “Rush” by Seatbelts using metadata and top five tags with language data excluded.

Artist	Track	Cosine Similarity
Steve Conte	Rain	0.6708
Mai Yamane	Want It All Back	0.5000
Candy Dulfer	Lily Was Here	0.3727
Frank Sinatra	The Girl From Ipanema	0.3482
Michael Franks	St. Elmo’s Fire	0.3482
Cortex	Chanson d'un jour d'hiver	0.3333
Miles Davis	John McLaughlin	0.3203
Cortex	Huit octobre 1971	0.2981
John Cameron	Liquid Sunshine	0.2887
Limes	Blas	0.2887

Note. Same artist recommendation filter applied. Release year converted to decade value.

The first three songs remain consistent with the results in Table 17. Here we see that “Want It All Back” has moved up to the second highest recommendation since its conflicting language value with “Rush” is not taken into consideration and furthering its frequency vector. Meanwhile, the remaining recommendations have had a big change. Rather than focusing on Limes’ album “Freshy Squeezed”, other songs take their places. Here we see Frank Sinatra’s “The Girl From Ipanema” is recommended in fourth place despite the song being partially in English and partially in Portuguese, Michael Franks’ “St. Elmo’s Fire” is entirely in English, Cortex’s “Chanson d'un jour d'hiver” and “Huit octobre 1971” contain some French sections, and

Miles Davis and John Cameron’s songs are purely instrumental. All songs here have some basis in jazz and focus on the overlap between its genre set rather than getting stuck on the language.

Overall, while not considering language data provides interesting diversification to the recommendations, most people typically listen to songs in languages they understand.

6.7. Results of Zero Vectors

Next, we will look at what happens when the user-entered track generates a zero vector. As previously mentioned, the cosine similarity does not work if either vector in the formula is a zero vector. One downside of Last.FM is that the same song can have multiple pages depending on the way the title is structured. A simple search for “go legend” brings up five pages to the same song within the first ten query results. The pages are “Go Legend (& Metro Boomin)”, “Go Legend”, and “Go Legend (feat. Travis Scott)” by “Big Sean”; “Go Legend (feat. Travis Scott)” by “Big Sean & Metro Boomin”; and “Go Legend” by “Big Sean, Metro Boomin, Travis Scott”. The track title in the dataset is listed as “Go Legend (feat. Metro Boomin & Travis Scott)” by “Big Sean”. When using the Last.FM API, the autocorrect did not identify this track as any of the other pages and thus collected no tags.

With no tags associated with “Go Legend (feat. Metro Boomin & Travis Scott)” the document creation function for tags returns an empty string. Since there are no words in the document, when converted to the frequency matrix, the vector for Go Legend becomes a zero vector. Since an angle requires two lines to be created (as demonstrated in Figure 5), and one vector “line” has a length of zero, the angle cannot form. Regardless of the frequency vector for all other songs, the angle can never be created and so the cosine similarity is zero for all songs. Table 19 shows the recommendation results for “Go Legend (feat. Metro Boomin & Travis

Scott)” when using tags. Note that all recommended songs have a cosine similarity of zero in this case.

Table 19. Recommendations for “Go Legend (feat. Metro Boomin & Travis Scott)” by Big Sean using tags.

Artist	Track	Cosine Similarity
100 gecs	hand crushed by a mallet	0.0000
100 gecs	money machine	0.0000
100 gecs	gec 2 U (Remix) [feat. Dorian Electra]	0.0000
100 gecs	ringtone (Remix) [feat. Charli XCX, Rico Nasty, Kero Kero Bonito]	0.0000
100 gecs	stupid horse (Remix) [feat. GFOTY & Count Baldor]	0.0000
100 gecs	xXXi_wud_nvrstøp_ÜXXx (Remix) [feat. Tommy Cash & Hannah Diamond]	0.0000
2 Chainz	Feds Watching (feat. Pharrell Williams)	0.0000
2 Chainz	Netflix (feat. Fergie)	0.0000
2 Chainz	Birthday Song (feat. Kanye West)	0.0000
2 Chainz	I'm Different	0.0000

Note. Same artist recommendation filter applied. Release year converted to decade value.

All the recommendations listed are by 100 gecs and 2 Chainz. This is because, as discussed in Section 6.1, the recommendation model sorts songs based on the following criteria: first in descending order by cosine similarity, then alphabetically by artist, then alphabetically by album, and finally alphabetically by track title. In this case, since the cosine similarities of all songs are zero, the first artist listed is 100 gecs. Then “hand crushed by a mallet” and “money machine” are listed in alphabetical order because they are from the album “1000 gecs” which comes first alphabetically. Finally, “gec 2 U (Remix) [feat. Dorian Electra]”, “ringtone (Remix) [feat. Charli XCX, Rico Nasty, Kero Kero Bonito]”, “stupid horse (Remix) [feat. GFOTY & Count Baldor]”, and “xXXi_wud_nvrstøp_ÜXXx (Remix) [feat. Tommy Cash & Hannah Diamond]” are recommended next because the next album by 100 gecs in the dataset is titled “1000 gecs and the Tree of Clues”. Next is 2 Chainz using a similar idea for his albums “B.O.A.T.S. II: Me Time” and “Based on a T.R.U. Story”. In this case, all songs are equally recommended, and the listed songs are only listed because of the sorting process used by the system.

6.8. Comparison Summary

To summarize and compare the results, Table 20 shows a side-by-side comparison of the cosine similarities for “Get Lucky” by Daft Punk for each combined feature set. The songs are sorted by using the highest cosine similarity from the four options. Looking at this table, the general trend is that recommendations using metadata have the highest cosine similarity, followed by metadata with top five tags, metadata with all tags, then only tags.

Table 20. Side by side comparison of cosine similarities for “Get Lucky” by Daft Punk

Track		Cosine Similarity			
Artist	Title	Metadata	Metadata + Top Tags	Metadata + Tags	Tags
Icona Pop	I Love It (feat. Charli XCX)	0.6172	0.5809	0.3243	0.2723
Clean Bandit	Rather Be (feat. Jess Glynne)	0.6172	0.5809	-	-
Mike Posner	I Took a Pill In Ibiza [Seeb Remix]	0.6172	0.5669	-	-
Flight Facilities	Crave You (feat. Giselle Rosselli)	0.6172	-	-	-
M.I.A.	BORN FREE	0.6172	-	-	-
M.I.A.	TEQKILLA	0.6172	-	-	-
M.I.A.	aTENTion	0.6172	-	-	-
The Knocks	Love Me Like That (feat. Carly Rae Jepsen)	0.6172	-	-	-
Sophie Ellis-Bextor	Murder On the Dance Floor	-	0.5809	-	-
Disclosure	F For You	-	0.5809	-	-
Disclosure	Latch (feat. Sam Smith)	0.5714	-	-	-
M.I.A.	Bring The Noize	0.5714	-	-	-
Yerin Baek	“HOMESWEETHOME”	-	0.5669	-	-
Jessie Ware	Ooh La La	-	0.5625	0.3024	-
Jessie Ware	Adore You	-	0.5625	-	-
Jessie Ware	Read My Lips	-	0.5625	-	-
Jessie Ware	Spotlight	-	0.5265	-	-
Justice	D.A.N.C.E.	-	-	0.3577	0.3146
Moloko	Sing It Back	-	-	0.3462	0.3066
Deee-Lite	Groove Is in the Heart	-	-	0.3436	0.3093
Gorillaz	DARE (feat. Shaun Ryder of Black Grape)	-	-	0.3402	0.3219
Dev	In The Dark	-	-	0.3258	0.2769
Rina Sawayama	Comme des Garçons (Like the Boys)	-	-	0.3125	-
Moloko	Cannot Contain This	-	-	0.3111	-
Jennifer Lopez	On The Floor (feat. Pitbull)	-	-	0.3014	0.2591
Nicki Minaj	Starships	-	-	-	0.2813
Michael Jackson	Billie Jean	-	-	-	0.2644
Sublime	Doin’ Time	-	-	-	0.2577

Out of all the songs listed, there is only one song recommended across all combined feature sets: “I Love It (feat. Charli XCX)”. Looking at the “Tags” and “Metadata + Tags” columns shows that, originally, “I Love It (feat. Charli XCX)” would have been further down the recommendation list. This is because “I Love It (feat. Charli XCX)” is a popular song, so it contains more tags than most other songs. Because of the increase in tags, the dimensions of the frequency matrix increase and lower the cosine similarity values. To balance the dimensional increase, using the metadata with top five tags is the best approach to reduce the effects of having many tags.

Despite the combination of metadata with the top tags providing a good balance of both metadata and tags, each combined feature set provides some form of insight. Table 21 breaks down the advantages and disadvantages of each combined feature set. While most of the recommendations were evaluated through domain knowledge, consistent observations were evaluated and noted.

Out of all the combined feature sets, the only time an application-breaking scenario occurred was when recommendations were given based on tags in which the user entered a track containing no tags.

Table 21. Advantages and Disadvantages of each combined feature set

Combined Feature Set	Advantages	Disadvantages
Metadata	• No zero vectors • Structured data	• Limited information on characteristics
Tags	• Characteristic insight • Additional information can be added	• Zero vectors are possible • Unstructured data • Arbitrary tags reduce cosine similarity • Number of values vary significantly
Metadata + Tags	• No zero vectors • Semi-structured data • Offers both aspects of metadata and tags • Defaults to metadata if there are no tags	• Increase in the number of values in the frequency matrix decreases the cosine similarity • Number of values vary significantly
Metadata + Top 5 Tags	• No zero vectors • Semi-structured data • Offers both aspects of metadata and tags • Defaults to metadata if there are no tags	• Increase in the number of values in the frequency matrix decreases the cosine similarity (to a lesser extent than Metadata + Tags) • Top tags can reinforce genres if the values are the same

Starting with metadata, while having a defined structure helps keep data organized and guarantees certain aspects of songs are included, it comes at the cost of missing the emotional aspects of a song that is provided by tags. Metadata does not consider tones, themes, stories, and other characteristics of a song. The cosine similarities are high when using metadata, but the lack of proper characteristic portrayal limits its recommendation diversity.

With tag data being unstructured, they can store a variety of information including metadata, record labels, critics' ratings, emotions, themes, etc. Aside from the zero-vector issue, there are two other issues when using tags. The first issue is that arbitrary tags increase the dimensions of the frequency matrix, lowering the cosine similarity. The second issue is that some tracks have more tags than others. Popular tracks will have more tags because more users will visit the track's Last.FM page and add tags. This makes it so that popular tracks have a higher chance of getting recommended simply because there are more tags to work with.

While combining metadata with tags brings the benefits of both together, it furthers the issues that using tags face. Having values for both metadata and tags further increases the dimensions of the frequency matrix and decreases the cosine similarity. Also, because popular tracks have more tags, there is a higher chance that they will be recommended.

Finally, by using metadata and limiting the number tags to five, some of the issues from the previous methods are addressed. Worst case, when there are no tags, this combined feature set recommends the same songs as if only using metadata. Meanwhile, limiting the number of tags to only the top five most popular tags lessens the reach that popular songs with many tags have. The downside to this method is when the tag values are the same as the metadata values; This scenario strengthens the recommendation reliance on metadata without adding any of the benefits offered by using tags.

7. FURTHER RESEARCH

7.1. Writer Data

Looking into further research, one step is to include writers in the credited collaborators data collection process. While people might like certain artists or certain producers, people may also like certain writers and their writing style. Originally, information on song writers was collected, but after looking at the data, the information was found to be inaccurate. After researching a few other sources, there were no consistent and accurate sources available. Figure 8 shows the original document creation function for combine_features_meta to include metadata with writers. Note that the line to add the list of writers was added similarly to how featured artists and producers were added.

```
def combine_features_meta(row):
    # returns string of combined features
    combined_features = ""
    if(len(str(row['artist'])) > 0):
        combined_features = combined_features + row['artist'].lower() + " "
    if(len(str(row['genre'])) > 0):
        combined_features = combined_features + row['genre'].lower() + " "
    if(len(str(row['featured_artists'])) > 0):
        combined_features = combined_features + \
            row['featured_artists'].lower() + " "
    if(len(str(row['producers'])) > 0):
        combined_features = combined_features + row['producers'].lower() + " "
    if(len(str(row['writers'])) > 0):
        combined_features = combined_features + row['writers'].lower() + " "
    if(len(str(row['language'])) > 0):
        combined_features = combined_features + row['language'].lower() + " "
    if(len(str(row['year'])) > 0):
        combined_features = combined_features + \
            str(math.trunc(row['year']/10)) + " "
    return combined_features
```

Figure 8. Code snippet of original combine_features_meta function.
Note. The code used to create the document of a track based on metadata including writers.

7.2. Theme Data

Another idea is to add themes from lyrics to the document and compare those results against metadata, user generated tags, and all three combined. In order to extract the themes of lyrics, the lyrics of all songs would have to be downloaded and sifted through. Using stop words could help extract words related to the theme of a song, but most likely those extracted words would need to be filtered further. Once the key words are extracted, they could be included in the same document creation function and cosine similarity calculations used in this experiment.

7.3. Evaluating Other Datasets

Another option is to apply this process to different and larger datasets. As discussed before, without extensive prior knowledge of the dataset, using a content-based filtering approach can be difficult. While I used a dataset that I had a full understanding of in order to assess the validity of the recommendations, applying other datasets and getting different perspectives can provide great insight on how to further strengthen the recommendation process. Regardless, having a greater number or more diverse track list could help alleviate any hidden biases in the dataset too. As previously mentioned, nearly fifty percent of songs in the dataset were listed under the “hip-hop/rap” genre. This may have affected the results such that songs that are classified under “hip-hop/rap” may have gotten stronger recommendations than songs in different genres simply because there are more songs for the system to work with. However, besides needing a greater understanding of the dataset, having a bigger and more diverse dataset also comes at the cost of space and processing time. Looking at the Million Song Dataset again, that dataset amasses over 280 gigabytes of space and holds service-specific audio characteristics and metadata information that would not have been relevant to this experiment. Examples include the track’s musicbrainz.org ID value, loudness data, start of fade out data, etc.

7.4. Surveys and Other Recommendation Methods

While on the topic of different comparison methods, surveying people and potentially having them rank recommendations provided by the system is another method of comparison. Because music can be an emotional subject and different people perceive the same song differently, surveys could give insight to the recommendation accuracy or help develop another recommendation process.

And while on the topic of processes, looking at other recommendation methods and comparing the effectiveness (either by performing a survey or by other means) is something to investigate.

7.5. Reevaluating Tag Ranks

Finally, Last.FM does not explicitly explain in their API documentation what system is used to rank user generated tags. Despite having a “count” value associated with each tag returned in the JSON response results, the value does not seem to directly reflect the number of users that agree/disagree with the tag. Having some form of robust voting system could give more information to the recommender system, potentially allowing for the weighting or filtering of tags. The issue with this approach is that creating this system would require time to develop and can suffer from the same downsides found in typical collaborative filtering methods.

8. CONCLUSION

After evaluating the combined feature sets of metadata, user generated tags, metadata with all tags, and metadata with top five tags, using metadata with the top five tags balanced providing a strong cosine similarity while also filtering, maintaining, and representing the diversity created by including user generated tags. Meanwhile, solely using metadata provided strong cosine similarities, but lacked diversity in the results. And while using user generated tags provided an increase in diversity, it came at the expense of weaker cosine similarities, managing irrelevant tags, and offered recommendations that did not contain obvious associations.

REFERENCES

[1] Devsnd. “Devsnd/tinytag: Read audio and music meta data and duration of MP3, ogg, opus, MP4, m4a, FLAC, WMA, wave and AIFF files with python 2 or 3,”. Github. https://github.com/devsnd/tinytag.

[2] C. Hu, “What is ‘Escape room’ and why is it one of my top genres on Spotify?,” *Medium*, 16-Dec-2016. https://festivalpeak.com/what-is-escape-room-and-why-is-it-one-of-my-top-genres-on-spotify-a886372f003f.

[3] F. Pedregosa et al., “Scikit-Learn: Machine Learning in Python,” J. Mach. Learn. Res., vol. 12, pp. 2825–2830, Nov. 2011.

[4] J. D. Hunter, “Matplotlib: A 2D graphics environment,” Computing in Science & Engineering, vol. 9, no. 3, pp. 90–95, 2007.

[5] J. Karlgren, “An algebra for recommendations: Using reader data as a basis for measuring document proximity.” Department of Computer and Systems Sciences, Stockholm University, 1990.

[6] L. Buitinck et al., “API design for machine learning software: experiences from the scikit-learn project,” 2013. https://arxiv.org/abs/1309.0238.

[7] Last.FM. “API Terms of Service.” Last.FM. https://www.last.fm/api/tos. (accessed March 04, 2022).

[8] Spotify. “Decades.” Spotify. https://open.spotify.com/genre/decades-page.

[9] T. Bertin-Mahieux, D. P. W. Ellis, B. Whitman, and P. Lamere, “The million song dataset,” 2011. http://millionsongdataset.com/.