INVESTIGATING BIAS IN MUSIC RECOMMENDER SYSTEMS

DANIEL GOMEZ GONZALEZ

Thesis supervisor: PETER KNEES (TU Wien)

Tutor: GERARD ION GALLEGO OLSINA (Department of Signal Theory and Communications)

Degree: Bachelor's Degree in Data Science and Engineering

Thesis report

Facultat d'Informàtica de Barcelona (FIB)
Escola Tècnica Superior d'Enginyeria de Telecomunicació de Barcelona (ETSETB)
Facultat de Matemàtiques i Estadística (FME)

Universitat Politècnica de Catalunya (UPC) - BarcelonaTech

Abstract

Music Recommender Systems (MRS) are software applications that provide personalized music recommendations based on user preferences and listening history. They analyze data to suggest music that aligns with individual tastes, enhancing the music discovery experience.

This thesis aims to investigate the influence of record labels across different music recommendation datasets and evaluate their impact on recommender systems. Additionally, it seeks to expand the scope and experimentation of prior research on bias within feedback loops of MRS.

To study their effect, the datasets are preprocessed and fed into a multi-stage web crawler that retrieves record label information for individual albums as well as an assignment to a major record company (Universal, Sony, Warner) or independent. This crawler is used to enrich our dataset collection. Based on the additional information, we can show different characteristics and identify particular biases in their user-generated music collections of playlists and listening profiles.

Moreover, recommender system experiments are conducted, presenting results of feedback loop simulations, where the stability of record label distribution in longitudinal recommendations are studied. All findings and gathered record label information are made publicly available to the research community.

Resum

Els Sistemes de Recomanació Musical (MRS) són aplicacions de software que proporcionen recomanacions de música personalitzades basades en les preferències i el històric d'escolta de l'usuari. Analitzen dades per suggerir música que s'ajusti als gustos individuals, millorant així l'experiència de descobriment musical.

Aquesta tesi té com a objectiu investigar la influència de les discogràfiques en diferents conjunts de dades de recomanació musical i avaluar el seu impacte en els sistemes de recomanació. A més, busca ampliar l'abast i l'experimentació de recerques prèvies sobre biaixos en els bucles de retroalimentació dels MRS.

Per estudiar el seu efecte, els conjunts de dades es pre-processen i s'insereixen a un rastrejador web de diverses etapes que recopila informació sobre les discogràfiques dels àlbums individuals, així com la seva classificació en una discogràfica principal (Universal, Sony, Warner) o independent. Aquest rastrejador s'utilitza per enriquir la nostra col·lecció de dades. Basant-nos en la informació addicional, podem mostrar diferents característiques i identificar biaixos particulars en les col·leccions de música generades pels usuaris, com ara llistes de reproducció i perfils d'escolta.

A més, es fan experiments en un entorn simulat de recomanacions, presentant els primers resultats de la simulació de bucles de retroalimentació on s'estudia l'estabilitat de la distribució de segells discogràfics en recomanacions longitudinals. Totes les troballes i la informació recopilada de segells discogràfics es posa a la disposició del públic per a la comunitat investigadora.

Resumen

Los Sistemas de Recomendación Musical (MRS) son aplicaciones de software que proporcionan recomendaciones de música personalizadas basadas en las preferencias y el historial de escucha del usuario. Analizan datos para sugerir música que se ajuste a los gustos individuales, mejorando así la experiencia de descubrimiento musical.

Esta tesis tiene como objetivo investigar la influencia de las discográficas en diferentes conjuntos de datos de recomendación musical y evaluar su impacto en los sistemas de recomendación. Además, busca ampliar el alcance y la experimentación de investigaciones previas sobre sesgos en los bucles de retroalimentación de los MRS.

Para estudiar su efecto, los conjuntos de datos se preprocesan y se alimentan a un rastreador web de varias etapas que recopila información sobre las discográficas de álbumes individuales, así como su clasificación en una discográfica principal (Universal, Sony, Warner) o independiente. Este rastreador se utiliza para enriquecer nuestra colección de datos. En base a la información adicional, podemos mostrar diferentes características e identificar sesgos particulares en las colecciones de música generadas por los usuarios, como listas de reproducción y perfiles de escucha.

Además, se realizan experimentos con el sistema de recomendación, presentando los primeros resultados de las simulaciones del bucle de retroalimentación, donde se estudia la estabilidad de la distribución de las discográficas en las recomendaciones longitudinales. Todos los hallazgos y la información recopilada sobre las discográficas se ponen a disposición de la comunidad investigadora.

Acknowledgements

I am grateful to express my sincere gratitude to my research supervisor, Dr. Peter Knees. Without his support and guidance throughout every step of this project, it would not have been possible to complete it successfully. I appreciate the opportunity to learn valuable skills during this project that I can apply in my future endeavors.

Contents

1. Introduction

1.1 Motivation

In today's information age, the amount of data available to individuals and organizations has grown exponentially. With the rapid increase in data, there is a growing need to efficiently and effectively process and analyze it to derive meaningful insights. Recommender systems have emerged as a powerful tool for making sense of big data by providing personalized recommendations based on users' behavior and preferences.

Recommender systems heavily depend on data to deliver personalized recommendations that align with user behavior and preferences. Consequently, the quantity and quality of data used for training these systems play a vital role in their performance. In order to enhance data quality, researchers are exploring automated techniques for information retrieval to augment existing datasets with additional information sourced from publicly available platforms. The initial part of this thesis focuses on gathering data from public sources and APIs, particularly collecting information from record labels regarding music tracks. This information is then utilized to augment music recommender datasets.

Music recommender systems have become instrumental in helping users navigate the vast amount of available items. While some of these systems, such as those based on collaborative filtering, can introduce users to new genres or bands, they often recommend items that are already popular and widely accepted. It has been shown [11] that this approach can result in feedback loops, leading to increasingly similar recommendations. This bias towards popularity can have a negative impact on the representation of specific groups in the recommendations. Additionally, recent research has highlighted the importance of fairness in recommendations, not only in terms of performance but also in terms of the interests of both shareholders and users [23]. Countermeasures can be implemented to augment the diversity of recommendations, but it is necessary to identify the existing biases beforehand.

The second part of this thesis is to explore whether the record label belonging a particular music piece influences the personalized recommendations that are presented to users. To conduct the investigation, datasets from music streaming services will be utilized, as these services are a major force in the contemporary music industry [1]. These datasets will be augmented with record label information gathered through the internet.

Previous work [26, 20] presented analyses of record label diversity on two datasets, the Spotify Million Playlist Dataset[1] and the LFM-2b[2] using Last.fm listening profiles. These datasets were augmented using a multi-stage web crawler that retrieves record label information. Also identified particular biases and showed the results of first experiments with regard to feedback loop simulation and the stability of record label distribution in the recommendation process.

[1]https://www.aicrowd.com/challenges/spotify-million-playlist-dataset-challenge
[2]http://www.cp.jku.at/datasets/LFM-2b

1.2 Aim of this Thesis

This thesis aims to further explore and expand upon the existing knowledge in biases present in music recommender datasets as well as in music recommender systems, particularly within the realm of record label companies and feedback loop recommendations. Through the integration of new datasets and alternative recommendation setups, this thesis strives to provide a deeper understanding into the subject matter. This section provides a detailed explanation of the advancements that this work aims to accomplish.

1.2.1 Preprocessing enhancement for the data augmentation step

Up until this point, the previous work has concentrated on enriching solely two datasets utilizing the crawler. Before the record label for each entry of these datasets could be obtained, they had to undergo preprocessing to ensure the suitable format for the crawler. These preprocessing steps were conducted separately for each of the datasets, therefore these steps would not be applicable if the crawler were to be executed on a new dataset.

One of the aims of this thesis is to develop a highly effective preprocessing method that can be utilized to incorporate any type of musical dataset (with minimal requirements) as an input for the crawler, this way, record labels could be obtained for any dataset. By achieving this objective, an analysis of the impact of record labels in the recommendation process can be conducted on a wider range of datasets, leading to more accurate conclusions.

As part of the high-value preprocessing to be developed, a previous step will be included to address situations where the dataset to be introduced in the crawler lacks the necessary fields to perform the data gathering. This step involves collecting the necessary fields for each entry of the dataset through the Spotify API. This will ensure that the crawler can be properly executed and record label information can be obtained for each entry of the dataset .

1.2.2 Impact of record labels in recommender systems

The datasets enriched with record label information obtained from the label-crawler will be utilized in systematic machine learning experiments. The aim is to investigate the impact of record label information on the performance of recommender systems and to simulate more complete feedback loops using diverse recommendation algorithms based on collaborative filtering as well as different methods to simulate user feedback across a broader range of music datasets. These algorithms will be ALS, BPR and LMF, explained in section 2.

The evaluation of feedback loops in the recommender systems will focus on evaluating the quality of recommendations with emphasis on the diversity of record labels. To measure the reproduction of biases and stability of record label distribution in the recommendation process, we will utilize the *Reach and Exposure* metrics proposed in [12].

1.2.3 Research questions and hypotheses

The aforementioned enhancements will provide more comprehensive responses to two research questions that had been raised in previous work as well as allow for the exploration and resolution of an additional research question.

RQ1 What biases related to record labels can be identified in music datasets? The focus is on identifying these potential biases.

RQ2 How do these biases affect the recommendations provided by a recommender system? The focus is on exploring the impact of biases related to record labels on the recommendations generated by a recommender system.

RQ3 To what extent do bias effects persist and generalize across a broader range of research datasets? The focus is on exploring the consistency and generalizability of bias effects in music recommendations across a broader range of datasets, with the ultimate goal of expanding the knowledge and understanding in this area of research.

Several hypotheses have been put forward to address these questions, which will serve as guiding principles for the thesis.

H1 Once record label information is collected and incorporated into music datasets, biases related to certain record labels will become evident. These biases may manifest in the form of unequal representation of music from different labels.

H2 As feedback loops are iterated, biases associated with record labels significantly contribute to a decrease in the diversity of recommendations. The recommendation process could result in a narrower emphasis on specific record labels, potentially restricting the variety and exposure of music recommendations.

H3 The bias effects observed will persist and generalize across a broader range of research datasets. Since these datasets are sampled from real-world listening scenarios, it is expected that they will exhibit similar bias effects. The effects observed in the previous study are not unique to the specific datasets used but rather representative of broader phenomena.

In summary, the two primary objectives pursued by this thesis are as follows:

1. Improving the preprocessing performed to the datasets prior to entering the web-crawler. The goal of this improvement is to develop a code that can adapt to any dataset, providing flexibility when selecting a new dataset to work with, and eliminating the need to create a specific preprocessing code for each new dataset. This will simplify the process and streamline the workflow.

2. Conducting more comprehensive feedback loop simulations using recommender systems that incorporate a wider range of datasets by leveraging the flexible preprocessing method developed previously. These simulations will encompass different recommendation algorithms and employ various techniques to simulate user feedback, resulting in a more comprehensive analysis of the feedback loop dynamics.

The enriched datasets and documented recommender system experiments resulting from this project will serve as technical contributions that will facilitate future research.

1.3 Structure of the Thesis

Following a brief introduction of the project's purpose and key objectives, it is provided a concise overview of its structure.

1. **Introduction:** This section explains in depth the motivation and main objectives of the project.
2. **Theoretical Background:** This section provides a summary of the existing theories and concepts relevant to the research topic. It establishes the theoretical foundation for the study and demonstrates the researcher's understanding of the subject.
3. **Related work:** The related work section provides a comprehensive review of relevant literature, including previous studies and concepts related to bias in music recommender systems. It shows how the thesis builds upon existing knowledge.
4. **Dataset Research:** This section of the thesis explains the process of searching useful datasets that match the requirements of our project, including the criteria for identifying relevant datasets and the sources of datasets.
5. **Methodology, Data Gathering:** This section describes the research design and techniques employed to address the first objective of the thesis. It includes information on the data collection used in the research.
6. **Methodology, MRS Experiments:** This section describes the research design and techniques employed to address the second objective of the thesis. It includes information on the machine learning experiments used in the research.
7. **Dataset Analysis:** In this section, the findings of the data gathering process are presented and analyzed. It includes tables, figures and charts to illustrate the data and statistical analyses.
8. **Feedback Loop Simulation:** In this section, the findings of the machine learning experiments and feedback loop simulations are presented and analyzed. It includes tables, figures and charts to illustrate the data and statistical analyses.
9. **Discussion:** The discussion section interprets the results, relates them to the research questions, and discusses their implications. It explores the significance of the findings and compares them with previous studies.
10. **Conclusion and Future Work:** This section contains the main conclusions extracted after the finalization of the project, as well as a list of future developments that could help to extend this research.

11. **References:** The references section lists all the sources cited in the thesis using the appropriate citation style. It includes books, journal articles, conference papers, websites, or any other references used in the study.

12. **Appendix:** The appendix section of this thesis includes two main components: a comprehensive description of the content of each dataset utilized in the study, and a detailed outline of the work plan and methodology implemented throughout the research process.

2. Theoretical Background

2.1 Music Information Retrieval

Music Information Retrieval (MIR) is a multidisciplinary field that focuses on developing techniques and algorithms to extract, analyze, and organize music-related data. It involves leveraging computational methods to retrieve, process, and understand various aspects of music, such as audio signals, musical notation, lyrics, metadata, and user-generated content.

One crucial aspect of MIR is the gathering of information from publicly available sources and Application Programming Interfaces (APIs). A variety of platforms fall under the umbrella of publicly available sources, including music streaming services like Spotify, online music databases, social media platforms such as Twitter, and websites dedicated to music-related content. These sources often contain valuable data about music tracks, albums, artists, genres, and other relevant information.

APIs, on the other hand, provide a structured interface to access and retrieve data from specific platforms or services. Music-related APIs offered by streaming platforms or music databases allow researchers and developers to programmatically access metadata, audio features, user preferences, and other valuable information.

The process of gathering information across these sources typically involves automated techniques, such as web scraping, data mining, and API queries. Researchers design and implement algorithms to collect data from websites, extract relevant information, clean and preprocess it, and integrate it into their research or application pipeline. By leveraging these sources and APIs, researchers can augment existing datasets, enrich them with additional information, and explore new perspectives and insights.

Overall, the gathering of information across publicly available sources and APIs in MIR plays a vital role in enabling researchers to access comprehensive and diverse datasets, enhancing the quality and depth of their studies, and ultimately advancing the field's understanding of music.

2.1.1 Spotify URI concept

The Spotify URI[3] term refers to Spotify's unique resource identifiers (URIs). An URI is a unique identifier that represents a specific resource within the Spotify music streaming platform. It is a string of characters that serves as a direct link to a particular song, album, artist, playlist, or other elements available on Spotify.

The Spotify URI is composed of the prefix "spotify:" followed by a specific identifier for the resource. For example, an album URI would have the format "spotify:album:{album_id}", where

[3]https://community.spotify.com/t5/FAQs/What-s-a-Spotify-URI

"{album_id}" represents the unique identifier for that specific album. The category is selected from a predefined set (such as "track:", "album:", or "artist:") and the ID is a base-62 number with a length of 22.

An example of final Spotify URI could be "spotify:album:5IB8F5PXr0OPf6ejak165-U", this is used for making requests to the Spotify API and is also reflected in the Spotify URL (`https://open.spotify.com/album/5IB8F5PXr0OPf6ejak165U`) which opens the track, album, or artist in a Spotify client.

Using Spotify URIs, users can easily share and access specific content within the Spotify ecosystem. By clicking or tapping on a Spotify URI, it can directly open the corresponding resource within the Spotify app or web player, providing a seamless way to navigate and listen to music.

2.2 Music Recommender Systems

A recommender system is a technology that suggests personalized recommendations to users based on their preferences and behaviors. It helps users discover relevant items from a large selection of options, using techniques like collaborative filtering and machine learning algorithms. This systems applied to the music field suggest music to users based on their preferences, listening habits, and other relevant factors. The goal is to help users discover new music, enhance their music listening experience, and provide tailored recommendations that align with their individual tastes. Music recommender systems aim to promote exploration and satisfaction for users.

In the field of music recommender systems, exploration refers to the discovery of new and unfamiliar music, while satisfaction relates to the contentment users feel when the system accurately recommends music that aligns with their preferences. Balancing these concepts is crucial to provide personalized recommendations while also encouraging users to explore new musical content.

This is achieved by collecting data on various aspects, such as explicit feedback (user ratings or reviews) and implicit feedback (listening history, skip patterns, or duration of play). Additionally, contextual information, such as time of day or location, can also be taken into account.

The impact of MRS extends beyond individual users. They also play a crucial role in the music industry by aiding music discovery, promoting new artists, and influencing music consumption patterns. Record labels, streaming platforms, and other industry stakeholders rely on MRS to engage listeners and drive revenue.

2.2.1 Understanding Bias in MRS

Bias in the context of music recommender systems refers to the presence of systematic and unfair preferences or discrimination towards certain music genres, artists, or user groups. Bias can occur at different stages of the recommendation process, including data collection, algorithm design, and user feedback analysis.

Data bias can arise when the training data used to build the recommender system is skewed or unrepresentative. For example, if the training data primarily consists of popular mainstream music, the system may tend to favor those genres and overlook niche or less popular genres. Similarly, if the training data predominantly represents the preferences of a particular demographic group, it can lead to biased recommendations that may not cater to the preferences of other user groups.

Popularity bias is a common issue that can arise in recommendation algorithms or machine learning systems. It occurs when the algorithm prioritizes popular or commonly selected options over less well-known or less frequently chosen ones. This bias can have significant consequences, such as reinforcing existing power structures and harmful stereotypes as well as, limiting opportunities for marginalized groups.

Algorithmic bias can occur when the recommendation algorithms themselves are designed in a way that systematically favors or discriminates against certain music genres or artists. For instance, an algorithm that heavily relies on collaborative filtering may reinforce existing popular trends, leading to a limited diversity of recommendations. This can perpetuate existing inequalities and hinder the visibility of emerging or underrepresented artists.

User feedback bias can emerge when the recommender system relies on user feedback signals, such as ratings or likes, which may be influenced by subjective biases or external factors. If users' feedback is skewed towards certain music genres or artists due to social or cultural biases, the system may reinforce these biases in its recommendations, further limiting exposure to diverse musical content.

Addressing bias in music recommender systems is crucial to ensure fair and inclusive music recommendations. By considering diverse user preferences, providing transparency in algorithmic decisions, and regularly evaluating and refining recommendation systems, it is possible to mitigate the impact of bias and enhance the overall quality and fairness of music recommendations.

Simpson Index

The Simpson index (SI), also known as Simpson's diversity index or Simpson's concentration index, is a statistical measure used to assess the diversity or concentration of a dataset. It is commonly applied in ecology to measure the diversity of species in a specific area, but it can also be adapted to analyze other types of datasets, such as music datasets.

The Simpson index takes into account both the number of different categories present in the dataset and the relative abundance or frequency of each category. It ranges from 0 to 1, where 0 represents maximum diversity (all categories are equally represented), and 1 represents maximum concentration (only one category dominates the dataset).

In order to calculate the Simpson index for music datasets and identify potential biases related to record labels, it is essential to have access to a dataset that includes information on the record labels associated with each track. In addition, certain track listening profiles such as playlists created by users or user listening histories (less conscious decisions) must be included.

Then the Simpson index, metric that measures the probability that two tracks from a subset belong to the same major record label, could be computed for each of the user profiles of the dataset, either playlists or listening events. To calculate the Simpson index using the following formula:

$$\lambda(U) = 1 - \sum_{i=1}^{N} p_i^2$$

where U is one user from the dataset, N is the number of major record labels from the dataset (4) and p_i is the probability for a class i that a randomly drawn track belongs to this class.

2.2.2 Collaborative Filtering in MRS

Music Recommender Systems employ sophisticated algorithms to process and analyze the collected data. These algorithms leverage techniques like collaborative filtering, content-based filtering, and hybrid approaches. The two main approaches are:

– **Collaborative Filtering:** This approach relies on similarities between users' listening behaviors and preferences. It identifies users with similar music tastes and recommends songs or artists based on what those similar users have liked or listened to.

– **Content-Based Filtering:** This approach focuses on the characteristics and attributes of songs or artists. It analyzes the musical features, genres, lyrics, or other metadata associated with the music and recommends similar items based on those attributes.

Content-based filtering is useful when explicit item characteristics are available, but it may struggle to capture complex user preferences. Collaborative filtering, on the other hand, can provide more personalized recommendations but requires a sufficient amount of user data to identify meaningful similarities.

Due to the lack of comprehensive song characteristic information in most music datasets, user-based collaborative filtering offers a promising solution, as it focuses on capturing similarities between users and their preferences, rather than relying on explicit song characteristic information. User-based collaborative filtering operates on the fundamental assumption that users with similar musical preferences in the past will exhibit similar preferences in the future. By analyzing user behaviors, such as their listening history or playlists, the system can identify patterns and connections between users, allowing for effective recommendation generation.

User-Item Matrix

The user-item matrix in collaborative filtering is a fundamental data structure that represents the interactions of users with different items. It is a matrix where rows correspond to users, columns correspond to items, and the values within the matrix represent user-item interactions. It captures user behavior, such as ratings, reviews, clicks, or purchases, and serves as the foundation for generating personalized recommendations. The values in the matrix can be explicit, such as explicit ratings provided by users, or implicit, such as implicit feedback derived from user actions like item views or duration of play.

This matrix is typically sparse because not all users have interacted with all items. As a result, the challenge lies in filling in the missing values to estimate the preferences of users for items they have not yet interacted with. Recommendation algorithms leverage this matrix to learn patterns and make predictions about user preferences, allowing them to suggest relevant items to users.

Various recommendation techniques utilize the user-item matrix, including collaborative filtering, content-based filtering, and hybrid approaches.

Collaborative Filtering Algorithms

Collaborative filtering is a common approach in recommendation systems that analyzes user-item interactions to make recommendations. ALS, BPR, and LMF are widely used collaborative filtering techniques that have been proven effective in various recommendation scenarios.

- **ALS (Alternating Least Squares):** Is a collaborative filtering algorithm commonly used for recommendation systems. It operates by factorizing the user-item matrix into two lower-rank matrices: one representing users' preferences and the other representing items' attributes. ALS iteratively alternates between optimizing these matrices to minimize the difference between the predicted ratings and the actual ratings. It is known for its ability to handle large-scale datasets and has been widely adopted in recommender systems.
- **BPR (Bayesian Personalized Ranking):** Is a matrix factorization-based recommendation algorithm that focuses on pairwise ranking of items. It aims to learn latent factors of items and users' preferences by maximizing the likelihood of the observed item rankings. BPR assumes that a user prefers one item over another if they have interacted with the former item. It optimizes the model parameters using stochastic gradient descent, allowing for efficient computation in large-scale datasets. BPR is particularly suitable for scenarios where explicit feedback is not available.
- **LMF (Logistic Matrix Factorization):** Is a recommendation algorithm that combines matrix factorization with logistic regression. It models user-item interactions as a binary classification problem and predicts the probability of user-item interactions based on latent factors. LMF is suitable for handling implicit feedback and generating personalized recommendations.

2.2.3 Feedback Loops in MRS

Feedback loops in music recommender systems refer to the cyclical process where user feedback influences the recommendations generated by the system, which, in turn, affects user behavior and further feedback. This iterative loop plays a crucial role in shaping the personalized music recommendations provided to users.

The feedback loop begins with the initial recommendations presented to the user based on various techniques such as collaborative filtering. As users interact with the recommended music, they provide implicit or explicit feedback through actions like listening, skipping, liking, or disliking songs. This feedback is then captured and utilized by the recommender system to update and refine subsequent recommendations.

There are two main types of feedback loops in music recommender systems: explicit and implicit feedback loops.

- **Explicit Feedback Loop:** In this loop, users provide explicit feedback explicitly expressing their preferences, such as rating songs, creating playlists, or providing direct feedback on recommended items. This type of feedback is valuable as it directly reflects user preferences and can be used to enhance the accuracy of recommendations.

- **Implicit Feedback Loop:** Implicit feedback is derived from users' actions and behavior without them explicitly expressing their preferences. Actions like play counts, skipping, duration of listening, or the number of times a song is added to a playlist are examples of implicit feedback. This type of feedback is automatically collected by the system, making it more scalable but also more challenging to interpret accurately.

User feedback

The user feedback term refers to the information provided by users about their preferences and interactions with recommended items. It helps personalize recommendations and improve their accuracy and relevance.

In recommendation setups where direct user feedback is unavailable, it becomes necessary to simulate such feedback. User feedback simulation involves generating or imitating user feedback in recommendation systems, primarily for research or testing purposes. By simulating user feedback, researchers and developers can gain insights into algorithm behavior and make informed decisions regarding system improvements.

We categorize the methods for simulating this feedback in the recommendation process into two types.

- **Top-N Selection:** This method involves selecting the N highest-ranked recommendations generated by the recommendation algorithm for each user.

– **Random Selection:** In this method, a subset of predictions generated by the recommendation algorithm is randomly sampled. From this subset, N random recommendations are selected for each user.

3. Related work

The field of Music Information Retrieval and Music Recommender Systems (see section 2) has seen significant research and exploration in recent years. Previous studies and theses have delved into various aspects of this subject, providing valuable insights and establishing a foundation for further investigation. Building upon the rich body of existing literature, this section aims to present a comprehensive overview of the related work in the field and identify the gaps and opportunities that this thesis seeks to address to extend and complement the content explored in previous research.

3.1 Data Gathering for Music Datasets

In the field of music datasets, the process of data gathering holds immense significance as it forms the foundation for conducting meaningful research and analysis (section 2.1). Within this context, the collection of record label information emerges as a crucial aspect, providing valuable insights into the music industry's dynamics and artist-label relationships.

The related work in data gathering for record label information in music datasets encompasses a comprehensive examination of the methods and techniques employed by previous researchers. There are various methods of data gathering for music datasets. A state-of-the-art survey paper offers various approaches involving machine learning, information retrieval, and signal processing for music data analysis [16]. These approaches have demonstrated their immense value in the generation of comprehensive and current record label datasets, facilitating extensive analysis and investigation of trends and dynamics within the music industry.

3.1.1 Multi-stage Crawler

In previous research, a multi-stage web crawler was developed to gather record label data for music datasets [20]. However, this existing crawler has certain limitations, particularly in terms of its compatibility with different music datasets. The major gap in this crawler lies in its inability to adapt and gather information from record labels for any given dataset. In other words, the current crawler implementation is constrained to operate exclusively with the MPD and LFM-2b datasets, as it relies on Spotify Album URIs for input. The definition of Spotify URI can be found in section (2.1.1). However, it is worth noting that only a small number of music datasets actually provide this specific information.

In light of these limitations, this thesis proposes an enhanced crawler capable of executing and collecting record label information from a wide range of datasets. The proposed extension seeks to address the incompatibility issue by developing a more flexible and adaptable crawler framework. This advancement will greatly enhance the crawler's utility and provide researchers with a valuable tool for obtaining record label information across a wide range of music datasets. Researchers will be empowered to explore and analyze record label data from different music datasets, opening new avenues for understanding the dynamics of the music industry and facilitating more comprehensive studies in this domain.

Section 5 of this thesis presents the methodology employed to enhance the multi-stage web crawler and outlines the step-by-step process it follows to collect record label information. This section provides a comprehensive overview of the crawler's operation, covering its entire journey from the initial stage to the final output.

3.2 Music Recommender Systems

3.2.1 Bias in Recommender Systems

Recommendation algorithms and machine learning systems are becoming increasingly popular in our daily lives. However, these systems are not immune to biases, which can have a significant impact on the recommendations they make. It is crucial, therefore, to be aware of bias in recommendation algorithms and machine learning systems and take steps to mitigate its effects. To have a better understanding of what biases are, it is recommended to first read section 2.2.1.

Popularity bias

Research has demonstrated that popularity bias towards popular items is mirrored in the consumption behavior of users, resulting in a greater focus on mainstream content over several recommendation iterations [22].

The consequences of popularity bias are particularly relevant in areas such as music recommendations, where it can perpetuate the dominance of established artists, while overlooking new or diverse voices. This can happen for a number of reasons. One possibility is that the system's algorithm is optimized to maximize overall accuracy, which can lead it to focus on popular items because they are more likely to have larger amounts of user feedback and, therefore, can more accurately predict user preferences [15]. Another possibility is that users themselves are more likely to choose popular items, which can create a feedback loop in which the system recommends more popular items, which then become even more popular, and so on.

To determine how artists perceive representation in recommendations, a study interviewed a diverse group of artists [13]. Their argument was that recommender systems should treat item providers fairly and equitably. The study suggested simple re-ranking strategies that discourage biased and repetitive recommendations. Although several methods exist to counteract popularity biases and feedback loops, such as incorporating diverse input sources, considering long-tail options, and using explicit feedback mechanisms to capture user preferences beyond simple popularity metrics [12, 25], identifying existing biases is crucial.

To mitigate popularity bias, personalized re-ranking takes into account user-specific preferences and behavior to adjust the order or weighting of recommended items. It aims to provide recommendations that are not solely based on global popularity but also consider the unique interests and preferences of each user [18]. Moreover, recommender systems based on audio content are not susceptible to popularity bias, and are therefore expected to reveal the "long tail" of music consumption [9].

Demographic bias

Demographic bias in music recommender systems is a well-documented issue. Studies have shown that certain demographic groups receive different recommendations than others. This bias can be based on demographic characteristics such as race, ethnicity, gender, class, sexuality, disability, age, and nationality.

Recommender systems can reinforce or exacerbate social advantages of some artists at the expense of others, leading to potential implications for music consumption. For example, a study found that MRS can lead to gender bias, with female artists being recommended less frequently than male artists, accounting for no more than 25% of all recommendations [24, 28]. Additional gender bias studies reported disparities in item popularities in recommendation lists where female users are exposed to more popularity biased results [21].

Several studies have found that recommender systems can lead to biases related to nationality, such as "home bias", which refers to the degree to which users in a particular country are recommended content from their own country. The most notable study of nationality appears to be that of a Spotify team, Way et al. (2020), who found on the basis of Spotify streaming logs from 2014 to 2019, that preferences for local content have increased through the streaming era, and that trend is consistent across different genres, listener age groups, and registration cohorts [10].

Although some studies have explored demographic bias in MRS, there is insufficient research on biases related to race, ethnicity, social class, and sexuality. This lack of research is likely due to a shortage of potentially sensitive data about racial and ethnic self-identification in the primary existing datasets, as well as the complexity of categorizing people into social classes. This research gap is considered unfortunate since it restricts our comprehension of the possible biases that exist in MRS.

Record label bias

The bias towards major record labels in Spotify's playlists [2] is a symptom of a larger power dynamic issue in the music industry. A few large record labels have an imbalanced amount of control over the production, distribution, and promotion of music. One aspect of this bias is related to the lack of diversity in the artists and genres represented by record labels. Typically, record labels prioritize mainstream, commercial music that has broad appeal and a high profit potential, rather than niche or experimental music that may be artistically significant but may have a smaller audience. As a result, certain genres, such as pop, rock, and hip-hop, are overrepresented in the music industry, while others, such as classical, jazz, and experimental music, are underrepresented.

There is a study that examined the proportion of tracks belonging to major and independent record labels in playlists owned by Spotify. The classification of the labels was based solely on copyright information and did not consider distributor companies. According to the study, 54.1% of tracks in Spotify-owned playlists that were tweeted were identified as major label content. This figure was compared to a random sample of 1001 Spotify samples from 2018, including playlists created by users, artists, and Spotify itself, which had a significantly lower percentage of major label content, at 43.93%. Thus, the study suggests that Spotify may exhibit a bias towards major label content, as opposed to what listeners and third parties promote through the playlists they create [27].

The existence of label biases in the music industry highlights the need for an improved method of gathering record label information. A more precise overview of these biases can be obtained by collecting enriched data, which would also enable recommender system experiments to be conducted. This approach could yield additional insights into the nature of the biases and help to address them.

Overall, the problem of bias highlights the importance of developing recommendation algorithms and machine learning systems that are not only effective but also ethical and equitable. By designing systems that are sensitive to bias and actively work to mitigate its effects, we can ensure that these tools are used to promote fairness, rather than perpetuating existing inequalities.

3.2.2 Fairness in Recommender Systems

There are many ways to define algorithmic fairness [19]. Two common definitions are individual fairness, which means that similar individuals should receive similar treatment, and group fairness, which means that members of a protected group should be treated the same as the rest of the population. In the field of information retrieval, fairness is often defined in terms of exposure [6]. However, it is not clear how to engage fair exposure or what fairness means in the context of music platforms.

Besides algorithmic fairness, there are numerous studies that explore how individuals perceive and rationalize the fairness of algorithms. One study conducted an online experiment where participants were asked to rate algorithms based on their perception of fairness [31]. The results showed that participants tended to rate systems as more fair if they favored their own demographic group, even when the algorithms were explicitly biased towards that group. This effect was observed across different participant groups and varied based on education level, gender, and other aspects of the participants. In other words, this study showed that people tend to perceive algorithms as fair when they receive a favorable outcome, even if the algorithm is biased.

Other work [14] discusses how artists feel about the way they are presented on music platforms. Some artists feel that their artist profiles on the platforms are inadequate because the most popular tracks from years ago are displayed first, requiring users to scroll down to reach the latest album. The context in which artists and their music are presented can also impact their public image, with one artist reporting that the platform's automatically generated playlist features music that they do not like and that is inconsistent with their ideological views. Other artists believe that current music platforms do not provide enough context about the music and emphasize that including more information about the song's history and creation would enhance the user experience. However, they argue that some music may not convey any deeper message and may exist solely for commercial reasons, such as Reggaeton, a genre where explicit videos may be more important for sales than the music itself.

That work [14] also discusses the opinions of several artists on the need for music platforms to be more transparent. They express concerns about how the platforms promote certain artists over others and how the recommendation system works. Some artists feel that transparency is particularly important for independent artists, and others believe that the platforms should explain to artists what they need to do to be recommended more often. They feel that the goals of music platforms are profitable only for some stakeholders, with an upper and lower class.

To address fairness concerns in MRS, researchers have proposed various approaches, such as using multiple recommendation algorithms to avoid relying on a single biased model [3], incorporating fairness metrics into the recommendation process [4], and considering user preferences and diversity in the recommendation process [7].

3.2.3 Feedback Loops in Recommender Systems

Prior research in the domain of feedback loops (section 2.2.3) in music recommender systems demonstrated that various recommendation algorithms magnify existing bias through successive iterations of user interactions (implicit feedback loop). Notably, it was observed that bias amplification was stronger for females, who constitute a minority group in terms of population and number of interactions, compared to males [22].

In relation to the preceding study on record label distribution, initial feedback loop effects were identified. While independent labels had a dominant presence in the dataset under analysis, major labels exhibited an over-representation in the recommendation process. This bias towards major labels, particularly Universal and Warner, was further magnified with successive iterations of the feedback loop simulation. They suggested the importance to conduct further analysis to examine the impact of popularity on the recommendation process, as it seemed to be driven by a bias towards highly successful tracks. [26].

4. Dataset Research

In any data-driven project, the availability of relevant and high-quality datasets is crucial for the success of the project. The process of searching for datasets involves identifying potential sources, evaluating the quality and relevance of the available datasets, and selecting the most appropriate dataset for the project.

This section of the thesis will explain this process of searching useful datasets that match the requirements of our project, including the criteria for identifying relevant datasets and the sources of datasets. A successful search process can significantly enhance the validity and reliability of the research outcomes.

4.1 Dataset Requirements

The datasets must meet certain requirements in terms of quality, completeness, and relevance. Here are some of the key requirements that must be satisfied to be suitable for training the music algorithm model in the feedback loop simulation scenario:

1. **Quality:** The dataset should be of high quality, with accurate and consistent data. It should also be free from errors, duplicates, and missing values.
2. **Size:** The dataset should be large enough to provide sufficient data points for training the music recommendation model. A larger dataset can help improve reliability of the model.
3. **Diversity:** The dataset should include a wide range of music artists, and songs. This will help ensure that the music recommendation model is able to provide recommendations for a diverse set of users and preferences.
4. **Relevance:** The dataset should be relevant to the target audience and the context in which the music recommendation model will be used.
5. **Data privacy:** The dataset should be obtained and used in compliance with data privacy regulations and guidelines. Appropriate measures should be taken to ensure that user privacy is protected throughout the data collection, storage, and processing stages.

Apart from fulfilling the fundamental criteria previously mentioned, there exist some additional prerequisites for a music dataset to be considered appropriate for this thesis. The dataset also needs to have a minimum level of information so that the data augmentation step can be carried out efficiently, as this step is responsible for gathering the necessary details about the record labels.

In previous research, the necessary information for running the crawler (section 3.1.1) was highly specific and challenging to locate across multiple datasets. However, due to one of the objectives of this thesis, the required minimal information has been simplified to a more fundamental level.

Previously, album or track URIs (section 2.1.1) were necessary, but now just the name of the track, album, or artist suffices. This is highly advantageous since this basic information is available in almost any dataset, regardless of whether it is anonymous or not with the user. Moreover, it is essential for these datasets to have a connection to the user in some manner, such as through user-generated playlists or listening history. This ensures that the data we obtain is relevant and can be used to efficiently train our recommendation algorithm.

4.2 Dataset Selection

It is important to carefully choose the appropriate datasets that align with the research question and objectives. The goal is to provide a comprehensive understanding of the dataset selection process and to ensure that the selected datasets are reliable, relevant, and representative of the research question.

The first datasets selected for this thesis were evident, as they were already used in previous work. The first two datasets selected were the MPD and the LFM-2b datasets. This approach allows for a more exhaustive analysis on these datasets previously used, in addition to enabling the comparison of the results obtained from these datasets with those from new datasets.

Subsequently, the exploration for additional datasets started. The study [30] provides a compilation of various datasets utilized in music recommendation research. Among these datasets, we selected several to determine their relevance to our specific project, but only a few proved to be suitable. The datasets selected for the first phase of validation included Spotify Playlists, #nowplaying, #nowplaying-rs, MusicMicro, MSSD, MSD, and MMTD.

4.2.1 Discarded datasets

Upon analyzing each of the datasets, it became possible to eliminate those that did not fulfill the fundamental criteria for our thesis.

The **#nowplaying** and **#nowplaying-rs** datasets were found to be interesting due to the significant amount of data collected from Twitter, including listening events of 139K users and 346K tracks. However, further analysis revealed that not only the user but the track names were anonymous, making it impossible to collect album URIs and subsequently record label information using the crawler. This limitation made the datasets unsuitable for our thesis.

The situation was similar with the Spotify **Music Streaming Sessions Dataset (MSSD)** [8], which includes 160 million streaming sessions along with user interactions, audio features, metadata for tracks streamed during sessions, and snapshots of playlists listened to during sessions. However, upon examination, it was discovered that all of this information had been anonymized, with the track names replaced by track IDs that could not be linked to actual track names.

The **MusicMicro** [29] dataset presented a different situation. It consisted of three subdatasets, with the primary one being Twitter listening events that contained track and artist names in ID format, while the other two subdatasets provided the mappings of these track id and artist id with their respective names. After analyzing this dataset and attempting to merge the three subdatasets to obtain the primary one with non-anonymized data, it was discovered that the IDs did not match and were misspelled, resulting in the merge operation yielding no matches.

4.2.2 Chosen datasets

After the elimination process, some datasets that fulfilled the required criteria to be included in this thesis were identified. The MPD and LFM-2b were the two pre-selected ones and the remaining ones consisted of those that passed the elimination process.

The **MPD**, also known as **The Million Playlist Dataset**[4], is a collection of 1 million playlists sampled from over 2 billion public playlists on Spotify. It contains more than 2 million unique tracks from almost 300,000 artists and is considered the largest dataset of music playlists in the world. These playlists were created by Spotify users between January 2010 and November 2017 and include metadata such as track IDs, playlist titles, and other information to protect user privacy.

The **LFM-2b**[5] dataset consists of the listening histories of more than 120,000 Last.fm users, encompassing over 2 billion individual listening events from February 2005 to March 2020. These events correspond to 50 million unique tracks by 5 million unique artists. In addition to the standard metadata of artist and track names, the LFM-2b dataset also includes additional information about users, such as their country, gender, and age, as well as detailed genre and style information and vector embeddings of track lyrics.

The **Spotify Playlists**[6] dataset is based on the subset of users in the nowplaying dataset who publish their nowplaying tweets via Spotify. In principle, the dataset holds users, their playlists and the tracks contained in these playlists. This dataset contains around 13 million tracks associated with playlists created by users.

The **Million Musical Tweet Dataset (MMTD)** [17] is obtained from crawling and examining tweets consistently for over 500 days, which is equivalent to 17 months. It is the largest publicly accessible database of music listening histories based on microblogs, containing informa-

[4]Spotify Million Playlist Dataset Challenge
[5]LFM-2b dataset web page
[6]Spotify Playlists Kaggle Notebook

tion on location, time, and other contextual details. The MMTD stands out from other datasets of music listening histories due to its comprehensive information.

The **Million Song Dataset** [5] is a freely-available collection of audio features and metadata for a million contemporary popular music tracks. The core of the dataset is the feature analysis and metadata for one million songs, provided by The Echo Nest. The dataset does not include any audio, only the derived features.

The thesis provides a comprehensive overview of the contents of each dataset in Appendix A, offering a detailed description of their specific characteristics and data composition.

5. Methodology: Data Gathering

This section provides an overview of the concepts and ideas behind the data augmentation step. In this section, the preprocessing step will be comprehensively discussed, which is an extension proposed by this thesis to the existing crawler developed in previous studies [20]. Additionally, the subsequent crawler steps will be explained with less emphasis on detail.

According to [20], the data augmentation process is founded on two underlying assumptions.

- **A1.** All tracks within an album have the same record label. There is no difference in ownership between individual tracks.
- **A2.** Major record labels have a vested interest in correctly marking the music they distribute or own with the appropriate rights.

Consequently, any classification made for one track can be extended to all other tracks within the same album, making album-level classification practical. Additionally, the copyright information available on Spotify is assumed to be accurate and the absence of copyright information is believed to be indicative of an *Independent* record label.

To classify record labels based on their granularity levels a naming convention is introduced. This convention will be used to facilitate the analysis of the impact of record labels on the recommendation process.

- The first level is the **low-level record label**, which is the first connection of an album or track to a record or music distribution company.
- The second level is the **major record label**, which includes Universal Music Group, Sony Music Entertainment, and Warner Records, as they are considered the major players in today's music market.
- Finally, the **top-level class** is the highest level of classification, which includes the **major record labels** and an additional class called **Independent**, which summarizes all other small labels including the unknown ones.

In order to explain the steps involved in enriching a dataset with record label information, it is necessary to first provide a brief overview of the structure of the classification process.

The full crawler comprises multiple steps, with each step building on the output of the previous one. The initial step, referred to as the "(1). Preprocessing", has been specifically developed within this thesis to facilitate the integration of any dataset into the crawler. On the other hand, the development of the subsequent steps has already been completed, and the only task remaining is to integrate them with the new preceding stage.

1. **Preprocessing:** The initial stage involves getting the dataset ready with the required information for carrying out the subsequent steps, specifically album URIs from Spotify. If the dataset already has this information, it just needs to be properly formatted for efficiency. However, if the dataset lacks this information, a script must be run for each entry in the dataset to collect this information.

2. **Spotify low-level crawler:** Starting from a list of album or track URIs (unique resource identifiers), the Spotify-API[7] is used to create a list of low-level record labels, which serves as a base for the label crawler.

3. **Label Crawler:**

   (a) **Mapping trivial cases:** In a first pass, all low-level record labels with trivial names are mapped to top-level classes based on matching tokens. Example: The low-level record label Universal Group belongs to major label Universal Music Group.

   (b) **Discogs label crawler:** Discogs[8] is a public, user-generated music information platform and marketplace with detailed metadata. It is used to link and classify low-level record labels using the provided API.

   (c) **Wikipedia label crawler:** Similar to the previous step, the label information is harvested from Wikipedia[9]. Unlike Discogs, Wikipedia provides information in a less structured way.

   (d) **Interim label mapping:** This step is responsible for evaluating collected information from the previous crawler steps to determine its relevance and usefulness. It requires traversing the extracted label hierarchies to identify top-level companies or previously classified labels.

   (e) **Copyright classification:** To recheck assignments made, we further analyse copyright information obtained in the first, preprocessing step to create an alias dictionary of frequent and decisive copyright tokens. The idea is that this information is usually more descriptive, hence by identifying frequent terms for known major assignments, additional links can be uncovered. This is used for both, classification of still unassigned labels and correction of previous assignments.

   (f) **Final label mapping:** For all still unknown and unclassified low-level record labels, we assume no connection to a major label and hence classify them as Independent. In this final step, also a manual check-up and possible corrections can be applied.

4. **Postprocessing:** Use the classification map created from the low-level record label to high-level class (major record label) to enhance the initial list of tracknames, album URIs or track URIs generated during the preprocessing phase.

During the Spotify low-level crawler step, an empty label map containing only low-level record labels is generated as an artifact. Each classification step then produces, at minimum, a new

[7]https://developer.spotify.com/documentation/web-api
[8]https://www.discogs.com/my
[9]https://en.wikipedia.org/wiki/Main$_{Page}$

label map with a filename suffix corresponding to the step, along with other artifacts such as archives to store previously executed lookups for the crawler steps. Each classification that occurs in every step is saved independently to allow for a statistical assessment of the classification progress in each step. These classifications are distinguished using flags to identify unsuccessful attempts, such as reaching the maximum depth limit during a crawler step.

To classify record labels, the top-level class *Unknown* is used as a temporary classification category until the final label mapping step, where all such classified low-level record labels are designated as *Independent*. This is due to a distinction at the start of the crawler process where certain album URIs have no record label information on Spotify (empty field, N/A or NaN values) while others are explicitly marked as Independent. Additionally, this distinction is used in the analysis of the stepwise classification progress of the crawler to ensure transparency throughout the process. The decision to merge *Unknown* and *Independent* in the final step was based on the belief that a lack of significant information during all previous crawler steps indicates that missing information itself is a characteristic of an album or low-level record label classified as *Independent*.

5.1 Preprocessing step

This step is placed at the beginning of the classification process, and aims to achieve one of the primary goals of this thesis, which is to develop an extremely effective preprocessing method that can be used to incorporate any type of music dataset as an input for the crawler. This enables record labels to be obtained for any dataset with minimal requirements.

5.1.1 Fitting any Dataset for Crawler Execution

The first part of the preprocessing is tasked with converting a common dataset, like any other, into a suitable dataset to be executed in the crawler. Previously, the crawler was only able to execute on specific datasets, such as MPD and LFM-2b.

To comprehend the various scenarios that can arise with musical datasets, it is crucial to understand the definition of a URI first. The specific definition can be found in section 2.1.1 of the document.

The easiest way to get record label information is through album URIs. This is because getting the record label for an album also gives us the information for all its tracks. Plus, reducing the dataset from tracks to albums, we make fewer calls to the Spotify API, which saves time.
After comprehending the importance of the concept and why it is essential to possess this information to gather record label information using the Spotify API, we can elaborate on the various scenarios.

- **Dataset with album URIs:** These datasets already have the necessary information to input the crawler. They just have to be preprocessed iterating over the dataset to build a file that collects album URIs with any other information available from that particular dataset (for example, album name, artist name, etc.) and the occurrences of each album and converts them in a listing dataset. An example of these datasets is the MPD one.

- **Dataset with track URIs:** In contrast to the previous situation, these datasets lack the essential information required to input the crawler, such as album URIs. However, they contain enough information to obtain them with ease through track URIs. During the preprocessing stage, it is possible to acquire this information through the track URIs by making a simple request to the Spotify API. When working with such datasets, the initial step involves transforming track URIs to album URIs therefore they become optimal for the crawler. An example of these datasets is the LFM-2b one.

- **Dataset without URIs:** These datasets are the most common, since it is not normal to have such precise information as the URIs for each track. To retrieve the album URIs associated with each track, certain basic information is required, specifically the *artist name* and *track name* for each individual song. In cases where this information is unavailable, it may not be possible to effectively obtain the corresponding album URI. Assuming the required information is present, it is advisable to first group together all occurrences of each song within the dataset to avoid duplicate Spotify API calls. The process then involves calling the API using a query that includes the *artist name* and *track name*, which returns the associated album URI. These URIs are stored for later use in subsequent stages. It is important to note that this process can be time-consuming as the API calls are made one by one.

The code used to collect album URIs from datasets for which only trackname or artistname is available can be found in the github repository[10].

To establish a connection with the Spotify API and thereby execute the required queries for retrieving album URI information, several steps need to be followed:

1. **Obtain a Spotify Developer Account:** Go to the Spotify Developer website and sign in or create a new account.[11]

2. **Create a New Application:** Once logged in, create a new application by providing necessary details such as the name, description, and redirect URI.

3. **Obtain Client ID and Client Secret:** After creating the application, you will receive a unique Client ID and Client Secret. These credentials are essential for authentication when making API requests.

4. **Implement Authentication in Python:** In your Python code, use the *Spotipy*[12] library to handle the authentication process. Install the library using *pip install spotipy* and import it into your project.

[10]https://github.com/DaniGmzGnz/TFG_RecordLabelAnalysis
[11]`https://developer.spotify.com/dashboard`
[12]`https://spotipy.readthedocs.io/en/master/`

5. **Configure API Access:** Provide your Client ID and Client Secret obtained from the Spotify Developer Dashboard as credentials in your Python code. It is recommended to use a separate file.

6. **Connect to the Spotify API:** Use the *spotipy* library or other HTTP request libraries like requests to send requests to the Spotify API. With the access token, you can access various endpoints and retrieve the desired information, such as album URIs.

5.1.2 Enhancing Crawler Efficiency

The second part of the preprocessing is tasked with improving the efficiency of the crawler's execution. Results obtained from previous crawled datasets can be reused to perform mappings before introducing a new dataset to the crawler. This approach ensures that the crawler only searches for new and unique data within the new dataset, which has not been previously obtained from any other dataset.

Considering the factors mentioned earlier, the following workflow (figure 1) has been implemented to improve the efficiency of the crawler's execution time for the collection of record labels for the datasets:

1. Once you have a new suitable dataset, it is recommended to perform a groupby operation based on the attributes {trackname, artistname} or just {trackname} before using the crawler on it. Doing so helps to significantly decrease the number of calls made to the Spotify API.

2. Execute the script with the grouped dataset in order to obtain the file containing all the album URIs.

3. If the first stage of the crawling process has not been done previously for any other dataset, the entire crawling process is executed for this new dataset. Otherwise, the resulting mapping of album URIs to low-level record labels (obtained at the end of first stage) of previous datasets is reused for this new dataset in the following way:

    (a) The album URIs that match are stored in a file for later use.

    (b) The album URIs that do **not** match are fed into the crawler to execute the entire crawling process and obtain the corresponding low-level and major record labels.

    By following this approach, the entire crawling process is only executed for a specific portion of the dataset, rather than the entire dataset.

4. If the entire crawling process has not been done previously for any other dataset, take the Album URIs that matched to low-level (3.a) and use the output mapping from low-level record labels to major record labels obtained for the portion of this particular dataset (3.b) in the following way:

    (a) The low-level record labels that match with major record labels are stored in a file for later use.

(b) The low-level record labels that do **not** match with major record labels are fed into the next stages of the crawler to obtain the corresponding major record labels.

Otherwise, the resulting mapping of low-level record labels to major record labels (obtained at the end of the crawling) of previous datasets is reused for this new dataset to map the low-level record labels that have been previously matched to major record labels.

5. Finally, a merge operation is performed where different parts of the dataset that were previously separated will join. This includes the file of album URIs that matched both low-level and major record labels (4.a), the one that matched low-level labels but failed to match major ones and were sent for crawling (4.b), and album URIs that did not match anything and required a complete crawl from the start (3.b).

5.2 Low-level Crawler step

The Spotify API is used to collect low-level record labels for the dataset by crawling through a list of album URIs. It is also possible to begin with a list of track URIs, in which case a previous step is introduced to gather complete track information from Spotify, including the album URI. The album URIs are then sorted in descending order based on their frequency in the dataset and are used as input for the next step.

The Spotify crawler goes through the album URI list and retrieves complete album information from the Spotify API. The information obtained from each album includes the low-level record label, as well as two distinct types of copyright information (P and C), which are then saved for future use. This step is typically less time-consuming than others since API calls can be packaged in groups of 20 entries.

The resulting list of low-level record labels serves as the foundation for the multi-step classification process and is referred to as the label map from this point forward, as it maps low-level record labels to their classified major labels. It is also possible to reuse a pre-existing label map, such as one created from a previous crawler run on a different dataset. In such cases, the newly generated label map is populated with all existing classifications that match the name of the low-level record label.

5.3 Label Crawler step

This step forms the core structure of the crawler, and it comprises various stages aimed at transitioning from a low-level record label (obtained in the preceding step) to a major record label. Each stage may or may not obtain this information, and as the stages progress, more entries in the dataset acquire a major record label. After the last stage, any entries that have not obtained a major record label are deemed independent, as no relevant information has been retrieved for them.

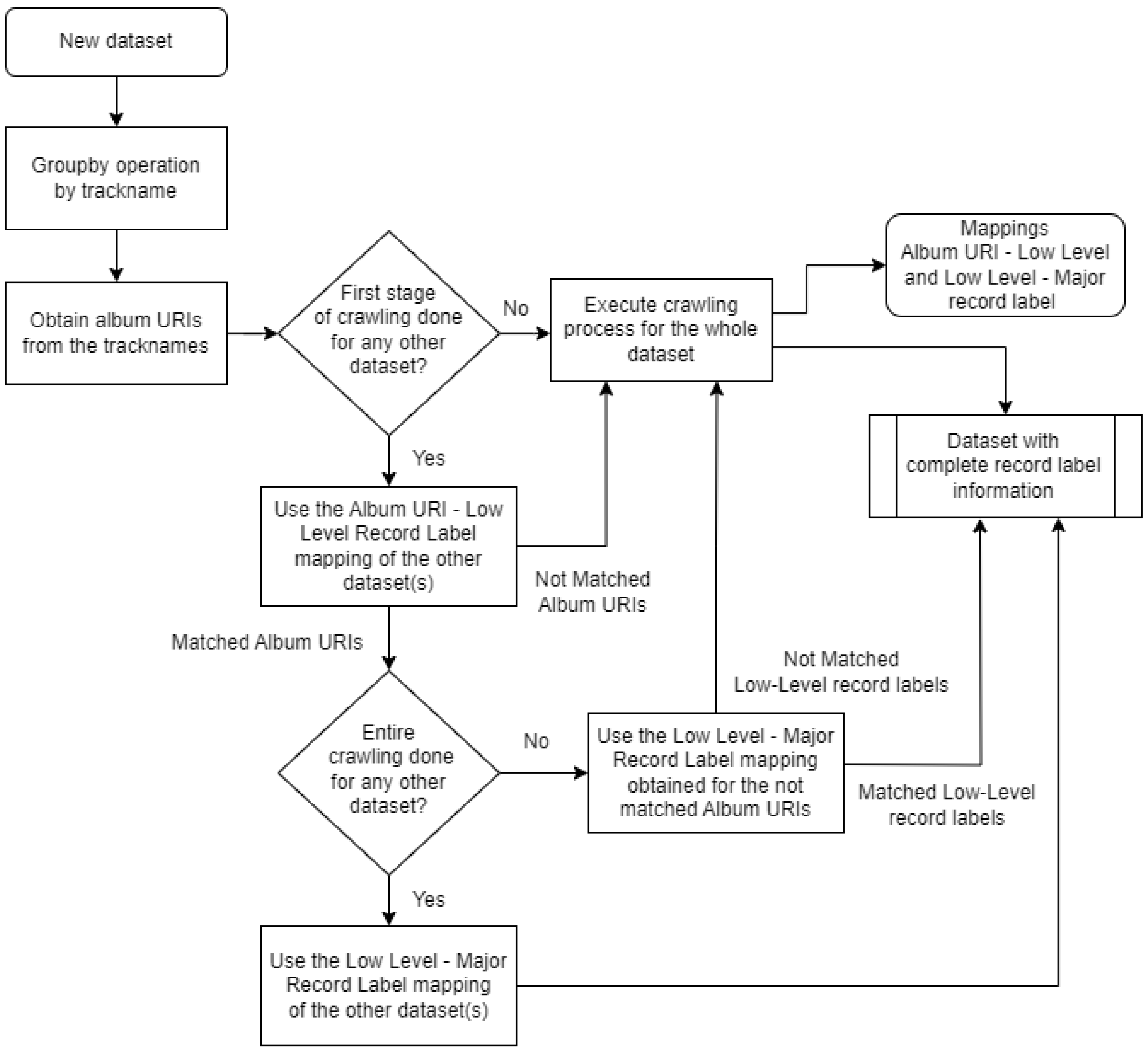

New dataset
Groupby operation by trackname
Obtain album URIs from the tracknames
First stage of crawling done for any other dataset?
No
Yes
Execute crawling process for the whole dataset
Mappings Album URI - Low Level and Low Level - Major record label
Dataset with complete record label information
Use the Album URI - Low Level Record Label mapping of the other dataset(s)
Not Matched Album URIs
Matched Album URIs
Not Matched Low-Level record labels
Entire crawling done for any other dataset?
No
Use the Low Level - Major Record Label mapping obtained for the not matched Album URIs
Matched Low-Level record labels
Yes
Use the Low Level - Major Record Label mapping of the other dataset(s)

Figure 1: Workflow diagram illustrating the steps followed to optimize the execution time of the crawler for collecting record labels for new datasets reusing previously obtained results.

The initial stage is **mapping trivial cases**. Certain low-level record labels in the label map can be immediately classified as major labels, as their names are identical or similar to those of major labels. To achieve this, a dictionary of aliases is employed for each major class, with a list of aliases assigned to each major label. The aim of this stage is to prevent unnecessary look-ups.

The following stage starts with a label search on **Discogs**, using the name of the low-level record label. Only the first search result is used for the subsequent lookup. If the search result is empty, a flag is set for this entry in the label map, and the lookup process is terminated. If a search result exists, the Discogs page is analyzed to determine if a parent label entry matches one of the major labels. However, if a parent label exists but it does not match any major label, the lookup process is repeated recursively for the parent label. The result of the parent label lookup is then set as the result for the current label. During the lookup, if any of the label's sites contain a link to the Wikipedia page, the link is saved and used in the next stage as a shortcut to skip the Wikipedia search.

The **Wikipedia** stage is used to gather additional information on the low-level record labels which could not be classified in the previous steps. The first step is to check if a Wikipedia link exists in the label map from previous crawling on Discogs. If such a link exists, the corresponding Wikipedia page is crawled for relevant information. If no link exists, a search is started via the Wikipedia-API on the low-level record label name, restricted to the English version of Wikipedia. The search may return no results or a Wikipedia page that does not contain the desired information.

The **interim** stage involves analyzing additional information obtained from the Wikipedia crawler to determine if a low-level record label can be classified as an independent label or to another record label. The evaluation is based on the sum of aggregated keywords and the presence of a link to the independent label page of Wikipedia. If the sum of the keyword aggregate is below a threshold, the entry is classified as an independent label but if it is above another threshold, the entry is classified to the record label with the maximum value in the keyword collection. If there is a tie, no classification is made. This step is motivated by the assumption that the keyword collection has significance, and if no parent label is found on Discogs or Wikipedia, the keyword aggregate can be a reliable classification source.

The next stage is the **copyright** one, that involves evaluating copyright information gathered through the use of the Spotify-API. This involves tokenizing the copyright information of each album and merging both P and C types of copyright. The copyright tokens are then aggregated by counting their occurrences and sorted descending. The top tokens are used to create an alias dictionary for big sub labels of the major labels. This dictionary is used to assign low-level record labels to major labels by analyzing the copyright tokens and checking for matches in the alias dictionary. This process overwrites the classifications done by previous stages, as it is assumed that the alias dictionaries are more accurate. This is motivated by the assumption that major labels have a high interest in keeping the copyright information on Spotify as correct as possible.

The last stage of the classification process, **final label mapping**, involves using a dictionary of corrections to assign single low-level record labels to a major label. This is helpful for cases where there are big low-level record labels that are independent or where manually researched classifications of bigger record labels are not yet classified. Additionally, the information from the Discogs crawler is used in this step to assign labels to major labels based on the highest value of the Discogs keyword aggregate, if it is above a certain threshold. If a label cannot be assigned to Universal Music Group, Sony Music Entertainment, or Warner Records at this point, it is classified as Independent. This is because it is assumed that low-level record labels with no significant information for classification on Discogs, Wikipedia, or in their copyright information are likely from an independent label.

5.4 Postprocessing step

During postprocessing, the classification output, which maps low-level record labels to major record labels, is mapped back to the corresponding trackname and artistname, album URIs or even track URIs, depending on the approach used during the preprocessing step. In addition to the major label, the output also includes the low-level record label and copyright information. This supplementary information is deemed useful for future recommender system experiments.

6. Methodology: MRS Experiments

Throughout this section, we will explore the methodologies and techniques involved in training the user-based collaborative filtering models using user data and performing feedback loop simulations.

To conduct our experiments two scripts will be developed, each responsible for a specific task. The first script, will be responsible for constructing the user-item matrix (section 2.2.2) structure to train the collaborative filtering model. The second script, will handle the execution of the feedback loop simulation.

We will use the datasets that have been enriched with record label information using the crawler. These datasets contain data on user interactions with tracks, which serves as the basis for training the collaborative filtering model.

In order to ensure compatibility with our scripts, each dataset should adhere to the following general structure:

- **train_dataset.csv:** This file is crucial as it includes the user interaction information. It is imperative that the file comprises the following columns:
  - **user_id:** Unique integer identifier for each user.
  - **track_id:** Unique integer identifier for each track.
  - **timestamp (optional):** Date and time the user listened to the track.
- **labels_dataset.csv:** This file contains the gathered record label information for each track. It is imperative that the file comprises the following columns:
  - **track_id:** Unique integer identifier for each track.
  - **major_record_label:** A string describing the assigned record label.

6.1 User-Item Matrix Generation

In our specific case, we adapt the concept of the user-item matrix described in the theoretical background of this thesis. Here, the items represent the tracks that users have either listened to or not. The interaction strength in the matrix is determined based on whether a user has listened to a track or not. If a user has listened to a track, the corresponding cell in the matrix will contain a value indicating the strength of that interaction. Conversely, if a user has not interacted with a track, the cell will remain empty to reflect the absence of an interaction.

Prior to constructing the user-item matrix, the script initially splits the datasets into training and test sets. To ensure an efficient split for any dataset, consideration is given to whether the dataset contains timestamp information. If the dataset contains this information, the split is performed based on the oldest and most recent dates available, partitioning from a designated point (e.g., a 90% split). However, if the dataset does not include timestamp information, a random split of the dataset rows is conducted, with the chosen percentage also taken into account.

Once the split is completed, we iterate through the partitions to construct two dictionaries, train and test. These dictionaries store the information about the tracks listened to by each user.

Subsequently, a filtering process is applied to delete users who have listened to fewer than a determined number of tracks and tracks that have been listened to by fewer than another determined number of users. This filering is done to address data sparsity issues, reduce noise, improve similarity calculations, and optimize the general performance. Nevertheless, it is crucial to select the threshold for removing users with low interactions cautiously to prevent the exclusion of valuable insights. As a default, we set these hyperparameters to values of 10 and 15, respectively.

Lastly, we construct the training user-item matrix designed to incorporate information about whether each user has listened to a particular song or not, at least once. Additionally, we prepare the data for the test users, along with two dictionaries. One dictionary maps the tracks, while the other maps the users. These 4 objects are saved for the simulations.

6.2 Feedback Loop Simulation

To gain a comprehensive understanding of the simulation setup, it is advisable to familiarize with the concept and functioning of feedback loops (section 2.2.3).

In this thesis, the datasets used do not contain explicit feedback from users. As we are conducting experiments in an offline environment, we need to determine the approach for simulating the feedback given by the users.

To enhance the scope of the experiments, we will not only increase the number of datasets used but also incorporate up to three distinct recommendation algorithms. Furthermore, another approach for simulating user feedback will be introduced.

6.2.1 Simulation Approach

To address the research questions of this thesis and examine the reciprocal effects of record label distribution in multiple iterations of a recommender system, the experimental designs described in [13] and [20] were employed. The first design utilized the ALS algorithm to analyze the gender distribution of artists within feedback loops. Additionally, the second study adapted upon that setup to investigate the distribution of record labels in recommendation systems. The last one, therefore, has been reused and improved for this thesis.

In order to conduct the simulations, a user-item matrix containing the interactions of users with tracks enriched with record label information exists. The objective is to simulate feedback loops, and this is achieved through two types of user behavior, detailed at section 2.2.3. In the first type, users accept the first 10 recommended tracks, while in the second type, they randomly select 10 tracks from the pool of 100 recommendations in each iteration. As a result, the interactions for these 10 tracks are increased in the user-item matrix for each user.

After each iteration, the recommendation algorithm, which is based on matrix factorization, is retrained using the modified user-item interaction matrix. This process leads to the creation of a new recommendation model at the conclusion of each iteration. This iterative procedure is repeated for a total of *n* iterations, allowing for the exploration of the evolving effects of user feedback on the distribution of record labels.

To further enhance the analysis, the simulations are carried out using three distinct recommendation algorithms: ALS, BPR, and LMF (section 2.2.2). By employing these different algorithms, the study aims to gain a comprehensive understanding of how record label distribution can vary depending on the model used for recommendations.

6.2.2 Metrics

To gain a comprehensive understanding of the recommender system's behavior, specific metrics are introduced. These metrics are designed to measure the probability and coverage of a record label within the recommendations. They are selected based on the metrics employed in previous research [20], ensuring consistency and comparability in the evaluation process.

- **First:** This metric calculates the average first position of a specific major label in the 100 recommendations across all users.
- **Recommended:** This metric represents the percentage of tracks from the 100 recommendations that are associated with a specific major label. It is calculated by averaging the coverage across all users.

7. Dataset Analysis

This section explores the results of the label crawler setup and presents descriptive analysis of the datasets used in this study. We will present visualizations and statistical summaries of the gathered data to gain insights into the properties of the datasets and to inform our subsequent analyses.

7.1 Descriptive Analysis

At each classification step, the crawler saves the current progress for analysis purposes. This allows for examination of not only the final distribution of classified major labels but also the gain in classification achieved after each step is completed.

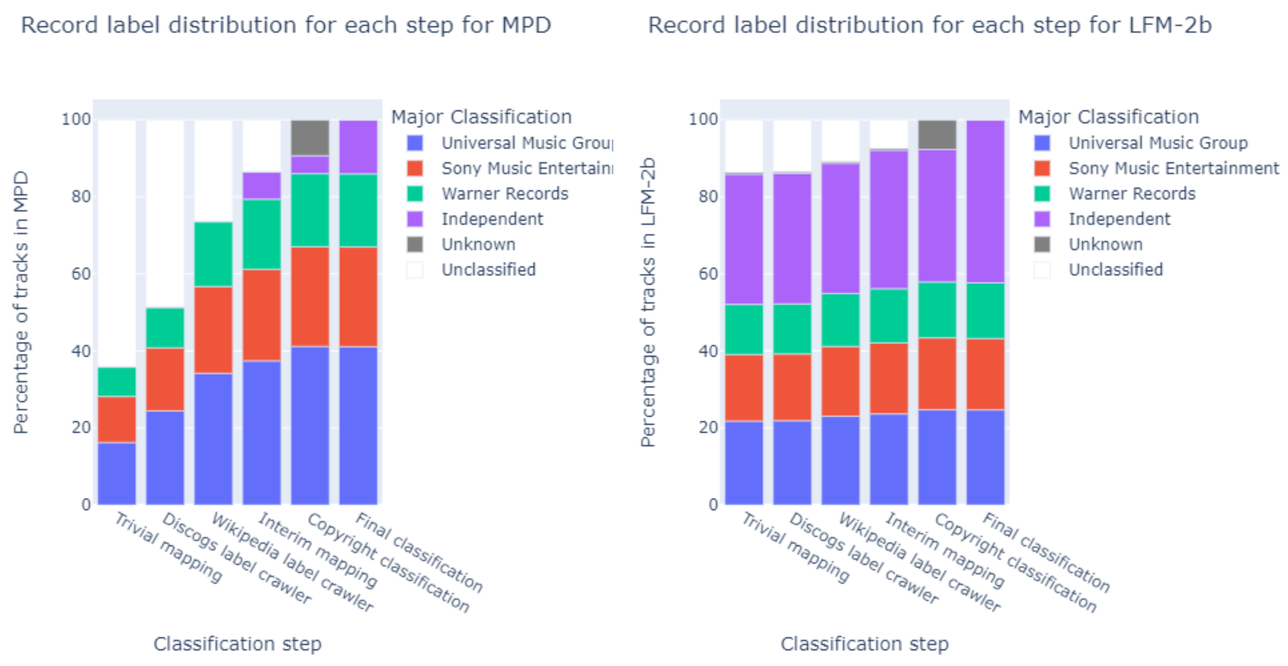


Figure 2: Development of major label assignment across individual steps of the crawler for MPD and LFM-2b datasets.

The results displayed (figure 2) showcase the outcomes of the process for the MPD and LFM-2b datasets. These results have been retrieved from previous work [20] as it would have been time-consuming and impractical to re-execute the crawler for these datasets when the precise results were already accessible.

The crawler initiates the classification process with an empty label map when working with the MPD dataset. However, in the case of the LFM-2b dataset, it leverages the pre-existing label map obtained from the MPD. This explains the significantly higher level of classification

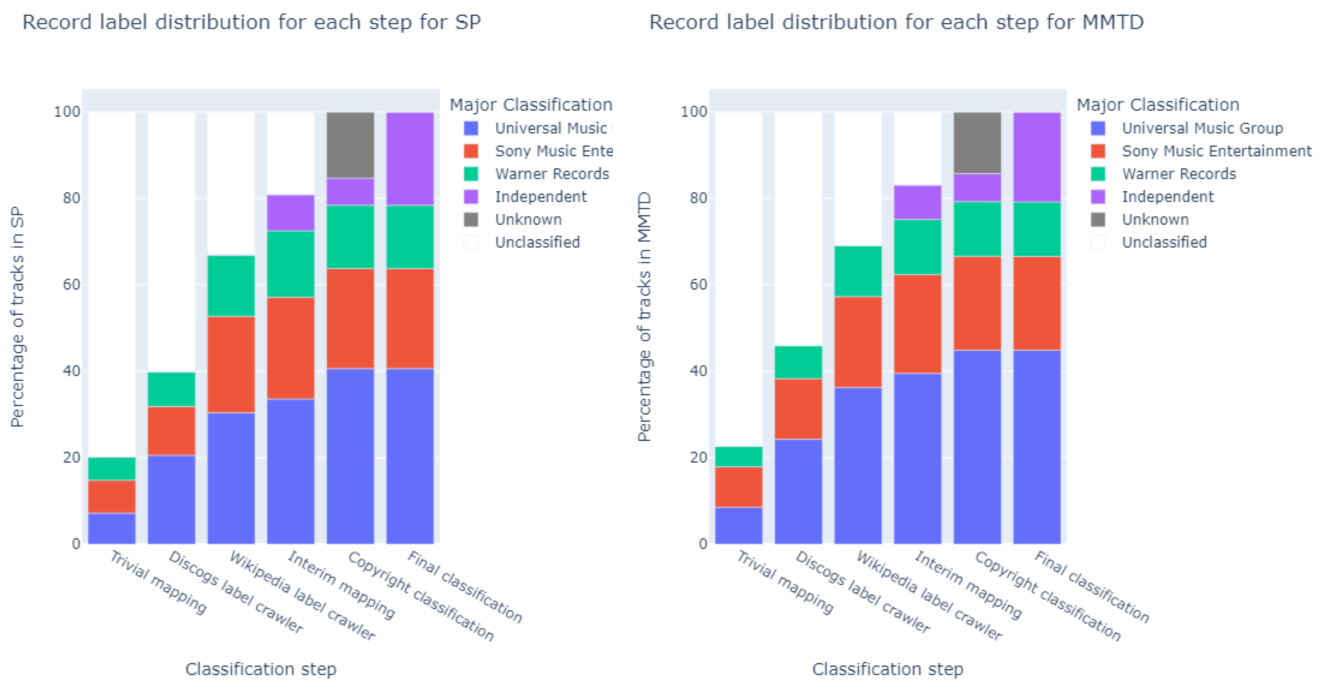


Figure 3: Development of major label assignment across individual steps of the crawler for SP and MMTD datasets.

achieved right from the start in the LFM-2b dataset.

The distinction between *Unknown* and *Unclassified* in the stepwise gain chart can be clarified as follows: entries that contain invalid or non-existent low-level record label names (such as "N/A" or "nan") are categorized as "Unknown." On the other hand, "Unclassified" encompasses all the low-level record labels that were either not addressed during the current classification step or were unable to be classified.

Similar to the approach followed for the MPD dataset, we chose to initialize the crawler with an empty label map for the three new datasets. This decision was made to obtain more accurate insights into the stage at which the major record labels are being assigned.

Figure 4 illustrates our latest dataset, where we emphasize that it exhibits the highest level of record labels classified as *Unknown* compared to the other four datasets, since the crawler does not seem so effective obtaining the labels in its stages.

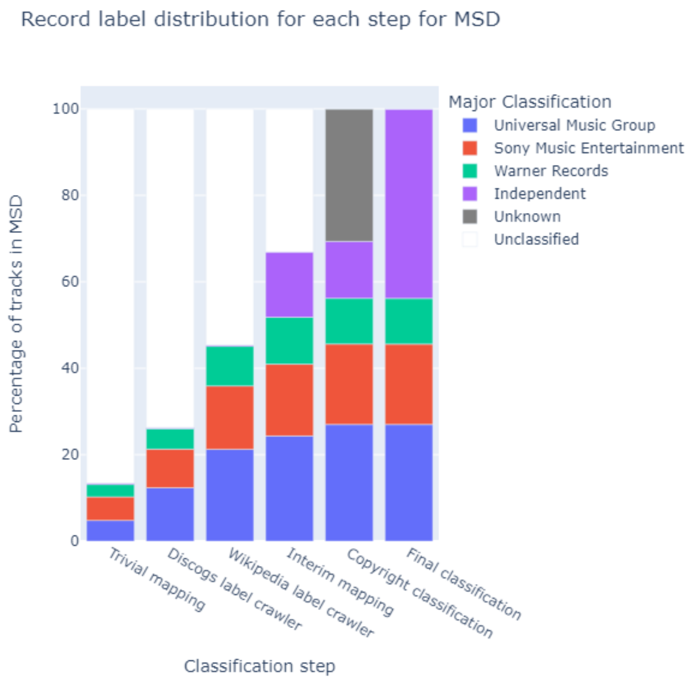


Figure 4: Development of major label assignment across individual steps of the crawler for MSD dataset.

7.2 Bias Analysis

A general overview over the distribution of low-level record label names and their assignment to major labels will help to identify biases regarding record labels in the datasets. Then, a more detailed examination of the distribution and diversity of top-level classes within subsets, such as playlists, will be conducted using the Simpson index. This in-depth analysis will provide insights into the composition and variety of record labels in the respective datasets.

The starting point for the bias analysis is the list of low-level record label names. To gain an understanding of their distribution, word clouds are used to visualize the importance of specific terms within this collection of words. In figures 5, 6 and 7, the word clouds display the top 100 label names derived from the list of low-level record labels, weighted by their frequencies.

To focus on the most important names across all low-level record labels, bigrams are disabled, and a minimum token length of 2 is set. This overview focuses mainly on the names of labels after removing a predefined list of stopwords provided by us. Furthermore, the script does not include any functionality to resolve abbreviations, which could have resulted in higher prominence for certain tokens, such as 'SME' representing 'Sony Music Entertainment'.

There are other plots, like treemaps, that provide an overview of the distribution of low-level record labels after being classified into a final class. However, they are visually complex. An interactive version of the plots, allowing pinning and zooming, can be accessed in the Google Colab notebook[13] .

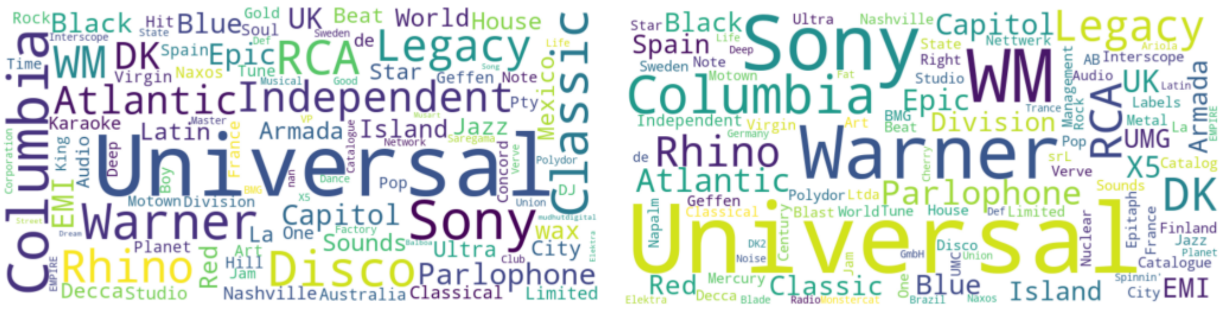


Figure 5: Word clouds of top 100 low-level record label names for MPD and LFM-2b datasets.

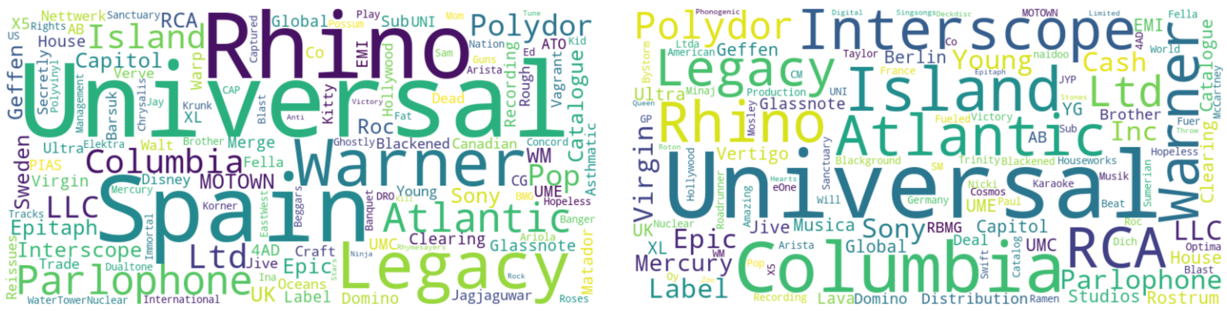


Figure 6: Word clouds of top 100 low-level record label names for SP and MMTD datasets.

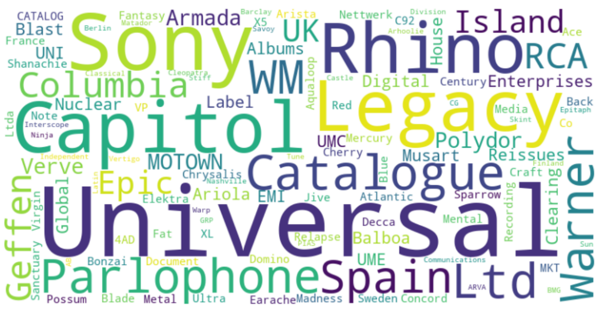


Figure 7: Word cloud of top 100 low-level record label names for MSD dataset.

Figure 8 presents the distribution of low-level record labels categorized into major label classes for all five datasets.

[13]https://colab.research.google.com/github/DaniGmzGnz/TFG$_R$*ecordLabelAnalysis/blob/main/Treemap_analysis.ipynb*

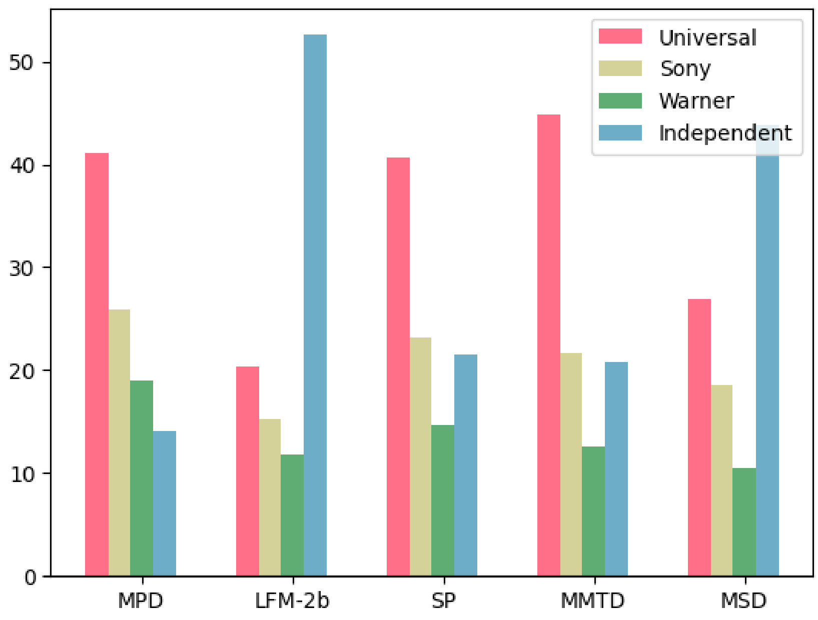


Figure 8: Distribution of low-level record labels categorized into top-level classes for all five datasets.

7.2.1 Users' Profile Analysis

By calculating the Simpson index (section 2.2.1) for different users within a music dataset, you can compare the diversity or concentration of the major record labels of each track selected by the user. If the index for a particular label is close to 0, it suggests that the label has a diverse roster with a relatively equal representation of different labels. On the other hand, if the index approaches 1, it indicates that the label is heavily biased towards a major record label.

This methodology has been employed in earlier research [20] to examine the variety of top-level classes within the MDP and LFM-2b datasets. Analyzing record label biases using the Simpson index can be useful for several reasons. It allows researchers, analysts, or music enthusiasts to:

- **Identify patterns of concentration:** The index helps to determine if a music dataset is dominated by a few influential record labels, potentially shedding light on market dynamics and the distribution of resources within the music industry.
- **Uncover underrepresented labels:** If the Simpson index highlights labels with low representation, it may indicate that these labels face challenges in gaining exposure or have limited resources compared to more prominent labels. This insight can inform efforts to support and promote diversity in the music industry.

In order to analyze the Simpson index distribution among the user profiles, it is necessary to impose a minimum profile length restriction. This is justified by the fact that very short profiles naturally exhibit a higher probability of having low diversity in major labels, which could potentially distort the Simpson index distribution. This decision aligns with the rule applied during the

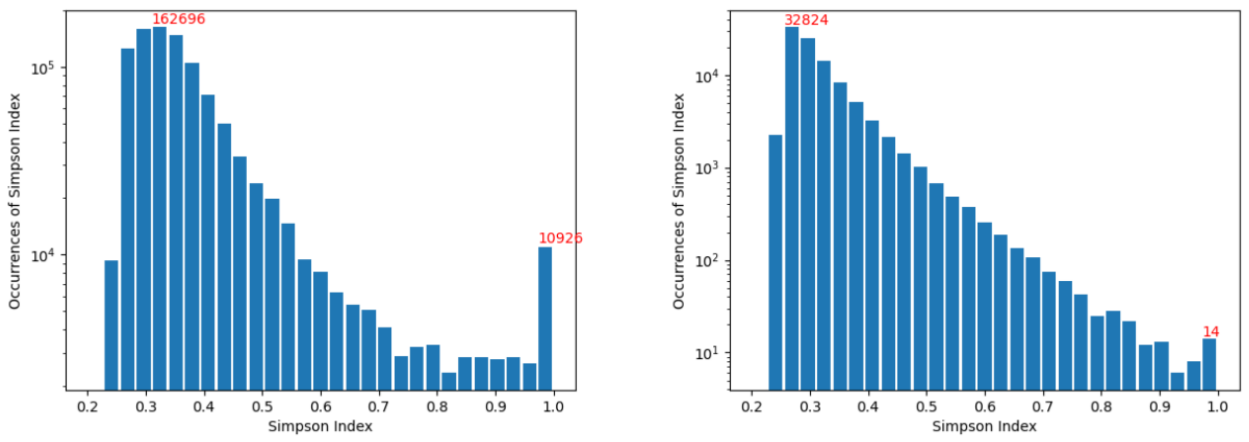


Figure 9: Distribution of Simpson index of major label diversity per playlist on MDP (left) and user profile on LFM-2b (right), obtained from previous research [20].

creation of the MPD, which also included a minimum length requirement. Hence, a minimum length threshold of 10 was selected, based on the datasets' average user listening length, which is double the size of the threshold used for the MPD dataset and smaller than the 30 used for the LFM dataset. However, it still ensures an adequate sample size of users and sufficiently long listening profiles to ensure representativeness. This minimum length threshold will be applied to all subsequent analyses of user profiles.

In the Simpson index formula, which describes how diversity is calculated, we specify that it is typically computed as $\lambda(U) = 1 - \sum_{i=1}^{N} p_i^2$. In our case, we will calculate it as $\lambda(U) = \sum_{i=1}^{N} p_i^2$. This approach aligns with the methodology used in previous studies. This adjustment allows for easier comparisons among different datasets.

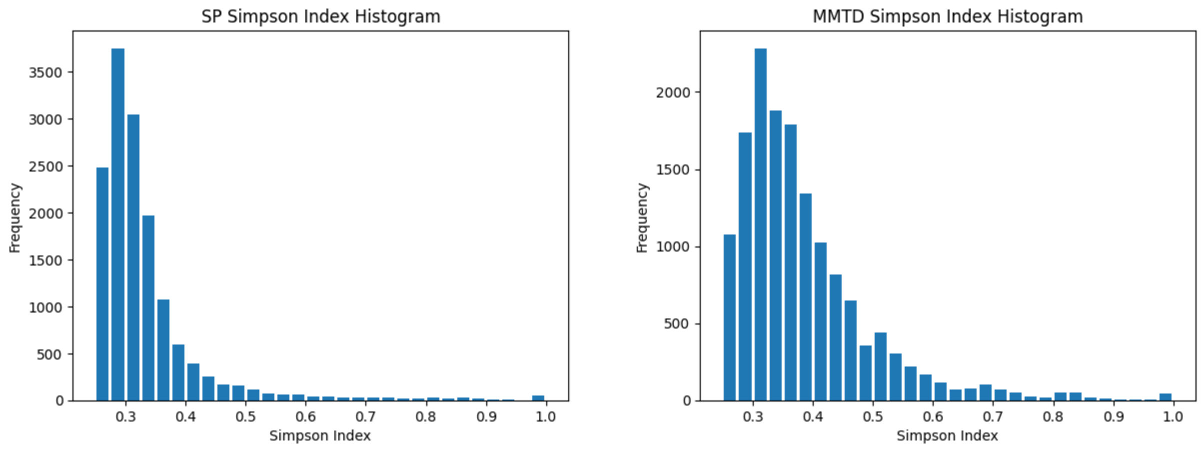


Figure 10: Distribution of Simpson index of major label diversity per user profiles for SP and MMTD datasets.

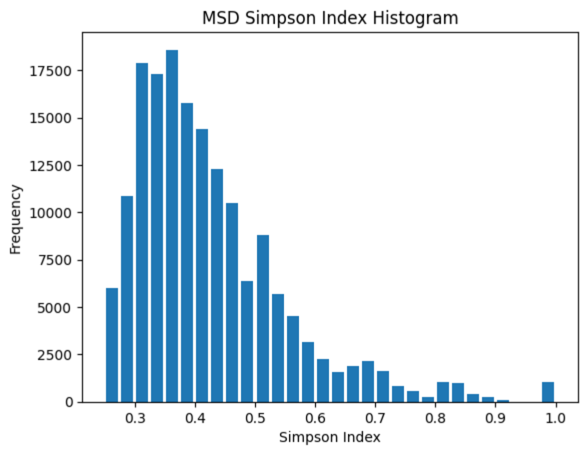


Figure 11: Distribution of Simpson index of major label diversity per user profiles for MSD dataset.

The results of the Simpson index distribution analysis for the MPD and LFM-2b datasets are presented in figure 9. Additionally, figures 10 and 11 showcase the outcomes of this analysis for the new SP, MMTD, and MSD datasets. It is evident that the distribution in the latter datasets appears sparser, likely due to a smaller number of users for whom these indices were obtained.

It is important to mention that in initial experiments using a minimum length threshold of 5 for the listening profiles, an outlier was observed at $SI = 1$ for all three datasets. This outlier indicated the presence of profiles consisting solely of tracks from a specific top-class. However, further analysis revealed that these profiles were of short length and did not carry significant relevance. Hence, when using a minimum length of 10 for the profiles, these outliers at $SI = 1$ were no longer present.

To conclude the investigation on Simpson indexes, we will analyze the significance of the four top-level classes at a high Simpson index value.

In previous research [20], similar findings were obtained for the MPD and LFM-2b datasets. It was discovered that when the Simpson index exceeded 0.8, the record label with the highest presence in MPD playlists was *Universal*, accounting for 64.98% coverage. *Sony*, *Warner*, and *Independent* followed with coverage percentages of 13.70%, 7.81%, and 13.52% respectively. Similarly, in the LFM-2b dataset, user profiles with a Simpson index of 1 exhibited almost complete dominance by the *Independent* record label.

For the new datasets, a table is constructed (table 1) that displays the number of user profiles falling within the range of $SI > 0.8$, along with the average count of each top-level class across all users for each of the new datasets. This allows us to determine which classes exhibit greater dominance when the index values indicate user profiles or playlists that are predominantly associated with a single class.

	N Users	Universal	Sony	Warner	Independent
SP	164	18.6086	17.6800	13.5714	5.2765
MMTD	176	17.4800	3.3913	6.7812	4.7142
MSD	3832	10.9433	3.5028	1.4710	3.7477

Table 1: Summary of the mean count of top-level classes observed averaged over all user listening profiles with a Simpson index greater than 0.8.

8. Feedback Loop Simulation

In section 6, the simulation setup was explained in detail for replicating feedback loops. Some simulations were conducted on an ASUS ZenBook with an i7-10510U CPU running at 2.30 GHz and equipped with 16GB of RAM, operating on Windows 11. Additionally, other simulations were run on a HPC cluster provided by TU Wien, utilizing 25GB of RAM and 4 CPUs. The datasets discussed in section 4 were used for these simulations.

Million Playlist Dataset (MPD)

In the case of the MPD, each playlist was treated as a user, and the tracks from the dataset were considered as items for the user-item matrix.

In the initial experiment, the pool available for recommendation included all playlists in the dataset, allowing for the possibility of tracks appearing multiple times within a playlist. Nevertheless, this experiment had to be terminated due to excessively long execution times, about half a day for each iteration.

To simplify the experiment, a condensed version was conducted by recommending tracks to only 5% of the total one million playlists. Specifically, 50,000 playlists were randomly selected for this purpose. This reduced setup enabled the execution of a complete set of 50 iterations as depicted in figure 12 in over two days with the HPC cluster.

LFM-2b Dataset

In this experiment, the LFM-2b dataset provides a predefined set of users, which is utilized with the condition of excluding tracks that appear less than 15 times. Additionally, no recommendations are made to users who have already interacted with certain tracks in the original dataset.

This dataset encountered a similar situation as the MPD due to its significant size. As it comprises 2 billion interactions, it was necessary to significantly reduce its size to ensure that multiple iterations of the feedback loop simulation could be computed within the required span of time. Consequently, the decision was made to retain 20 million interactions, which corresponds to 1% of the total interactions. This reduction in size allows for more manageable computations while still preserving a representative portion of the dataset.

This reduced setup also enabled the execution of a complete set of 50 iterations as depicted in figure 13. The executions were performed on the ASUS ZenBook since extensive resources were not required. This setup allowed for an execution time of approximately one and a half days.

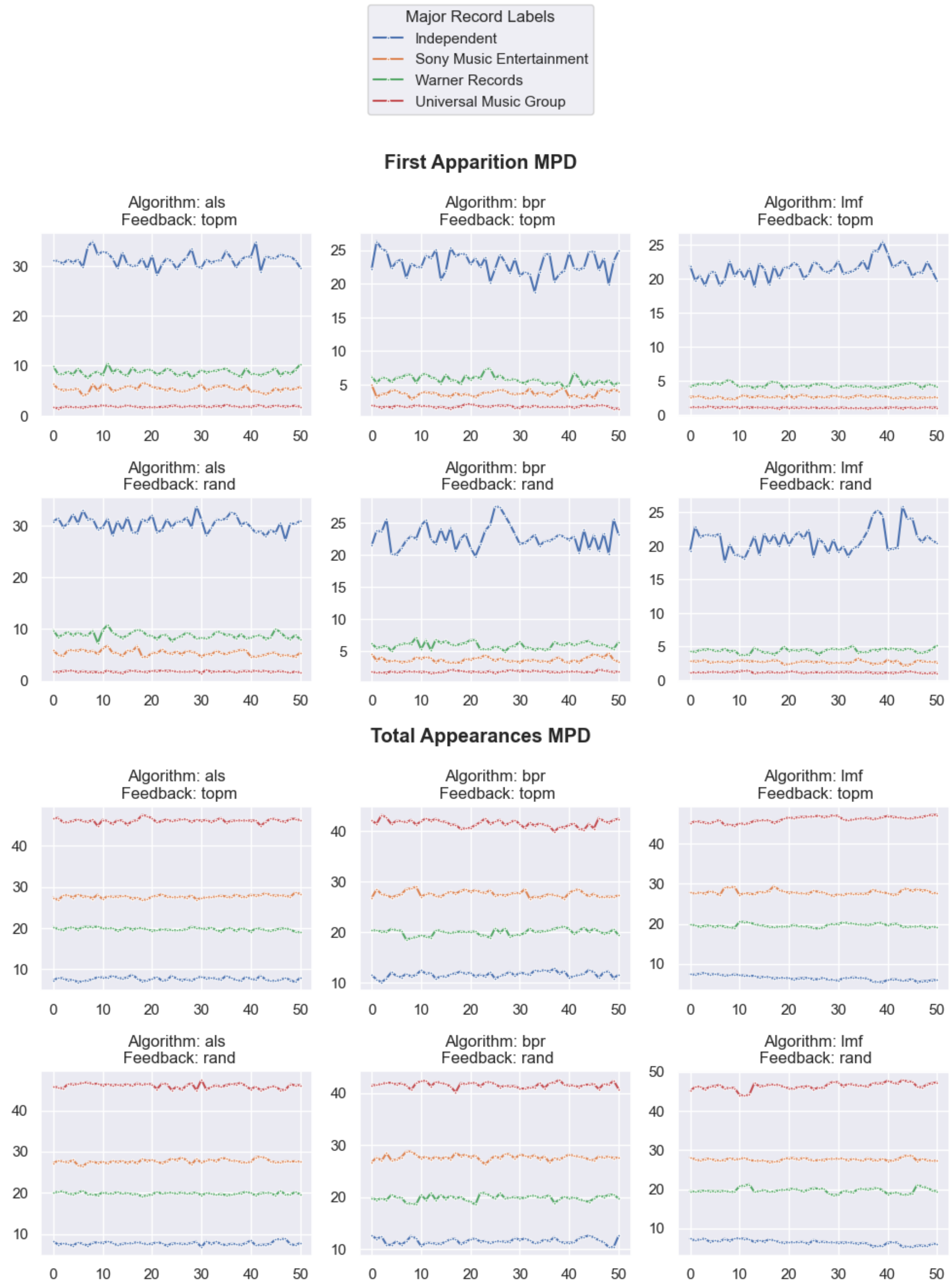


Figure 12: Result of feedback loop simulation on the MPD dataset.

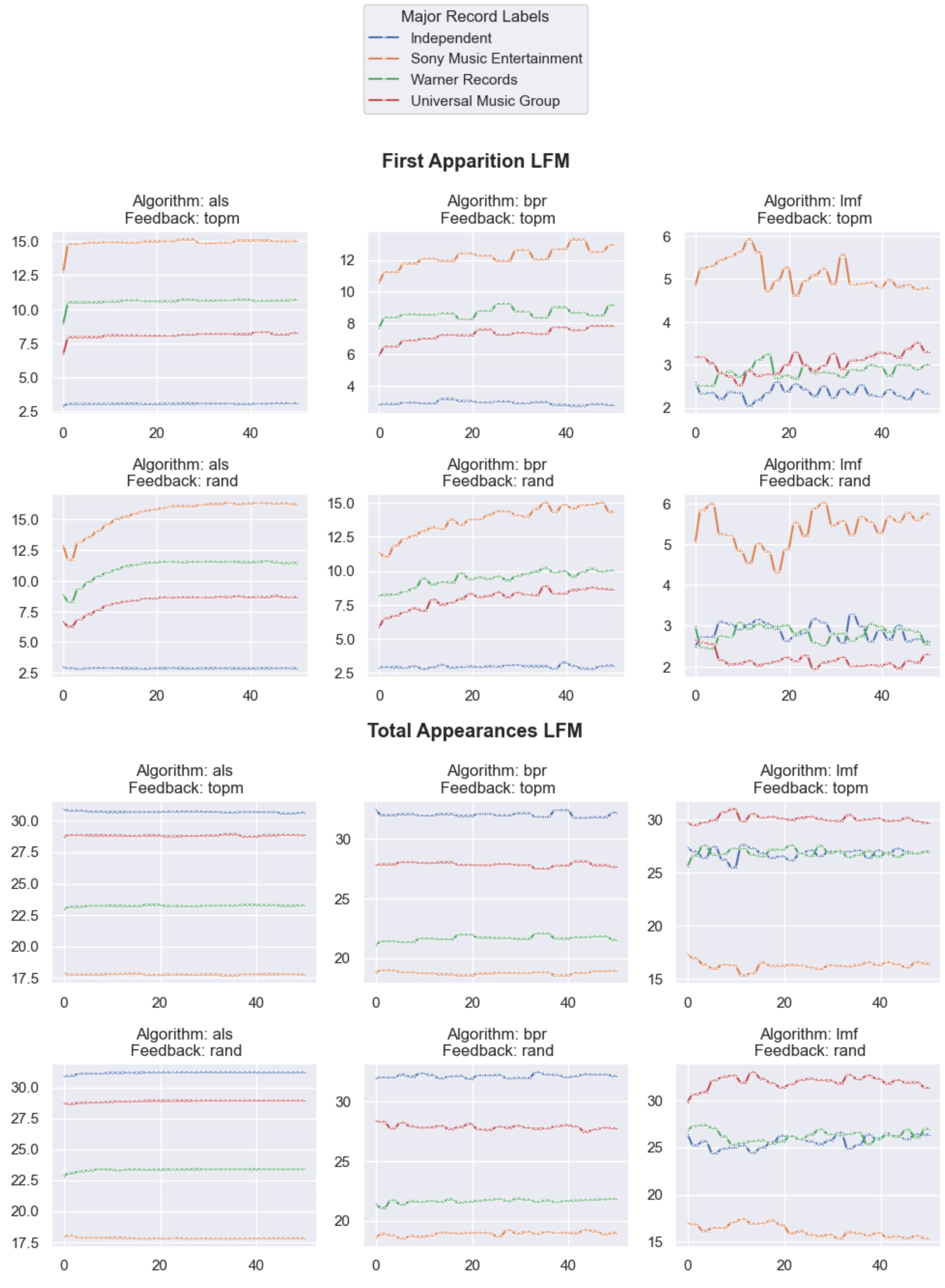


Figure 13: Result of feedback loop simulation on the LFM-2b dataset.

Spotify Playlists Dataset (SP)

For the SP dataset, there also exists a collection of users and the tracks they have included in their playlists, resulting in a total of 8.5 million interactions after the data filtering. This dataset's substantial number of interactions is ideal for conducting simulations, eliminating the need for subsampling as was required with previous datasets.

This setup enabled the execution of a complete set of 50 iterations as depicted in figure 14. The simulations were conducted using the HPC cluster, which enabled runtimes of approximately 4-5 hours per experimental setup, with minor variations. Considering that there were 6 different setups, the overall execution time exceeded one day.

Million Musical Tweet Dataset (MMTD)

With the MMTD dataset, a similar situation occurs where a collection of twitter users and the tracks they have tweeted is available, resulting in a total of 11 million interactions after the data filtering. This dataset's number of interactions is also ideal for conducting simulations, so no subsampling was required.

This setup enabled the execution of a complete set of 50 iterations as depicted in figure 15. The simulations were conducted using the ASUS ZenBook in order to make parallel executions to MSD, which was with the cluster. The runtimes were of approximately 8 hours per experimental setup, with minor variations. The overall execution time was around two days.

Million Song Dataset (MSD)

For the last dataset, the MSD, user listening behaviour is also available, with a total of 6 million interactions after the data filtering. No subsampling was required for this dataset due to its suitable size for execution.

This setup enabled the execution of a complete set of 50 iterations as depicted in figure 16. The simulations were conducted using the HPC cluster because it was noticed that it took too long with the Asus Zenbook. The overall execution time was around a day.

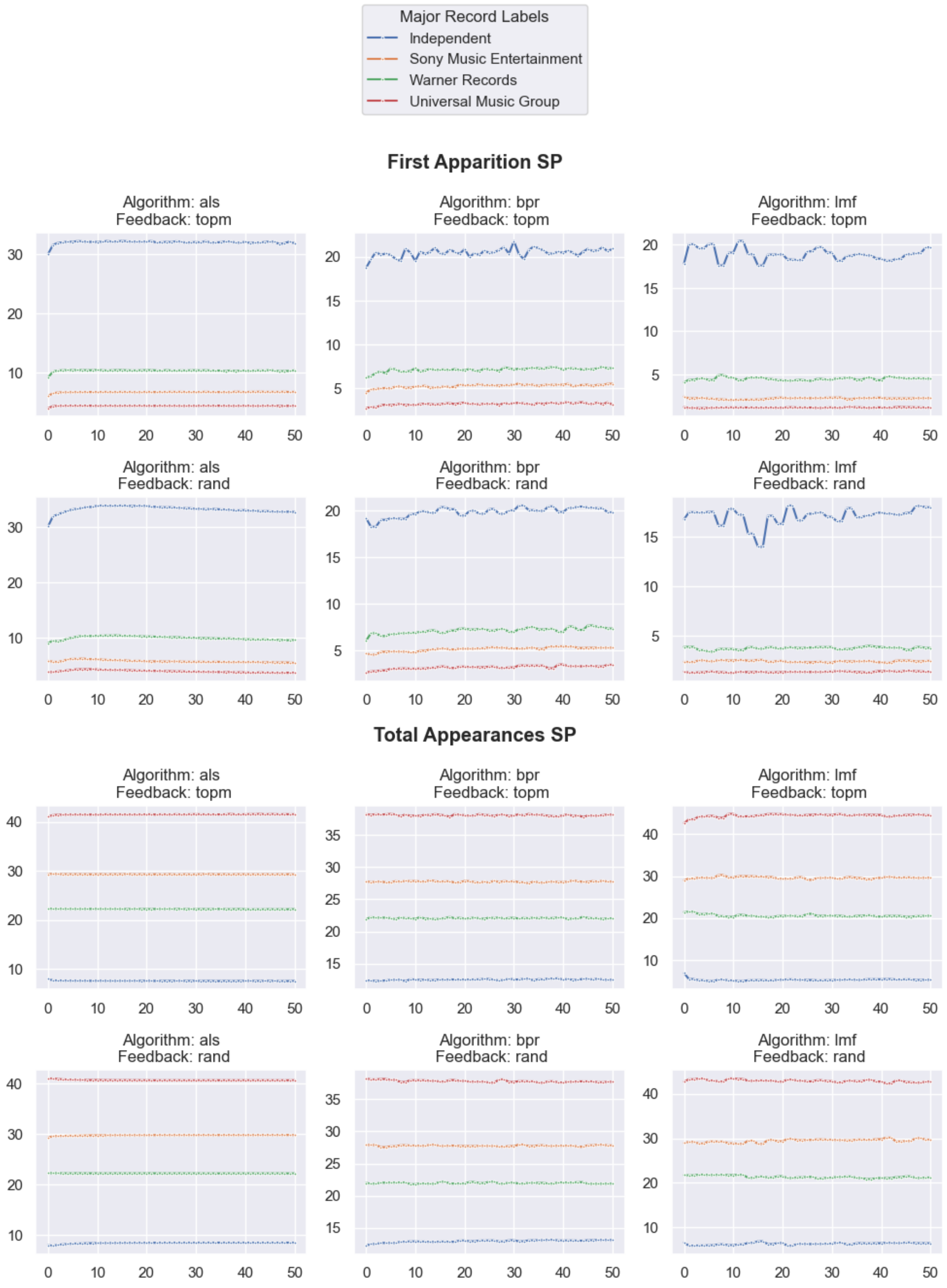


Figure 14: Result of feedback loop simulation on the SP dataset.

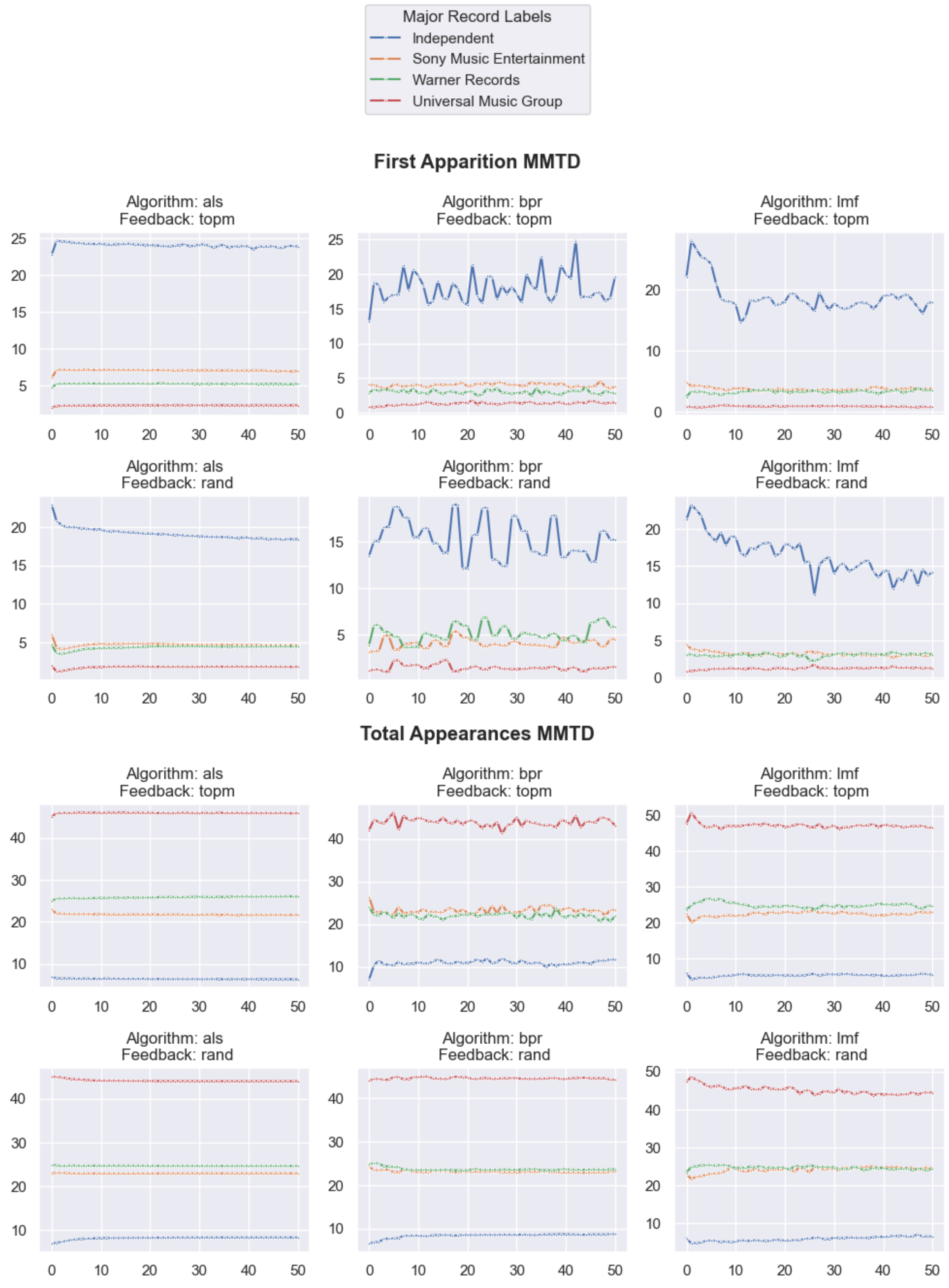


Figure 15: Result of feedback loop simulation on the MMTD dataset.

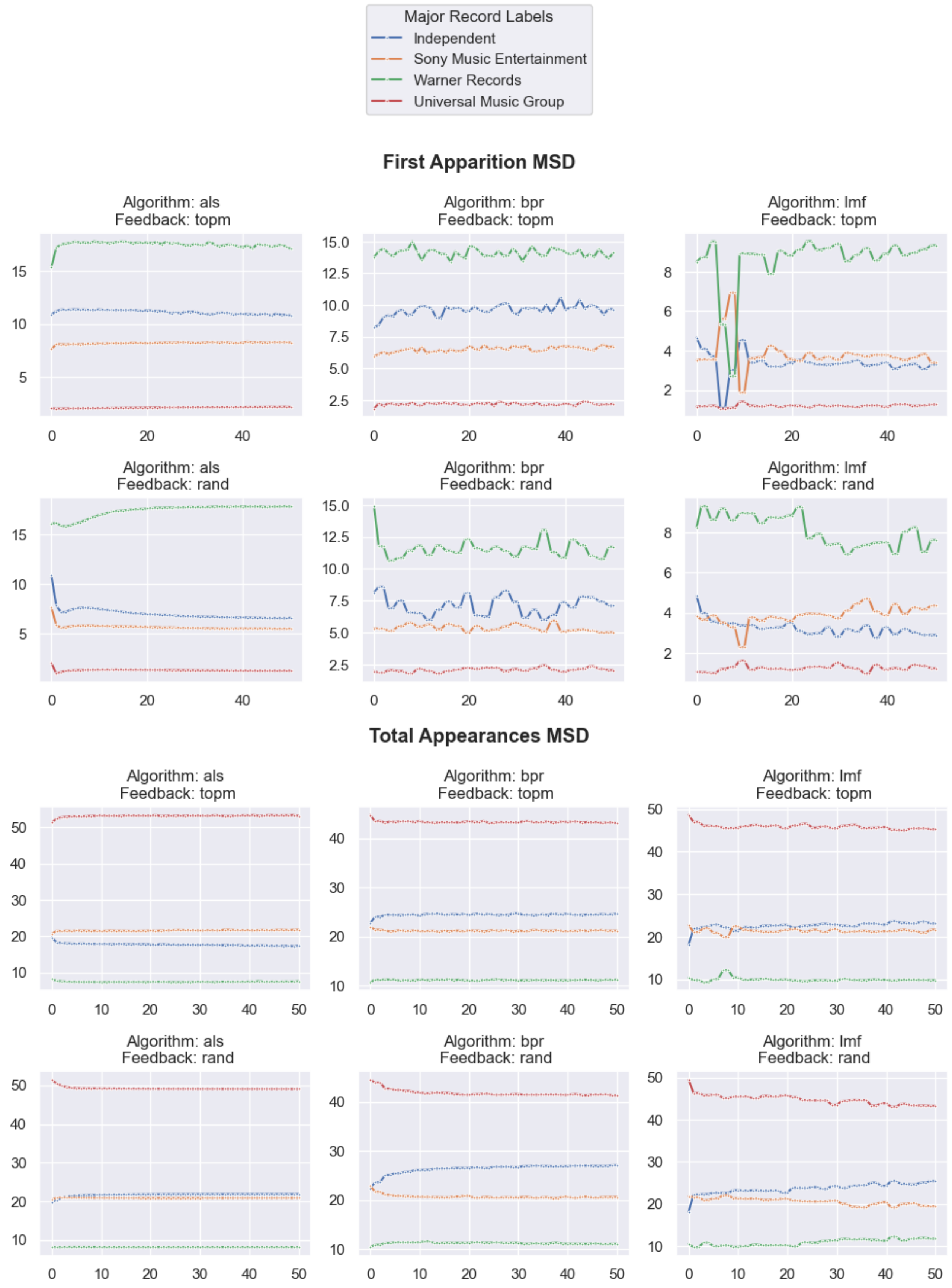


Figure 16: Result of feedback loop simulation on the MSD dataset.

9. Discussion

This section presents the findings of multiple experiments that simulated feedback loops in music recommender systems. Its purpose is to address the research questions and hypotheses of the study, focusing on investigating biases associated with record label distribution and their influence on recommendations. The analysis of these results provides valuable insights into the presence and consequences of biases, contributing to the advancement of fairer music recommender systems.

9.1 Descriptive Analysis

9.1.1 Label Crawler Results

Looking at the development of top-level classes in all datasets, significant differences can be observed in the results (see figures 2, 3 and 4).

For the MPD dataset, notable improvements are seen with the Discogs and Wikipedia stages, regarding assignment of major labels. The Wikipedia crawler achieved higher classification gains of 9.8%, 6.2%, and 6.3% (for Universal, Sony, and Warner, respectively) compared to the Discogs crawler's gains of 8.2%, 4.4%, and 2.8%. The distribution of major labels remains consistent, while the classification rate improves with each additional source.

The LFM-2b dataset starts with a notably higher classification rate of approximately 86%. This is due to leveraging the previously assigned classifications from the MPD dataset, where the top-level classifications for low-level record labels overlap significantly between the two datasets. Interestingly, there is a substantial difference in the distribution between major record labels and independent labels. There is a much higher proportion of independent labels at the beginning (around 34%) compared to the final assignment in the MPD dataset (around 14%). It is possible that this bias was introduced by filtering the LFM-2b tracks to get Spotify URIs.

The outcomes for the SP and MMTD datasets, illustrated in figure 3, also provide insights into the higher number of assignments that can be accomplished during the Discogs and Wikipedia stage. Furthermore, the figure demontrates that the classifications are nearly identical, both the final distribution and step gains for each major label exhibit remarkable similarities between these two datasets. One potential explanation for the similarity between the Spotify Playlists datasets and the Million Musical Tweet Dataset is their common source: posts on the social network Twitter. Since both datasets are obtained from Twitter, it is possible that they share similar characteristics and patterns in terms of the major record labels represented. However, it is important to note that additional research and analysis would be necessary to confirm this hypothesis.

For the MSD dataset (figure 4), it can be seen a similar behavior to what would have been expected from LFM-2b if the results had not been reused. It is evident that each stage of the crawler allows for the collection of record label information, although not as effectively as in the previous datasets. As we reach the final stage of this dataset, the number of unknown assignments is the highest. Ultimately, the dataset concludes with a significant predominance of independent labels, accounting for 43% of the total. Consequently, it becomes the second dataset with the highest proportion of independent labels.

The difference in results between the Discogs and Wikipedia stages of the crawler could be attributed to the fact that Wikipedia has more extensive resources and faster API access. Additionally, the variation may also be due to the different search engines used by the platforms, while Discogs relies on strict string matching, Wikipedia's search engine allows for fuzzy search tokens, increasing the chances of finding matches.

9.2 Bias Analysis

9.2.1 Overview

To obtain a broad perspective of record label distributions and the existing biases, word clouds (figures 5, 6 and 7) and interactive treemaps[14] are utilized. These visual representations help identify patterns and trends within each major record label, allowing for a more comprehensive analysis of the data.

In these five word clouds, it is evident that the three major record labels (*Universal*, *Warner* and *Sony*) consistently appear prominently. However, there are instances where other labels, such as *Rhino*, *Legacy*, and *Columbia*, can have equal or even greater significance. Following these labels, the next prominent entities are major distribution companies like *Atlantic*, *Spain*, *Parlophone*, *DK*, *RCA*, *Capitol* and others. Although there are various distributors of significant importance in each dataset, *Universal* consistently maintains the top position. Furthermore, except for the MPD dataset, the wordclouds do not show any significant relevance of the *Independent* token.

The discrepancy between the most dominant and least dominant record labels is more evident in the LFM and SP datasets. In contrast, the other three datasets demonstrate a presence of record labels at various scales. In previous research [20] it was noted that the discrepancy could be attributed to a lower percentage of low-level record labels being classified as major labels. This was evident in the LFM dataset, where the classification rate was 58%. However, in the case of the SP dataset, a similar situation is observed in the wordcloud despite having a higher classification rate which is approximately 80%. Interestingly, for the MSD dataset, which also has a classification rate of around 58%, this effect is not as easily observable in the wordcloud.

For the detailed analysis conducted using treemaps, record labels with more than 1000 occurrences were included in the plot of all datasets, except for MSD where a threshold of 500 was used

[14]https://colab.research.google.com/github/DaniGmzGnz/TFG$_R$*ecordLabelAnalysis/blob/main/Treemap_analysis.ipynb*

due to the relatively low frequency of record labels in that dataset. By imposing this restriction, the observed dominance of each top-level class may be slightly altered. This is evident in the case of MSD, where the top-level class *Independent* appears to have significantly decreased in representation compared to previous analyses (figure 4), where its importance was more pronounced.

Similar to the findings in MPD and LFM-2b stated in previous research, the dominance of top-level classes followed the order of *Universal*, *Sony*, *Warner*, and *Independent*. With the inclusion of the new datasets, for MMTD and MSD this pattern remains consistent, although for SP the *Independent* class slightly surpasses *Warner* in prominence. As observed in the previous plots, the data consistently shows that *Universal* remains the record label with the highest representation across all the datasets.

9.2.2 Distribution of Top-level classes

The final distribution of record labels, after assigning all the low-level labels to their respective top-level classes, is depicted in figure 8. This figure illustrates some variability among the datasets. While *Universal* remains the most dominant major record label, LFM-2b and MSD exhibit higher peaks of importance for the *Independent* class compared to the peaks observed for *Universal* with the other three datasets. The pattern of *Universal* $>$ *Sony* $>$ *Warner* persists within the major record labels, as previously mentioned. However, when considering the inclusion of the *Independent* class, we observe that, except for MPD, *Independent* holds greater prominence than *Warner* in all datasets.

The distribution of top-level classes exhibits a similar pattern across the SP and MMTD datasets, with *Universal* peaking at approximately 42%, followed by *Sony* at around 22%, Independent at 21%, and *Warner* at 13%. The distribution pattern in these new datasets closely resembles that of MPD, with the only notable difference being a higher representation of independent record labels in the two new datasets.

In the MSD dataset, the distribution of top-level classes shows that *Independent* covers nearly half of the tracks, accounting for 43.83%. Following *Independent*, the major labels *Universal*, *Sony*, and *Warner* have proportions of 27%, 18.6%, and 10.57% respectively. This distribution highlights the distinct dominance of *Independent* record labels in the MSD dataset, aligning with the dataset's previously identified characteristics of being less biased towards popularity and major labels.

9.2.3 Simpson Index Distribution

By analyzing the top-class distribution per playlist in the MPD dataset using the Simpson index, a peak of playlists with an SI ranging from 0.25 to 0.35 (figure 9) is highlighted. This peak indicates a relatively equal distribution of the four top-level classes across most playlists. Beyond that, a gradual decline in diversity it is noticed, with an intriguing outlier peak at 1. This outlier represents playlists where all tracks belong to the same top-level class.

In previous research, certain observations were made regarding this dataset. By focusing on playlists with a Simpson index greater than 0.7, it was found that the dominance of Universal becomes more prominent as playlist diversity decreases. Specifically, Universal covered 72.57% of playlists with an SI of 1, indicating high homogeneity, compared to its overall coverage of 41.1% in the dataset.

A similar pattern is observed in the distribution of Simpson indexes for user profiles in the LFM-2b dataset (figure 9), although there are slight differences in the shape of the tail. The overall distribution shows a more uniform pattern compared to the MPD dataset, with a peak around 0.25 to 0.3 and a nearly linear decay thereafter. Unlike the MPD, the LFM-2b dataset has fewer profiles in the SI 1 category. Previous research have also indicated that these user profiles with a Simpson index of 1 demonstrate absolute dominance of the top-class *Independent*.

The histogram shape for the three new datasets (figures 10 and 11) undergoes a significant change. The characteristic linear decay of the tail observed in the MPD and LFM datasets is no longer present. Instead, the decline of the histograms' tails is more noticeable, with the decline occurring towards the middle of the histogram, specifically in the range of SI values between 0.4 and 0.6.

The SP dataset shows a higher concentration of listening profiles in lower SI ranges, suggesting that after filtering out profiles with fewer events, the dataset exhibits diversity in terms of different top-level classes. The peak of the histogram remains in the range of 0.25 to 0.3. The absence of user profiles in the range of 0.5 and above indicates that this dataset does not clearly exhibit bias towards any specific label.

The histogram distributions for the MMTD and MSD datasets exhibit some differences compared to the SP dataset. In both cases, the decay of the tail starts slightly later and is less pronounced. The peak of the histogram remains in a similar position for MMTD, while for MSD, it shifts to a slightly higher SI range. These histograms demonstrate that the concentration of user profiles in the lower ranks of SI (less than 0.4) is not as dense as observed in the SP dataset. Additionally, there is a relatively low presence of users in the higher ranks of SI. It is worth noting that the histogram scale for the MSD dataset is larger, indicating a larger number of users and greater statistical significance in the analysis.

Based on the information provided in the table 1, several conclusions can be drawn. For the new datasets, the table examines the significance of each top-class among users with a high SI, specifically greater than 0.8. This analysis allows us to observe how the classes behave when the SI value indicates that these user profiles predominantly listen to tracks belonging to a single class. It can be seen how there is a high dominance of Universal in the 3 datasets, although the dominance for SP is less significant.

Moreover, the table indicates that in the case of MSD, where the *Independent* class had been observed as significant (figure 4), these user profiles that exclusively listen to one class do not belong to the *Independent* class. Therefore, the importance of the *Independent* class lies in user profiles with higher diversity on their tracks, rather than profiles focused on a single class.

9.3 Feedback Loop Simulation

Prior to delving into the discussion of the feedback loop simulation results, it is important to acknowledge that these findings are based on the specific setup used and may not necessarily reflect absolute truth. It is important to mention that certain datasets have undergone subsampling, which may result in the loss of a representative portion of the sample. Moreover, due to time limitations, the experiments were not repeated multiple times to validate the absence of significant variations, although such variations are not expected.

Million Playlist Dataset (MPD)

In this initial dataset, when examining both the first appearance and total appearances (figure 12), we can observe that the order in which the top-level classes appear closely resembles the distribution observed in the previous dataset (figure 2). There is a clear dominance of Universal, which has been favored in the recommendations since the first iteration, while the other classes are significantly lower in comparison. There is also some noise or imbalance noticeable in the First Appearance of the *Independent* class, although there is no discernible upward or downward pattern.

In terms of potential feedback loops, no patterns are clearly identified as the iterations progress. However, when examining the total appearances, which indicate the coverage of each class in each iteration, there is a slight upward trend for *Universal* and a downward trend for *Independent* in the LMF algorithm with both user feedback methods. This observation suggests that as users provide feedback, the LMF algorithm tends to recommend fewer songs from the *Independent* category and instead prioritize songs from *Universal*.

LFM-2b dataset

In the case of the LFM-2b dataset (figure 13), we observe distinct behavior. Firstly, when examining the first appearances, it is clear that the *Independent* class consistently emerges as the preferred choice for the top positions. Additionally, in terms of recommendation coverage, the majority of recommendations also belong to the *Independent* class. However, it is worth noting that the coverage of each class across iterations does not directly mirror the distribution observed in the dataset (figure 2).

Examining the total appearances for the ALS algorithm, we can observe that the class coverages are approximately 31% for Independent, 28% for Universal, 23% for Warner, and 18% for Sony. These values deviate significantly from the distribution present in the dataset, which is 53% for Independent, 20% for Universal, 15% for Warner, and 12% for Sony. This observation is even more significant for the LMF algorithm, as we can see that both *Universal* with a 33% and *Warner* going up to 26% have a higher coverage than the *Independent* class itself, despite the *Independent* class being dominant in the dataset. The results indicate that the algorithms tend to overrepresent the three major record labels while representing the *Independent* label to a lesser extent.

Regarding possible feedback loops, some behaviors can be seen. When examining the first appearances of the classes, a clear ascending pattern can be observed for the three major labels in the ALS and BPR algorithms, while the *Independent* class consistently remains the lowest. This observation suggests the existence of a potential feedback loop specifically related to the *Independent* class, where tracks belonging to the three major labels are recommended less frequently in the top positions as the iterations progress, and are instead included in later recommendations. In contrast, the LMF algorithm and random user feedback exhibit an unusual behavior where *Universal* is positioned ahead of *Independent* in the recommendations, deviating from the dominant pattern observed in this dataset where *Independent* had consistently been the top recommendation.

Spotify Playlists Dataset (SP)

For the SP dataset (figure 14), no significant observations are made in this regard. It is apparent that the three major labels consistently receive low positions in terms of first apparition, indicating that they are expected to always appear among the top recommendations. In contrast, the *Independent* class is considerably distant from these three labels.

All the algorithms exhibit a similar pattern, where there are no noticeable changes in the evolution of the plot. The line representing the coverage and first positions remains consistent and uniform for all the classes as the iterations progress. In terms of the percentage of recommendations allocated to each class, we observe that *Universal* maintains a similar proportion of 40% as observed in the dataset distribution (figure 3). However, there is a slight overrepresentation of *Sony* and *Warner*, with their percentages increasing from 23% to 30% and from 15% to 22%, respectively. This increase in *Sony* and *Warner* recommendations comes at the expense of the *Independent* class, which experiences a decrease from approximately 22% to 8%.

No discernible trends are visible for any of the algorithms across the 50 iterations, resulting in a lack of detected feedback loops for the SP dataset.

Million Musical Tweet Dataset (MMTD)

Similarly, abnormal patterns are not observed on a significant scale for the MMTD dataset (figure 15). There are some similarities to the behavior observed in the SP dataset, where the three major labels dominate the top positions in the recommendations, while the *Independent* class remains relatively distant.

The distribution of coverage percentages among different classes reveals that *Universal* receives approximately 46% of the recommendations, while *Warner* and *Sony* receive around 24% each. On the other hand, *Independent* only covers about 6% of the recommendations. This pattern of overrepresentation for major labels, as observed in previous datasets like LFM-2b and SP, indicates that *Warner* is particularly overrepresented in this dataset compared to its distribution (figure 3), where it originally had a coverage of only 12%. As a result, *Independent* class experiences an underrepresentation in the recommendations.

When examining potential feedback loop behaviors, we notice a minor downward trend for the *Independent* class in the ALS algorithm with random feedback. This trend becomes more evident in the LMF algorithm using the same random feedback method. The use of random feedback can potentially contribute to the *Independent* class appearing in higher positions over iterations. By selecting tracks randomly, there is a greater chance for the *Independent* class to be chosen and receive user feedback. Conversely, in the absence of random selection, user feedback would primarily be directed towards the classes that consistently appear at the top positions, namely the three major labels. We can further support the existence of this feedback loop towards the *Independent* class by observing that, with random feedback across all three algorithms, the total appearances of the *Independent* class slightly increase over iterations, while the total appearances of the *Universal* class decrease.

Million Song Dataset (MSD)

The MSD dataset presents intriguing observations (figure 16), particularly in the first appearances of the LMF algorithm with top recommendation feedback. Although some irregularities and unusual behaviors are observed, there are still discernible patterns that allow for certain conclusions to be drawn.

Upon initial examination, it appears that this dataset exhibits the most notable bias towards a major record label. This can be observed in the first appearances plots, where *Universal* consistently occupies the top position by a significant margin although only 28% of the tracks belong to the *Universal* label (figure 4). Furthermore, the total appearance plots clearly demonstrate that *Universal* enjoys a coverage of approximately 45-50%, whereas its actual relevance in the dataset was only 28%, as previously mentioned. This highlights the overrepresentation of the *Universal* label in the recommendations compared to its representation in the dataset.

Lastly, there are indications of potential feedback loops. Observing the BPR algorithm, we notice patterns of growth and decrease of the coverage of classes during the initial iterations, which eventually stabilize around iteration 10. Notably, the most significant observation is the decline of *Universal* and the rise of *Independent*. Furthermore, when examining the LMF algorithm with random feedback, we observe that the aforementioned pattern persists, but unlike the BPR algorithm, it does not stabilize around iteration 10 but continues throughout the remaining iterations. Additionally, we observe a decline in *Sony*'s coverage and an increase in *Warner*'s coverage.

9.4 Research Questions

In this section, we will address the research questions and hypotheses presented in the thesis (section 1.2.3), drawing conclusions based on the results discussed earlier.

Research Question and Hypothesis 1

The first research question posed the following, “What biases related to record labels can be identified in music datasets?”.

In the MPD dataset, a significant prevalence of the three major labels was identified, accounting for around the 84% of the tracks. This indicates a bias towards major labels and a relative underrepresentation of independent record labels. Specifically, there is a notable bias towards *Universal*, which holds the largest overall coverage in the dataset (41%). Moreover, playlists with a high Simpson index tend to exhibit an increased coverage of *Universal*, further emphasizing this bias.

In contrast, the LFM-2b dataset presents a distinct pattern with a strong emphasis on independent record labels. Approximately 53% of the LFM-2b dataset consists of tracks belonging to the *Independent* label, encompassing all user profiles with a Simpson index of 1. This observation indicates a bias towards *Independent* record labels within the LFM-2b dataset.

Similar to the observations in the MPD dataset, the SP and MMTD datasets also exhibit some bias towards a major record label. However, the dominance of all three major labels is not as pronounced in these datasets. *Warner* and *Sony* have relatively less representation, while *Independent* has seen an increase, but *Universal* remains the most represented class by a significant margin. This indicates a clear bias towards *Universal* in both datasets.

In the case of MSD, there is a notable presence of independent labels. However, there is a relatively more balanced representation of the three major labels as compared to the LFM-2b one. Thus, it can be concluded that while there is still a bias towards the *Independent* class, it is smaller in this dataset.

The biases in MPD and LFM-2b are evident, whereas the SP, MMTD and MSD show more balanced distributions among the classes. However, even in the new datasets, there are still biases that resemble those found in the initial datasets. Furthermore, a consistent hierarchy is observed among the three major labels, with *Universal* consistently holding the highest position, followed by *Sony* and finally *Warner*.

The first hypothesis stating “Once record label information is collected and incorporated into music datasets, biases related to certain record labels will become evident”, is supported by the findings. Therefore, the hypothesis can be accepted based on the observed evidence.

Research Question and Hypothesis 2

The second question posed “How do these biases affect the recommendations provided by a recommender system?”.

In the case of the MPD dataset, no discernible bias was detected in the recommender system’s behavior. The coverage of each class aligned with the expected representation across the iterations. As for the LMF algorithm, a potential feedback loop favoring *Universal* was identified, although not to a significant extent.

Regarding the LFM-2b dataset, there is a notable bias towards the three major labels, indicating an overrepresentation of their distribution in the coverage of recommendations. Furthermore, a feedback loop is apparent, favoring the *Independent* class, as the other classes experience a decline in their top positions throughout the iterations.

The SP dataset shows an overrepresentation bias towards the record labels *Sony* and *Warner*, while no feedback loop is observed for any particular class.

For the MMTD, the *Warner* label exhibits an overrepresentation bias, and there is a clear feedback loop favoring the *Independent* class, particularly noticeable in the LMF algorithm.

In the MSD there is an overrepresentation bias towards the *Universal* label, along with two distinct feedback loops. One feedback loop favors the *Independent* class as the iterations progress, while the other adversely affects the *Universal* class.

These simulations provide clear evidence that biases towards record labels can manifest within recommender systems, potentially leading to the emergence of feedback loops over multiple iterations.

The second hypothesis stated “As feedback loops are iterated, biases associated with record labels significantly contribute to a decrease in the diversity of recommendations”. Based on the data, we can reject the hypothesis as there were no significant changes in diversity observed over the iterations. Instead, in cases where feedback loops occurred, one class gained representation while another class experienced a decrease in representation.

Research Question and Hypothesis 3

Based on the findings of sections 9.2 and 9.3, as well as the conclusions drawn from the previous research questions, we will provide an answer to the third research question “To what extent do bias effects persist and generalize across a broader range of research datasets?”.

Based on the analysis of this research project, it can be concluded that the extent to which bias effects persist and generalize across a broader range of research datasets varies. By including more datasets for comparison and verification of existing biases related to record labels, more accurate information can be obtained. The results indicate that biases and their effects continue to manifest in recommender systems, influencing coverage and causing feedback loops. However, these biases do not manifest uniformly across all datasets. The specific characteristics, distributions, and behaviors of each dataset and algorithm contribute to the manifestation of biases and feedback loops towards certain record labels, varying in magnitude and scale.

Therefore the third hypothesis, "The bias effects observed will persist and generalize across a broader range of research datasets", must be rejected. Despite all of them being sampled from real-world listening scenarios, each dataset has exhibited biases in its own unique manner, challenging the expectation of uniform bias effects across all datasets.

10. Conclusion and Future Work

Results and endeavors aimed at expanding the research into investigating the influence and significance of major record labels in music recommendation have been presented. By incorporating two previously analyzed datasets, namely MPD and LFM-2b, and introducing three new datasets; SP, MMTD, and MSD, we have conducted more extensive analyses. Through these analyses, we have been able to demonstrate distinct characteristics and biases regarding the distribution of record labels within the datasets.

The biases towards the three major labels, especially *Universal*, are clearly evident in the MPD dataset. Additionally, the LFM-2b dataset exhibits a bias towards the *Independent* class. In contrast, the SP, MMTD, and MSD datasets demonstrate more balanced distributions among the classes. However, these datasets still exhibit biases, with SP and MMTD showing a bias against *Universal*, and MSD displaying a bias against *Independent*. Furthermore, a consistent hierarchy is observed among the three major labels, with *Universal* occupying the highest position, followed by *Sony* and *Warner*.

Recommender systems have been found to have an impact on record label distribution, with the identification of certain feedback loop effects. Despite the prevalence of independent labels in the LFM-2b and MSD datasets, major labels are over-represented in the recommendation process. In particular, *Universal* exhibits an even greater over-representation. This effect is also observed in the other datasets, even when the three major labels are the ones that dominate them. Moreover, several feedback loops have been found in the simulations, most favoring the *Independent* class but some also favoring the *Universal* label. However, throughout the iterations there was no decline in the diversity of recommendations but some changes in the distribution among the different record labels.

In summary, this research project concludes that the generalization of bias effects in recommender systems vary across different research datasets. Including more datasets allows for better understanding and verification of biases related to record labels. The results indicate that biases and their effects are present in recommender systems, impacting the coverage of certain record labels and leading to the formation of feedback loops that favor certain classes over iterations. However, these biases are not consistent across all datasets, as they are influenced by the distribution of record labels within each dataset and the specific recommendation algorithm employed.

A potential future research direction could be to investigate the effectiveness of integrating user feedback and preference customization as a means to mitigate biases in music recommender systems. This research could focus on evaluating the impact of personalized recommendations on recommendation quality, user satisfaction, and music consumption patterns. Additionally, it could explore how biases in music datasets and recommender systems influence the diversity of music consumption and whether integrating user feedback can lead to a more balanced distribution of record labels. The findings would provide valuable insights for enhancing recommendation algorithms and improving the overall user experience in music recommender systems.

Additional research could explore the impact of changing record label distribution on user satisfaction in music recommender systems. This research could examine whether altering the distribution of recommended music from different record labels affects users' satisfaction with the system. By conducting user studies and collecting feedback, you can assess how users perceive and respond to recommendations from various record labels. This research would discuss strategies for enhancing recommendation algorithms to better align with users' preferences.

References

[1] Ifpi global music report: Global recorded music revenues grew 18.5% in 2021, march 2022. retrieved july 15 2022 from https://www.ifpi.org/ifpi-global-music-report-global-recorded-musicrevenues-grew-18-5-in-2021/.

[2] Luis Aguiar and Joel Waldfogel. Platforms, power, and promotion: Evidence from spotify playlists*. *The Journal of Industrial Economics*, 69(3):653–691, 2021. https://doi.org/10.1111/joie.12263.

[3] Christine Bauer. Report on the ismir 2020 special session: How do we help artists? *Vol. 54*, 12 2020. http://sigir.org/wp-content/uploads/2020/12/p15.pdf.

[4] Christine Bauer and Markus Schedl. Global and country-specific mainstreaminess measures: Definitions, analysis, and usage for improving personalized music recommendation systems. *PLOS ONE*, 14(6):1–36, 06 2019. https://doi.org/10.1371/journal.pone.0217389.

[5] Thierry Bertin-Mahieux, Daniel Ellis, Brian Whitman, and Paul Lamere. pages 591–596, 01 2011. https://www.researchgate.net/publication/220723656_The_Million_Song_Dataset.

[6] Asia J. Biega, Krishna P. Gummadi, and Gerhard Weikum. Equity of attention: Amortizing individual fairness in rankings. In *The 41st International ACM SIGIR Conference on Research amp; Development in Information Retrieval*, SIGIR '18, page 405–414, New York, NY, USA, 2018. Association for Computing Machinery. https://doi.org/10.1145/3209978.3210063.

[7] Reuben Binns. On the apparent conflict between individual and group fairness. In *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency*, FAT* '20, page 514–524, New York, NY, USA, 2020. Association for Computing Machinery. https://doi.org/10.1145/3351095.3372864.

[8] Brian Brost, Rishabh Mehrotra, and Tristan Jehan. The music streaming sessions dataset. In *The World Wide Web Conference*, WWW '19, page 2594–2600, New York, NY, USA, 2019. Association for Computing Machinery. https://doi.org/10.1145/3308558.3313641.

[9] Celma and Oscar. *Music Recommendation and Discovery: The Long Tail, Long Fail, and Long Play in the Digital Music Space*. 01 2010. https://link.springer.com/book/10.1007/978-3-642-13287-2.

[10] D. Bondy Valdovinos Kaye Zhongwei Li David Hesmondhalgh, Raquel Campos Valverde. The impact of algorithmically driven recommendation systems on music consumption and production - a literature review. 02 2023. https://www.gov.uk/government/publications/research-into-the-impact-of-streaming-services-algorithms-on-music-consumption.

[11] Andres Ferraro, Dmitry Bogdanov, Xavier Serra, and Jason Yoon. Artist and style exposure bias in collaborative filtering based music recommendations, 2019. https://arxiv.org/abs/1911.04827.

[12] Andres Ferraro, Dietmar Jannach, and Xavier Serra. Exploring longitudinal effects of session-based recommendations. In *Proceedings of the 14th ACM Conference on Recommender Systems*, RecSys '20, page 474–479, New York, NY, USA, 2020. Association for Computing Machinery. https://doi.org/10.1145/3383313.3412213.

[13] Andres Ferraro, Xavier Serra, and Christine Bauer. Break the loop: Gender imbalance in music recommenders. In *Proceedings of the 2021 Conference on Human Information Interaction and Retrieval*, CHIIR '21, page 249–254, New York, NY, USA, 2021. Association for Computing Machinery. https://doi.org/10.1145/3406522.3446033.

[14] Andres Ferraro, Xavier Serra, and Christine Bauer. What is fair? exploring the artists' perspective on the fairness of music streaming platforms. In Carmelo Ardito, Rosa Lanzilotti, Alessio Malizia, Helen Petrie, Antonio Piccinno, Giuseppe Desolda, and Kori Inkpen, editors, *Human-Computer Interaction – INTERACT 2021*, pages 562–584, Cham, 2021. Springer International Publishing. https://link.springer.com/chapter/10.1007/978-3-030-85616-8_33.

[15] Kim B Serra X Bogdanov D Ferraro A, Jeon JH. Artist biases in collaborative filtering for music recommendation. *In: Proceedings of the 37 th International Conference on Machine Learning*, pages 13–18, 07 2020. https://repositori.upf.edu/bitstream/handle/10230/45185/ferraro_ICML2020_arti.pdf?sequence=1&isAllowed=y.

[16] Shubhanshu Gupta. Music data analysis: A state-of-the-art survey. 11 2014. https://arxiv.org/pdf/1411.5014.pdf.

[17] D. Hauger, M. Schedl, A. Košir, and M. Tkalčič. The million musical tweets dataset - what we can learn from microblogs, 11 2013. http://www.cp.jku.at/datasets/MMTD/.

[18] Robin Burke Himan Abdollahpouri and Bamshad Mobasher. Managing popularity bias in recommender systems with personalized re-ranking. 08 2019. https://arxiv.org/pdf/1901.07555.pdf.

[19] Ben Hutchinson and Margaret Mitchell. 50 years of test (un)fairness: Lessons for machine learning. In *Proceedings of the Conference on Fairness, Accountability, and Transparency*, FAT* '19, page 49–58, New York, NY, USA, 2019. Association for Computing Machinery. https://doi.org/10.1145/3287560.3287600.

[20] M. Hübler. Analysing music collection datasets to investigate the impact of record labels on music recommender systems. *[Diploma Thesis, Technische Universität Wien]. reposiTUm.*, 2022. https://doi.org/10.34726/hss.2022.98121.

[21] Oleg Lesota, Alessandro Melchiorre, Navid Rekabsaz, Stefan Brandl, Dominik Kowald, Elisabeth Lex, and Markus Schedl. Analyzing item popularity bias of music recommender

systems: Are different genders equally affected? In *Proceedings of the 15th ACM Conference on Recommender Systems*, RecSys '21, page 601–606, New York, NY, USA, 2021. Association for Computing Machinery. `https://doi.org/10.1145/3460231.3478843`.

[22] Masoud Mansoury, Himan Abdollahpouri, Mykola Pechenizkiy, Bamshad Mobasher, and Robin Burke. Feedback loop and bias amplification in recommender systems. In *Proceedings of the 29th ACM International Conference on Information amp; Knowledge Management*, CIKM '20, page 2145–2148, New York, NY, USA, 2020. Association for Computing Machinery. `https://doi.org/10.1145/3340531.3412152`.

[23] Rishabh Mehrotra, James McInerney, Hugues Bouchard, Mounia Lalmas, and Fernando Diaz. Towards a fair marketplace: Counterfactual evaluation of the trade-off between relevance, fairness amp; satisfaction in recommendation systems. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, CIKM '18, page 2243–2251, New York, NY, USA, 2018. Association for Computing Machinery. `https://doi.org/10.1145/3269206.3272027`.

[24] Alessandro B. Melchiorre, Navid Rekabsaz, Emilia Parada-Cabaleiro, Stefan Brandl, Oleg Lesota, and Markus Schedl. Investigating gender fairness of recommendation algorithms in the music domain. *Information Processing Management*, 58(5):102666, 2021. `https://www.sciencedirect.com/science/article/pii/S0306457321001540`.

[25] Yoon-Joo Park and Alexander Tuzhilin. The long tail of recommender systems and how to leverage it. In *Proceedings of the 2008 ACM Conference on Recommender Systems*, RecSys '08, page 11–18, New York, NY, USA, 2008. Association for Computing Machinery. `https://doi.org/10.1145/1454008.1454012`.

[26] Moritz Hübler Peter Knees, Andres Ferraro. Bias and feedback loops in music recommendation: Studies on record label impact. 2022. `https://ceur-ws.org/Vol-3268/paper6.pdf`.

[27] Robert Prey, Marc Esteve Del Valle, and Leslie Zwerwer. Platform pop: disentangling spotify's intermediary role in the music industry. *Information, Communication & Society*, 25(1):74–92, 2022. `https://doi.org/10.1080/1369118X.2020.1761859`.

[28] Shrikant Saxena and Shweta Jain. Exploring and mitigating gender bias in recommender systems with explicit feedback. 12 2021. `https://arxiv.org/pdf/2112.02530v1.pdf`.

[29] Markus Schedl. Leveraging microblogs for spatiotemporal music information retrieval. In Pavel Serdyukov, Pavel Braslavski, Sergei O. Kuznetsov, Jaap Kamps, Stefan Rüger, Eugene Agichtein, Ilya Segalovich, and Emine Yilmaz, editors, *Advances in Information Retrieval*, pages 796–799, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg. `https://link.springer.com/chapter/10.1007/978-3-642-36973-5_87`.

[30] Markus Schedl, Peter Knees, Brian McFee, and Dmitry Bogdanov. *Music Recommendation Systems: Techniques, Use Cases, and Challenges*, pages 927–971. Springer US, New York, NY, 2022. `https://doi.org/10.1007/978-1-0716-2197-4_24`.

[31] Ruotong Wang, F. Maxwell Harper, and Haiyi Zhu. Factors influencing perceived fairness in algorithmic decision-making: Algorithm outcomes, development procedures, and individual differences. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, CHI '20, page 1–14, New York, NY, USA, 2020. Association for Computing Machinery. `https://doi.org/10.1145/3313831.3376813`.

A. Datasets Description

In this appendix, we provide an overview of the datasets' structure used in this study.

A.1 Million Playlist Dataset (MPD)

The Million Playlist Dataset (MPD) is comprised of a collection of user-generated playlists totaling one million, which were created during the time period spanning January 2010 to October 2017. These playlists were specifically chosen from a much larger pool of available user-generated playlists by satisfying a range of predefined criteria.
For example, the user who created the playlist must be from the United States and at least 13 years old. Additionally, the playlist must contain between 5 and 250 tracks, feature at least 3 distinct artists and 2 unique albums, and have at least one follower aside from the creator.

The dataset is split into 1000 JSON files with 1000 playlists each. Each slice file contains, among others, the following items:

- Info: General information about this slice of the dataset.
- Playlists (array): An array of playlists, where each playlist contains the following items:
  - PID (integer): Unique identifier of the playlist.
  - Name (string): The name the user gave the playlist.
  - Num_tracks (integer): Number of tracks in the playlist, equal to the length of the list.
  - Num_albums (integer): Number of unique albums of the tracks in the playlist.
  - Num_artists (integer): Number of unique artists of the tracks in the playlist.
  - Tracks (array): An array of tracks, where each track holds:
    - Track_name (string): The name of the track.
    - Track_uri (string): URI of the track in the form of 'spotify:track:' + spotify-id where the spotify-id is a base-62 number.
    - Album_name (string): The name of the album of the track.
    - Album_uri (string): URI of the album in the form of 'spotify:album:' + spotify-id.
    - Artist_name (string): Name of the artist of the track.
    - Artist_uri (string): URI of the artist in the form of 'spotify:artist:' + spotify-id.

The MPD contains over 6.6 million tracks, of which 2.2 million are unique. The dataset also includes more than 734,000 unique albums by nearly 300,000 unique artists. On average, a playlist consists of 66 tracks, with a median of 49 tracks per playlist.

A.2 LFM-2b Dataset

The LFM-2b dataset is a new and extensive dataset that contains actual listening events from the Last.fm online music platform. It includes over two billion listening events collected from more than 120,000 users, encompassing approximately 50 million tracks from 5.2 million artists. The data was gathered over a 15-year period, spanning from 2005 to 2020.

This dataset is separated into different files, although for this study it is enough with 'listening-events.tsv' and 'spotify-uris.tsv'. The structure of these files is as follows:

- **listening-events**: Contains a row for each track a user has listened to, and has the following columns:
  - **user_id**: Unique identifier for each user in the dataset.
  - **track_id**: Unique identifier for each track in the dataset.
  - **album_id**: Unique identifier for each album in the dataset.
  - **timestamp**: The date and time the user listened to the track. The format is "%Y-%m-%d %H:%M:%S".
- **spotify-uris**: Contains a row for each track a user has listened to, and has the following columns:
  - **track_id**: Unique identifier for each track in the dataset.
  - **track_uri**: URI of the track in the form of 'spotify:track:'+ spotify-id.

In order to make the LFM-2b dataset more manageable for experimentation, a decision has been made to reduce its size. Specifically, a sample of listening events that occurred during the year 2020 (between January 1st and March 20th) has been selected. Additionally, only events that involve a track with a corresponding Spotify URI in the 'spotify-uris.tsv' file have been retained, thus avoiding the need for preprocessing. As a result, the dataset has been significantly reduced in size, from 2 billion listening events to just 22 million. After the reduction process, the dataset retained 15072 unique users, 718499 unique albums, and 1408416 unique tracks.

A.3 Spotify Playlists Dataset

The Spotify Playlists dataset is derived from a portion of the nowplaying dataset that includes only users who share their nowplaying tweets through Spotify. Essentially, it includes information about users, their playlists, and the specific tracks within those playlists. This particular dataset encompasses approximately 13 million tracks that are linked to playlists made by users.

The dataset is composed of only one file 'spotify_dataset.csv' that has the following structure:

- **spotify_dataset**:

- **user_id**: Unique identifier for each user in the dataset.
- **artistname**: The name of the artist who created the track.
- **trackname**: The title of the track
- **playlistname**: The name the user gave the playlist.

As it has been mentioned, it includes over thirteen million listening events collected from over 150,000 different playlists generated by about 16,000 users. Additionally, encompasses approximately 2 million unique tracks from 300,000 unique artists.

A.4 Million Musical Tweet Dataset (MMTD)

The MMTD dataset consists of inferred listening histories from microblogs. Each listening event is identified by twitter-id and user-id, and is accompanied by detailed annotations. These annotations include temporal information such as date and time, as well as spatial information including longitude, latitude, continent, country, state, and city. Contextual information about the country is also provided. Furthermore, the dataset includes references to artists, tracks, and other music-related platforms such as MusicBrainz, 7digital, and Amazon.

It is possible to download the dataset as a whole or to obtain each component separately. The complete dataset 'mmtd.txt' has the following appearance (Only 5 columns have been highlighted over the total of 291):

- **mmtd**:
  - **tweet_tweetId**: Unique identifier for each tweet in the dataset.
  - **tweet_userId**: Unique identifier for each user in the dataset.
  - **tweet_trackId**: Unique identifier for each track in the dataset.
  - **tweet_datetime**: The date and time the tweet has been published. The format is "%Y-%m-%d %H:%M:%S".
  - **artist_name**: The name of the artist who created the track.
  - **track_title**: The title of the track.

The MMTD consists of Twitter-identified listening events, which amount to over one million, and were generated by 215,000 distinct users. Additionally, the dataset encompasses over 134,000 individual tracks by almost 25,000 unique artists.

A.5 Million Song Dataset (MSD)

The Million Song Dataset is a collection of audio features and metadata for a million popular contemporary music tracks, which is freely accessible. Its objectives are to inspire research on algorithms that can be scaled to commercial sizes, provide a reference dataset for research evaluation, help new researchers get started in the MIR field, and serve as a quicker alternative to creating a large dataset with APIs. The core of the dataset consists of the feature analysis and metadata for one million songs, which are derived from The Echo Nest.

Additionally, the Million Song Dataset includes several complementary datasets that have been contributed by the community, such as cover songs, lyrics, song-level tags and similarity, user data, genre labels, and more. The project began as a collaboration between The Echo Nest and LabROSA and was partially funded by the NSF.

This dataset is separated into different files, although for this study it is enough with 'unique_tracks.txt' and 'train_triplets.txt'. The structure of these files is as follows:

- **unique_tracks**: Contains a row for each one of the million songs, and has the following columns:
  - **track_id**: Unique identifier for each track in the dataset.
  - **song_id**: Unique identifier for each song in the dataset.
  - **artistname**: The name of the artist who created the track.
  - **trackname**: The title of the track.
- **train_triplets**: Contains a row for each song a user has listened to, and has the following columns:
  - **user_id**: Unique identifier for each user in the dataset.
  - **song_id**: Unique identifier for each song in the dataset.
  - **count**: The number of times this user has played this song.

The initial file will serve as our entry point for the crawler and will be used to acquire record label data for the one million songs. The second file, on the other hand, contains information about the songs that each user has listened to, so it will be used to train our recommendation algorithm.

The file that contains user information includes over 48 million listening events generated by about one million users. Additionally, encompasses approximately 384,000 unique tracks that will be related to the information obtained for the million songs.

B. Workplan and methodology

B.1 Tasks and milestones

Code: WP1	**Name: Familiarization**
Objective	The objective of this package is to become familiar with the environment of the project to be carried out.
Tasks	Literature study, Familiarization with existing datasets and resources
Deliverables	None
Start Date: 20/02/2023	**End Date:** 07/03/2023

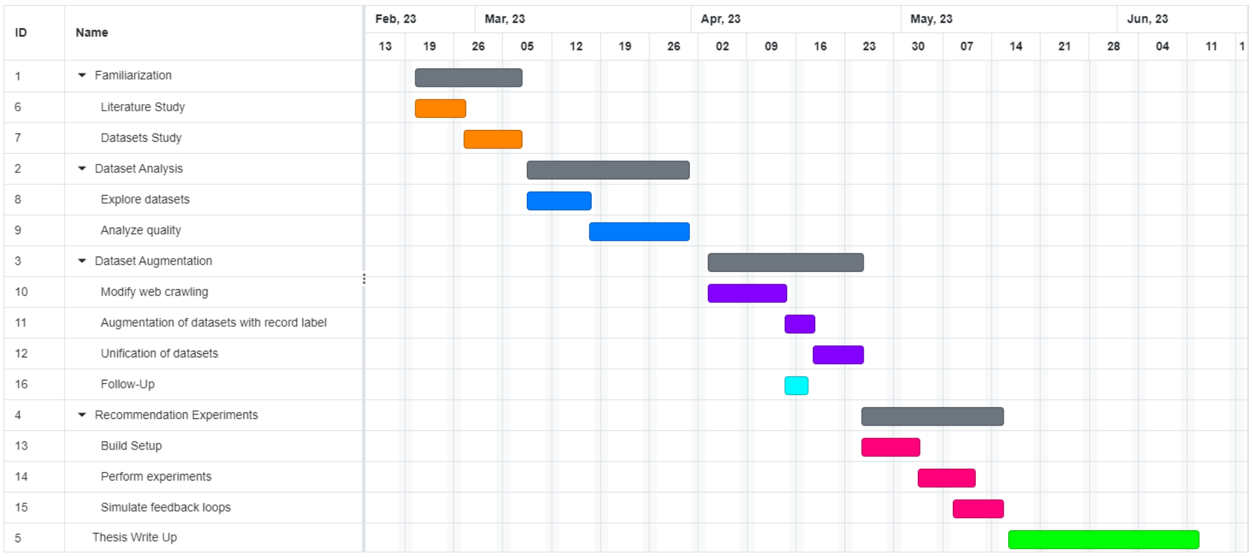
Code: WP2	**Name: Dataset Analysis**
Objective	The objective of this package is to collect, analyze and assess the quality of the different datasets that may be useful to carry out the project.
Tasks	Explore different datasets and resources, Analyze these datasets and it's quality
Deliverables	Code Repository
Start Date: 08/03/2023	**End Date:** 31/03/2023

Code: WP3	**Name: Dataset Augmentation**
Objective	The objective of this package is to perform the dataset augmentation with the record label information through a multi-stage web crawling approach that retrieves this data.
Tasks	Modify the multi-stage web crawling approach, Augment all datasets, Unify these datasets
Deliverables	Code repository, Modular and readable code
Start Date: 01/04/2023	**End Date:** 24/04/2023

Code: WP4	**Name: Recommendation Experiments**
Objective	The objective of this package is to observe how datasets perform in different experiments and feedback loop simulations.
Tasks	Build a setup for experimentation, Perform recommendation experiments, Simulate feedback loops
Deliverables	Code repository, Visualizations, Explanatory Slides
Start Date: 25/04/2023	**End Date:** 14/05/2023

Code: WP5	**Name: Thesis Write Up**
Objective	The objective of this package is to document in detail everything previously done, all modifications, simulations, experiments, etc. In addition to writing the thesis.
Tasks	Documentation, Thesis write up
Deliverables	Code used, Thesis document
Start Date: 15/05/2023	**End Date:** 12/06/2023

B.2 Gantt diagram



ID	Name
1	Familiarization
6	Literature Study
7	Datasets Study
2	Dataset Analysis
8	Explore datasets
9	Analyze quality
3	Dataset Augmentation
10	Modify web crawling
11	Augmentation of datasets with record label
12	Unification of datasets
16	Follow-Up
4	Recommendation Experiments
13	Build Setup
14	Perform experiments
15	Simulate feedback loops
5	Thesis Write Up

B.3 Communication plan

In order to supervise the project, a bi-weekly meeting is scheduled every Tuesday at 14:00 (CET) with Dr. Peter Knees at the TU Wien faculty of informatics. In these meetings, we review the work done and results obtained so far and agree on the next research steps.

Regarding daily basis communication, it is established by email to solve specific doubts or schedule other meetings in addition to those already scheduled.

UPC

TU WIEN