TU WIEN Informatics

Analysing music collection datasets to investigate the impact of record labels on music recommender systems

MASTERARBEIT

zur Erlangung des akademischen Grades

Master of Science

im Rahmen des Studiums

Data Science

eingereicht von

BSc Moritz Hübler
Matrikelnummer 01426077

an der Fakultät für Informatik

der Technischen Universität Wien

Betreuung: Privatdoz. Dipl.-Ing. Dr.techn. Peter Knees

Wien, 1. August 2022

Moritz Hübler   Peter Knees

Moritz Hübler   Peter Knees

Informatics

Analysing music collection datasets to investigate the impact of record labels on music recommender systems

MASTER'S THESIS

submitted in partial fulfillment of the requirements for the degree of

Master of Science

in

Data Science

by

BSc Moritz Hübler
Registration Number 01426077

to the Faculty of Informatics

at the TU Wien

Advisor: Privatdoz. Dipl.-Ing. Dr.techn. Peter Knees

Vienna, 1st August, 2022

[illegible] [illegible]

Moritz Hübler Peter Knees

Erklärung zur Verfassung der Arbeit

BSc Moritz Hübler

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit – einschließlich Tabellen, Karten und Abbildungen –, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

Wien, 1. August 2022

[illegible]

Moritz Hübler

Acknowledgements

A special thanks to Andrés Ferraro for working together on the simulation of feedback loops, making use of his experimental setup on which he ran both enriched datasets.

Kurzfassung

Diese Arbeit untersucht die Dimension von Plattenlabels in Musik Datensätzen für Empfehlungssysteme und deren Einfluss darauf. Um diese Wirkung zu untersuchen, wird zunächst ein mehrstufiger Web-Crawling-Ansatz vorgestellt, der Plattenlabel-Informationen für einzelne Alben sowie eine Zuordnung zu großen Plattenfirmen (Universal, Sony, Warner) oder Independent ermittelt.

Dieser Crawler wird verwendet, um zwei Datensätze anzureichern, nämlich das Spotify Million Playlist Datenset und das LFM-2b Datenset mit Hörprofilen von Last.fm. Anhand der zusätzlichen Informationen können verschiedene Merkmale aufgezeigt und bestimmte Verzerrungen in den nutzergenerierten Musiksammlungen von Playlists und Hörprofilen identifiziert werden.

Darüber hinaus werden Experimente mit Empfehlungssystemen durchgeführt, bei denen Label-Informationen verwendet werden, um Empfehlungen neu zu ordnen und die Leistung von Offline-Empfehlungssystemen zu verbessern. Zusätzlich werden erste Ergebnisse einer Feedbackschleifen-Simulation vorgestellt, in der die Stabilität der Plattenlabel-Verteilung in längeren Empfehlungs-Zyklen untersucht wird. Alle Ergebnisse und die gesammelten Informationen über Plattenlabels werden der Forschungsgemeinschaft öffentlich zugänglich gemacht.

Abstract

This thesis is investigating the dimension of record labels in music recommendation datasets and studying their impact in recommender systems. To study their effect, first, a multi-stage web crawling approach is presented 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 two datasets, namely the Spotify Million Playlist Dataset and the LFM-2b dataset using Last.fm listening profiles. 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.

Furthermore, recommender system experiments are conducted, using label information to re-rank recommendations to improve offline recommender system performance. Additionally, first results of feedback loop simulation are presented, 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.

Contents

CHAPTER 1

Introduction

1.1 Motivation

The age of big data cannot only be interpreted as the age of an increase of machine accessible information, but also as an ever-growing number of available items in online shops or streaming services. Therefore, recommender systems are gaining more and more relevance, as they support users in selecting items from this huge mass of options.

Many recommender systems are data driven, trained on item features or the history of user-item interactions. The quality and quantity of data used for the training is decisive for the performance of such systems. For this purpose, automated information retrieval is a key research field, which includes enriching and augmenting existing datasets with additional information from public sources. This thesis will investigate the gathering of such additional information from publicly available data sources in the domain of music data analysis.

Music recommender systems, as a dominant group of music data analysis processes, have the goal to find the most suited tracks or artist for their user base. Current recommender systems are heavily influenced by the data on which they are trained, and additional information can have a huge impact on their performance [SKM+15, SKMB22]. Furthermore, the question of fairness in recommendations is raised, as current research is shifting the focus from the sole comparison of performance towards the aspect of fairness in recommendations [MMB+18], including fairness regarding shareholders and users [FSB21].

The topic of this thesis is to investigate whether the record label behind a piece of music has an impact on the personalized recommendation result which is presented to the user. The datasets used for these experiments stem from music streaming services, which are the driving force in today music industry [IFP22].

The finding should help in bringing a foundation for a discussion of fairness in music recommender systems. A definition of fairness in this area is beyond the scope of this thesis.

Recommender systems help us to select wanted items from the vast abundance of possible items. They are used by big companies like Netflix, Amazon and Spotify to support users combing through their supply of movies, products and songs. While recommender systems with collaborative filtering sometimes let us discover new sections of items, like a new genre of movies or a new band, they often recommend popular and already approved items. Ferraro et al. [FBSY19] found out that feedback loops can occur, which will lead a system to recommend increasingly similar items. This bias on popularity can have a negative impact on the representation of specific groups in the recommendations like an unfair gender bias, as Ferraro et al. [FSB21] analysed in an earlier study.

It is possible to apply countermeasures to increase the diversity of the recommendations but first the existing biases need to be identified to implement such actions. For data driven approaches of recommender systems this includes a detailed analysis of used datasets to see if unfairness or unwanted imbalances in the data source have an impact on the recommendations.

Previous work [KH19] investigated the distribution of record label companies in the *one million playlist dataset* (MPD), which was used for the ACM RecSys-Challenge 2018[1]. To gather the label information a web crawler [Naj09] was proposed, which focuses solely on Wikipedia as information source. Google was used to get from the album name to its corresponding Wikipedia article, crawling linked record- and distributor companies. This exploratory approach already yielded satisfying results on the MPD dataset with a total coverage of classified tracks of around 96%.

1.2 Aim of this Work

This thesis will improve upon the existing work in two key points with the goal to increase performance and coverage of the crawler. The resulting web-crawler with the enriched datasets and the documented recommender system experiments are technical contributions which will allow future research.

1. Use Spotify-API to get information about publisher companies. Instead of using the name of the album as the starting point for the lookup process, the Spotify-API will be used to get the publisher company. This will remove the first step of mapping the album name to a record label company which is currently realized by simply googling the album name.

2. Increase efficiency of data collection process. The hierarchy between record label companies will be persisted to reduce the amount of repetitive keyword-search

[1]aicrowd.com/challenges/spotify-million-playlist-dataset-challenge, retrieved on 3rd Dec. 2021

and computation the crawler must do. Appending new information to the record label hierarchy with each album will result in a complex network of dependencies between the single companies.

These improvements will allow to address the following three research questions:

RQ1 To which extent is it possible to identify the owner of a piece of music?

RQ2 What kind of biases in music datasets can we identify regarding record labels?

RQ3 Do these biases impact the recommendations of a recommender system?

To answer RQ3, the following two hypotheses will guide the research:

H1 Including the record label information will improve the offline evaluation performance of recommender systems.

H2 Over iterations of recommendations a decrease in diversity of record labels can be observed.

The result of the planned improvements and the research question can be separated in two interdependent parts:

Dataset augmentation

Using the Spotify-API to get information of the publisher companies should result in a broad base for analysing the distribution of publishers before the lookup, making it possible to identify high densities of specific record label companies. While there might not be a publisher company for every URI in a dataset, the number of tracks with unknown record companies should be reduced significantly as the vague mapping between album title and first record company is not needed anymore.

This thesis proposes a multistage process of using the improved web-crawler to gather public information on the publisher companies, which are used to get further insight of the hierarchy of these companies. While the achieved coverage of 92% of the existing web-crawler was satisfiable, the goal this time is to include the long tail of a dataset and get possible insights on the structure of record label companies there.

Finally, applying the suggested improvements to the existing crawler should open new resources: The automatic collection of hierarchical record label information for any given set of publisher companies. The enriched datasets and the crawler are technical contributions, which will be made available to the community.

The descriptive data-analysis of the used datasets and the distribution of record labels after data gathering should answer the first two research questions (RQ1, RQ2).

Insight on impact of record labels in the recommendation process

The enriched datasets will be used in systematic machine learning experiments. This includes analysing effects of record label information on the performance of recommender systems and experiments regarding the simulation of feedback loops and the longitudinal stability of record label distribution in the recommendation process. Therefore, the collected data will be used to answer RQ3 and both hypotheses H1 and H2:

1. **Performance comparison.** Existing approaches, tailored to the Rec-Sys Challenge 2018, will serve as baseline, where *R-precision*, *normalized discounted cumulative gain (NDCG)* and a Spotify's own metric *recommended song clicks* were used for evaluation. [CLSZ18]

2. **Evaluation of feedback loops in recommender systems.** Following the experimental layout of Ferraro et al. [FBSY19] we will be testing the quality of recommendations with a focus on the diversity of record labels, making use of the proposed *Reach* and *Exposure* metrics [FJS20] to measure the reproduction of biases and stability of record label distribution in the recommendation process.

For this thesis a naming convention will be introduced, to distinguish between different levels of granularity of record labels. First: A music entity belongs to a record label or is distributed by a record label. We define three groups of record labels:

- **Low-Level record label**: This is the label or company which is found on the lowest level of the classification process. In other words: this is the first connection of an album or track to a record- or music distribution company.
- **Major record label**: Today's music market is being divided in 3 major record companies: *Universal Music Group*, *Sony Music Entertainment* and *Warner Records*. These major labels from a subset of the **top-level classes**.
- **Top-level class**: This is the highest classified record company and the end of the classification process. This class consists of the **major record labels** and an additional class *Independent*, which summarizes all other small labels.

1.3 The Structure of this Work

This work is primarily focused on data scientists, but an additional objective is to explain the key concepts understandable for music industry scholars, as the thesis combines these two industry fields.

The development of the data gathering process is based on previous work, where many leads are already outlined, summarized in chapter 2. Chapter 3 explains the methodology for this profoundly to mark the reusable character of the web-crawler. Sources for the code of the crawler, the final label assignments and implementation details can be found in the implementation chapter 4. This is followed by the evaluation of the result of the proposed web-crawler on real world datasets in chapter 5.

The systematic machine learning experiments are described in the methodology chapter 6. This is followed by the second evaluation chapter 7 which presents results on using the enriched datasets in recommender system experiments.

Both evaluation chapters are then discussed together in chapter 8, to answer all posed research questions and hypotheses. The ethical statement 9 discusses all ethical question which were raised during the research.

Finally, the most important findings are wrapped up in the conclusion chapter 10 which includes leads for future research.

CHAPTER 2

Related Work and State of the Art

This chapter argues why fairness in machine learning gains such interest currently and explores definitions of fairness in machine learning. Although, many definitions of fairness exist, a general, context-independent definition is currently unknown and likely illusive. Related work on this question will serve as foundation for discussion. Then some of the current challenges of analysing biases in music recommendation systems are explained and how this field of work can be extended to study the influence of label policy in music recommendations. Finally existing work on automated retrieval of record label information for large scale music datasets will be discussed.

2.1 Fairness Concepts for Machine Learning

The topic of fairness and analysis of biases in the field of machine learning generates an increased level of awareness, as these systems are evolving to play a more and more important role in our everyday life [CR18, Nob18]. But defining fairness in the context of artificial intelligence is not a trivial question, as *"Fairness is often context-dependent, and many digital systems need fairness notions that go beyond nondiscrimination."*[LGHT+20].

Corbett et. al [CG18] summarizes three classes of formal fairness definitions, developing from discussions in the research community on fairness in the context of risk assessment algorithms:

1. *Anti-classification*: Algorithms should not consider characteristics for estimations which are defined as protected (e.g., race or gender).
2. *Classification parity*: Shared measures of predictive performance must be equal across all groups: *"Under this definition, a risk assessment algorithm that predicts loan default might, for example, be required to produce similar false negative rates for white and black applicants."*[CG18]

3. *Calibration*: The resulting outcomes of the algorithm must be *"(...) independent of protected attributes after controlling for estimated risk"*[CG18]. This means that a overall possible result of e.g., 10% loan default must be distributed evenly over all groups and this has to be ensured by applying calibration.

While these formal definitions have strong statistical limitations, they manifest an intuitive insight on what characteristics we must take into account when discussion fairness in machine learning.

One reoccurring thought, running through nearly all fairness definitions like a unifying thread, is that all design decision, made to converge the goal of fairness, must be evaluated in the context of their application. *"This means that the development of theory must depend not only on the internal constraints of the science but also upon its external constraints."* [KO76]

Applying this to music recommender systems, two additional fairness categories can be introduced concerning the shareholders and user base:

- *Participation fairness for shareholders*: Fairness towards the label- and distributor companies as well as the artists concerning fair promotion and distribution of their music.
- *Fairness for users*: Am I - as a user - getting the best recommendations because of my profile (e.g., listening profile) or are there other influences affecting this and how transparent are these communicated to me?

In answering the proposed research questions and hypotheses, we want to address these fairness categories in particular, including ethical questions which derive from this theme complex.

2.2 Popularity Bias in Recommendations

While recommender systems seem to have a less severe impact then said risk assessment tools, their influence on our daily lives in societal and economic categories cannot be underestimated. Therefore, research on fairness in the context of automated recommendations is attracting more and more attention [EBD19, KSL19].

There is a general problem of class imbalance in machine learning, typically leading to unfair classification, with the famous example of racial bias of face recognition software trained on mainly white people [GF16]. The corresponding challenge for recommendation systems is the one of popularity bias vs. "long tail", where already popular items are rated more frequently and are therefore promoted stronger by the system than other items. We can therefore say that recommender systems are biased towards popular items [AM20]. Furthermore, studies have shown that this popularity bias is reflected in users' consumption patterns, increasing the attention on more mainstream items over multiple recommendation iterations [MAP+20].

Music Recommendations

Let's narrow down our field of view to music recommendations, where multiple studies exist on popularity bias [MMB+18, FBSY19]. Connecting these insights on the previously quoted fairness definition classes of *classification parity* and *participation fairness for shareholders*, leads to discussing biases concerning features of music creators - artists - like location, period, and gender and how these are reflected in the recommendations of a system. [SPGC20]

In [FJK+20] Ferraro et. al showed that collaborative filtering systems tend to improve their accuracy by recommending popular items and therefore reinforce their popularity, creating so called feedback loops over multiple iterations of a recommender system. To simulate such feedback loops, the assumption is made, that users accept the recommended items fully. Feedback loops result in less diverse recommendations in the long run, when taking artists' styles and genres into account. While Steck [Ste11] showed in a user study on movie data, that users like only a small, if any, bias to reinforce the recommendation of less popular items, we cannot automatically conclude that a general popularity bias satisfies the possible diverse interests of different user groups.

An additional dimension is the perspective of the artists as stakeholders regarding fairness and representation in recommendations [DB22]. Ferraro et. al [FSB21] conducted interviews with a diverse group of artists on the perceived representation in recommendations, arguing that recommender systems should represent and serve item providers in a fair way. The study suggested simple re-ranking strategies, which penalize repetitive and unfair recommendations. While different counter measures to popularity biases and feedback loops exist [FJS20, PT08], first existing biases must be identified.

Record Label Bias

In this work we will zoom in on the aspect of ownership of music, more precisely the distribution of record label companies in music recommendations. For this we will focus - although not exclusively - on Spotify, as *"(...) the leading interactive music streaming platform (...)"*[AW21].

In the IFPI Global Music Report 2022 [IFP22] streaming services make up for 65% of the total music industry revenues. Compared to last year streaming services could register a growth of 24.3%. Paid subscription streaming is said to be the driving force for a growing global music market, with 523 million users worldwide.

But streaming services like Spotify and Apple Music are much more than large music databases. Researchers found playlists to be the format of choice for music consumption on streaming platforms, replacing albums as primary format of music collection [MP15]. Playlists differ inherently from a log of listening events as found on Last.fm where the former represents a personally structured music collection of which usually even the title has a personal and relevant meaning [PZS15, LCK16]. For Spotify there is a special interest on playlist created by Spotify itself, which are heavily promoted, being *"the most visible and most followed playlists on the platform. The 35 most followed playlists on Spotify (as of January 2019) are all Spotify-owned playlists; as are 99 of the top 100 playlists."*[PVZ22] Re-bundling the platforms content in this way, gives Spotify a big leverage on the visibility of artists and label companies.

In [PVZ22] Prey et. al explored the intermediary role of Spotify on the record music industry, using the official @Spotify Twitter account as proxy. The research was based on tweets from 2012 to 2018, where Spotify is promoting the platforms content, above all playlists. The results of the study showed that Spotify is promoting playlists in over 50% of its tweets, which is well above the 30% that Spotify itself stated as percentage of listening time spent in playlists by users [SC18]. *"Spotify thus appears to be actively pushing listeners to consume music through playlists in place of other formats."* [PVZ22]

Additionally, the study investigated what percentage of tracks in Spotify-owned playlists is owned by major record labels (*Universal Music Group*, *Sony Music Entertainment* and *Warner Music Group*) and what share belongs to independent record labels. The label classification was done solely based on copyright information, not taking distributor companies into account. Using this label assignment, Prey et. al identified 54.1% of all tracks from tweeted Spotify-owned playlists as major label content. This number was compared to a random sample of 1001 Spotify samples from 2018, which included playlists created by users, artists and Spotify. This sample had a significant lower percentage of major label content of 43.93%. *"This seems to indicate that Spotify exhibits a bias towards major label content, relative to what listeners and other third parties are promoting through the playlists they create."*[PVZ22].

This motivates an improved method for record label gathering to get a more precise overview over existing label biases and achieve additional insights by performing recommender system experiments with such enriched data, including the previously described *feedback loop* simulation.

2.3 State of the Art

To the best of our knowledge there is currently no dataset which could function as ground truth for record label assignment of large-scale music datasets. There are multiple information sources worth consideration like copyright information on platforms like Spotify, music related databases or more unstructured sources like Wikipedia. To achieve best results multiple information sources should be combined, implying the challenge of information retrieval from unstructured data sources as proposed in [Ora14].

In [KM11] Knees and Schedl explored the automation of creating grammar rules for information extraction, comparing the performance of supervised learning and rule-based methods to manual rule selection. The result of the study showed that the best performing algorithm (support vector machine) was inferior to manual rule selection, which showed the high grade of complexity of gathering additional information from unstructured sources like social media.

In prior work [KH19] we investigated the distribution of record label companies in a large sized music dataset. A web-crawler was used to gather the record label information, with the input being a list of the names of the albums of the MPD, sorted descending by their occurrences. As information source for the classification solely Wikipedia was used by googling the album names and download relevant Wikipedia pages via DBpedia[1]. A defined set of record label classes was used as target for the classification. For each of these target classes a set of keywords is defined, for which the crawler is counting occurrences in the source text of the relevant Wikipedia pages.

Which pages are thought to be relevant? Starting from the Wikipedia page of the album, the infobox usually contains information of the record label distributing the album, which typically is one or more links to other record labels companies which act as distributors. It is also possible that no further record label company is linked in the infobox of an album or a single distributor. This is usually the case for albums of independent artists or small record labels which define themselves as Independent. For each album the crawler goes through a defined maximum number of tiers of linked record labels. Finally, a heuristic is used to classify each album to the defined set of target record labels based on the gathered information.

This exploratory approach already yielded satisfying results on the MPD dataset with a total coverage of classified tracks of around 96%. The final assignment for *Universal*, *Sony*, *Warner*, *Independent* and *Unknown* was 34%, 20.8%, 18.4%, 18.8% and 7.9% resp.

[1] wiki.dbpedia.org/about, retrieved November 20th 2021

CHAPTER 3

Methodology: Data Gathering

This chapter describes the ideas and concepts of the data augmentation step. Details about the concrete implementation can be found in chapter 4.

Before explaining the steps of enriching a dataset with record label information in more detail, the following is a brief overview of the structure of the classification process. The full crawler process consists of multiple steps, where each step builds on the output of the previous one:

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

2. **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 is a public, user-generated music information platform and marketplace with detailed metadata.[1] We harvest information to link and classify low-level record labels using the provided API

   c) **Wikipedia label crawler**: Similar to the previous step, we harvest label information from Wikipedia[2]. In comparison to Discogs, Wikipedia provides information in a less structured way. Therefore, we resort to infoboxes of pages on artists and record labels, in particular the items *parent company*, *distributors*, and *labels*.

[1] https://www.discogs.com/my
[2] https://en.wikipedia.org/wiki/

d) **Interim label mapping**: Evaluation and incorporation of the additional collected information from the previous crawler steps. Beyond mere similarity matching as performed in the previous steps, this involves traversing the label hierarchies extracted 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.

3. **Postprocessing**: Use the generated classification map from low-level record label to top-level class to enrich original track URI or album URI list from the preprocessing step.

The motivation for the defined order of crawling steps stems from previous research work on crawling record labels [KH19] and extending this research with additional crawling steps. The general trend of the steps performed can be characterised as progressing from "more precise but slow" to "less certain but fast". This is motivated by the descending order of the low-level record labels by their occurrences in the dataset, marking their importance and weight, resulting in a compromise in precision and runtime, having the most precise classification for the most important record labels. Discogs for example makes a more precise lookup possible than Wikipedia as the search can be restricted to labels and the entries are organized in a very clear fashion, having a defined parent label field. At the same time Discogs has a very strict search engine where the search string has to match perfectly and no fuzzy matches are allowed. Additionally, the Discogs API is much slower than the one from Wikipedia, so it makes sense to try to classify the top albums with the Discogs crawler first before running the Wikipedia crawler on them.

The classifications that happen in each step are persisted separately so that a statistical evaluation of the classification gain of each step is possible. This includes flags to differentiate between unsuccessful classifications, like reaching the defined maximum depth in a crawler step. The evaluation of these flags was part of the process of defining and fine tuning the implementation itself, like changing the maximum depth after a test run.

With the preprocessing creating an empty label map of just the low-level record labels as artifact, each classification step outputs at least a new label map with a file name suffix that relates to the step, accompanied by other artifacts like archives to persist already made lookups for the crawler steps.

The additional top-level class *Unknown* is used as interim classification category, before classifying all as such classified low-level record labels as *Independent* in the final label mapping step. This stems from an actual difference in the beginning of the crawler process where for some album URIs no record label information is available on Spotify (empty field or *N/A*, *NaN* values), while others are marked decidedly as *Independent*. We decided to keep this difference during the crawling process as this information might be useful for further research. It is also used in the analysis of the stepwise classification gain of the crawler to keep the process as transparent as possible. The decision to merge *Unknown* and *Independent* in the final step, has already been motivated by arguing that a lack of significant information during all previous crawler steps suggests that missing information itself might categorize an album or low-level record label as *Independent*.

General Assumptions

The augmentation process is based on two assumptions:

1. **A1**: All tracks on an album share the same record label and there is no difference of ownership between single tracks. This means that all assumptions concerning the record label of a track of an album can be applied to all other tracks of this album, making the classification for whole albums feasible.

2. **A2**: Major record labels have an interest to mark the music which they distribute or own with the appropriate rights, so the copyright information on Spotify is assumed to be true. Furthermore, this implies that missing copyright information on Spotify can be associated with an independent record label.

3.1 Preprocessing

To gather the low-level record labels for the dataset the Spotify API is used, crawling through a list of album URIs which serves as input for this step. The album URIs are sorted descending by their occurrences in the dataset. It is also possible to start from a list of track URIs, for which an additional preprocessing step is being introduced, gathering first the full track information from Spotify which includes the album URI. The album URIs are sorted descending by their occurrences on track level and are used as input for the previously described step.

Gather label and copyright information

The Spotify crawler is iterating through the list of album URIs and requests the full album information from the Spotify API. From the available information of each album the following entries are persisted:

- *Label*: The stored label from Spotify, which we will refer to as low-level record label.
- *Copyrights*: The available copyright information which appears in two different types P and C where the P-type is a copyright specific to music.

The copyright information is being preprocessed for further usage by removing the copyright symbols ℗ and ©.

Postprocessing

The resulting list of low-level record labels of the dataset can be reduced by removing duplicate record labels and sorting them descending by the sum of occurrences. The resulting list is the base for the multi-step classification process and will be referred to as label map from this point on, as it is a mapping of low-level record labels to their classified major labels.

It is possible to reuse an existing label map e.g., a previous crawler run on a different dataset. If so, the newly created label map is filled with all existing classifications where the low-level record label name matches.

3.2 Multi-step Label Crawler

3.2.1 Mapping trivial cases

This is the first step of the actual classification process. Some of the low-level record labels in the label map can be classified straight away to major labels as their names are identical or similar to the major labels. A dictionary of alias for each major class is being used for this purpose where each major has a list of aliases assigned.

This step was introduced to prevent unnecessary lookups and to add an analytic category in the final evaluation which represents all classification which are possible on first glance, a category that seemed to be interesting in the analysis of the distribution of prominent low-level record labels in the dataset.

3.2.2 Discogs crawler

In this step a web-crawler is extracting information from the public, user-generated music database Discogs.

Discogs describes itself as *"(...) an online database and marketplace of music releases that helps any enthusiast — from casual fan to professional collector — savor music in a more personal and meaningful way."*[3] Since 2000 it gained over 602,000 users which

[3] discogs.com/company, retrieved on 1st April 2022

contributed to build a very detailed music database which contains entries of more than 15 million releases and 1.7 million labels.[4]

Discogs uses a combination of type and id to identify its database entries. There are types for: labels, artists, releases and master releases. This is very useful for gathering label information as the search can be restricted to include only labels.

The typical Discogs entry of a label contains among other things:

- **Profile**: A short description of the history and ownership of the record label, possibly including links to other entries.
- **Parent Label**: a single link to the Discogs entry of the label owning the current label. Note: While there are many record labels without parent label on Discogs, no label entry with multiple parent labels was found during the research on the used datasets.
- **Sublabels**: The opposite of parent label: A list of links to Discogs entries of record labels owned by the current label.
- **Sites**: A list of links of websites connected to the current label. Common types of websites are Wikipedia, Facebook, Twitter, YouTube and SoundCloud.

Crawler concept

The crawler is iterating through the label map and starts a lookup process for each entry which has neither a final major label classification nor a flag marking an unsuccessful Discogs lookup.

A label search is started on Discogs with the name of the low-level record label. Only the first result is used for the following lookup and if the search result is empty a flag is set for this entry in the label map and the lookup is finished.

Otherwise, the Discogs page is analysed to see if a parent label entry exists. If so and the parent label matches one of the major labels the classification is returned. If a parent label exists but it does not a match a major label, the lookup process is repeated recursively for the parent label. The result of the lookup on the parent label will also be set as the result for the current label. A defined maximum depth ϕ prevents endless lookups. If no parent label exists a flag is set and the lookup is done. Figure 3.1 is visualizing this decision process.

The described process is responsible for all classifications or set flags to identify failed lookups in the label map but further information is saved for later usage. For each entry in the label map for which a lookup of the crawler happened, the following information is persisted:

[4] ibid.

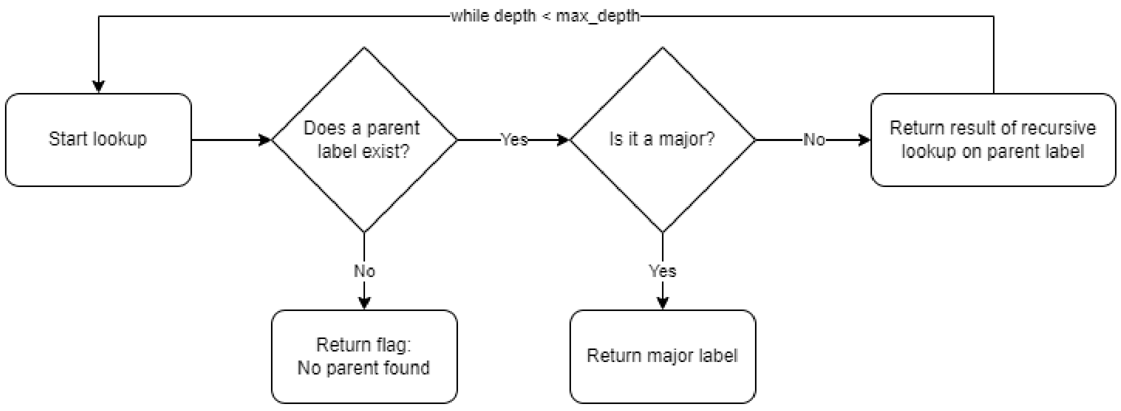

while depth < max_depth
Start lookup
Does a parent label exist?
Yes
Is it a major?
No
Return result of recursive lookup on parent label
No
Return flag: No parent found
Yes
Return major label

Figure 3.1: Flowchart of decision making in the lookup process of the Discogs crawler.

- **Wikipedia-Link**: If during the lookup one of the sites of a label contained a link to the Wikipedia page the topmost Wikipedia link is being saved. This link is being used in the Wikipedia crawler as a shortcut to skip the search on Wikipedia. Picking the topmost Wikipedia link is motivated by preventing the Wikipedia crawler to climb up the label hierarchy again if the link of a label on a lower level would have been chosen.

- **Keyword collection**: For each Discogs page of a label the profile text is being searched for keywords which represent the major labels and the occurrences of these keywords are counted. The collection of all keywords of all levels of the lookup process is being aggregated to result in a single entry in the label map.

Keyword Aggregation

The chosen aggregation function is to summarize the counts for each major label, dividing each count by 2 to the power of d, where d is the level of depth on which the keywords were collected during the Discogs crawler run:

$$\sum_{d=0}^{max\_depth} \frac{count}{2^d} \tag{3.1}$$

This helps to distinguish between keywords which are collected with a different distance to the starting point of the lookup, the first result of the Discogs label search. The chosen formula is motivated by rewarding information which was found "closer" to this starting point.

Output

The input label map of the trivial mapping step is extended with the following information:

- *Classification*: This is either a final classification, a Discogs flag or empty.
- *Wikipedia url*: The last reference to a Wikipedia entry which was found while crawling a low-level record label.
- *Keyword aggregate*: Multiple columns representing the aggregated keywords.

3.2.3 Wikipedia crawler

This step consists of a web-crawler which is using Wikipedia as information source.

A typical Wikipedia article contains information in a much more unstructured way compared to Discogs. For music related articles often infoboxes exist, which portrait distilled information of the article.

From the infoboxes of a music related Wikipedia article the following items are interesting for classifying record labels:

- *Parent company*: A company owning the record label. Sometimes multiple parent companies are listed to show the history of the ownership of the label. This infobox item usually only exists on Wikipedia pages of record labels.
- *Distributors*: A list of label companies which organize the distribution for the record label or music group. This infobox item usually exists for Wikipedia pages of record labels and music groups.
- *Labels*: A list of record labels which own or distribute the music of a group or interpret. This infobox item usually only exists on Wikipedia pages of music groups.

Concept

The Wikipedia crawler is based on the extended output of the previous step, the Discogs crawler. A lookup is started for each low-level record label from the label map which has no successful classification yet.

If a Wikipedia link exists from the Discogs crawling it is used to get to the Wikipedia page of the label. If no such link exists, a search via the Wikipedia-API on the low-level record label name is started. The search is restricted to the English version of Wikipedia because testing lookups manually showed that infoboxes happen to be more likely on the English version. Similar to the Discogs crawler a flag is returned if the search fails or is empty.

The classification process and information extraction of a Wikipedia is structured in a similar fashion as the previous step. The main difference lies in the plurality of possible points of forwarding: Now not only the parent label/company is interesting but the distributors and connected labels as well. All these categories are checked recursively in a defined order until a classification to a major is possible, no further links are found

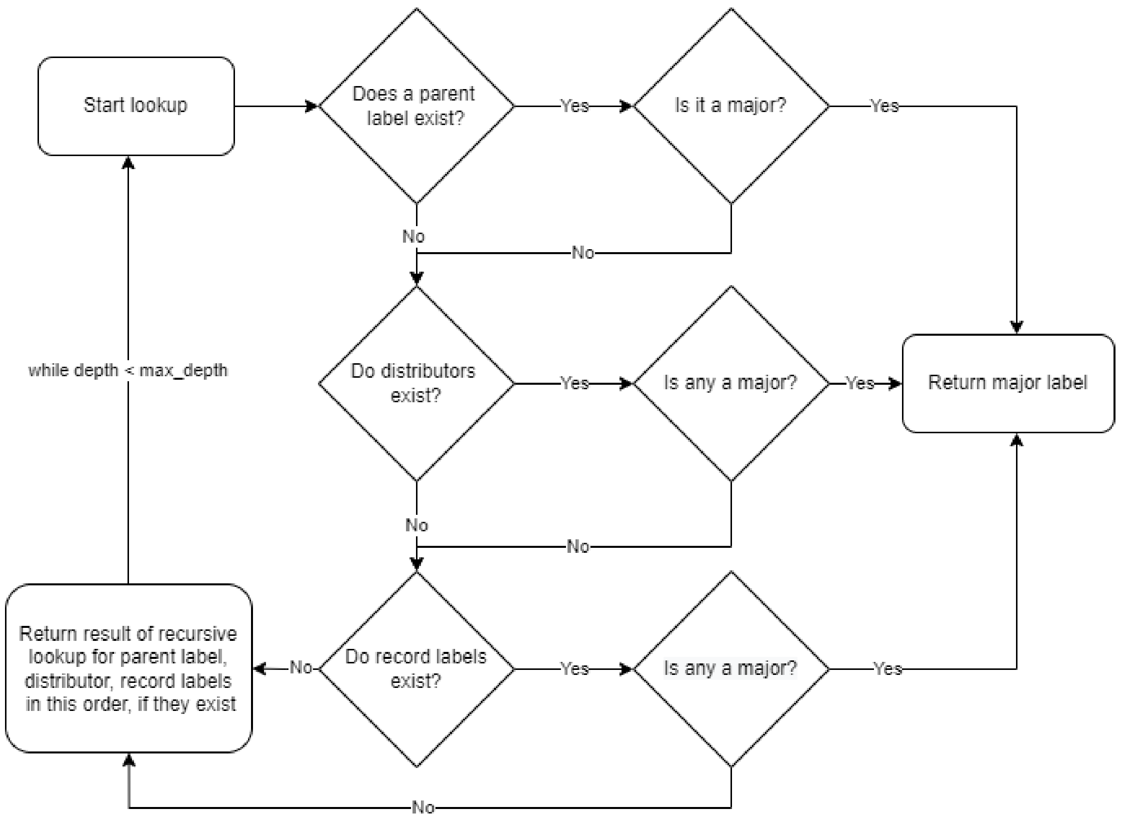


Figure 3.2: Flowchart of decision making in the lookup process of the Wikipedia crawler.

or the defined maximum depth φ is reached. Again, flags help to evaluate the output of the crawler which is helpful for fine-tuning it. A classification happens if any of the forwarding links matches the Wikipedia page of a major label or if the lookup on this page already led to a classification. Figure 3.2 is visualizing this process.

It is noteworthy to say that on each level of the recursive lookup the lookup for each forwarding category (parent company, distributors, labels) is being completed exhaustively before the next category is touched. The motivation for this is that this ranks the forwarding categories, which is reflecting the difference between them when classifying a low-level record label to a major because for this a parent label is more interesting than the distributors and so on.

Like the Discogs crawler, additional information is being extracted during the recursive lookup process. This includes an aggregation of keywords counted in the free text, which is defined as everything outside the infobox of a Wikipedia article. There is also a general Wikipedia article on independent record labels[5] which is linked sometimes in the free text. For each lookup a boolean is saved if such a link appeared or not. Both of this

[5] wiki/Independent_record_label

information is evaluated in later steps.

Output

The input label map of the trivial mapping step is extended with the following information:

- *Classification*: This is either a final classification, a Wikipedia flag or empty.
- *Contains Independent Link*: A boolean if the generic Wikipedia article on independent record labels was quoted in any Wikipedia article during the lookup.
- *Keyword aggregate*: Multiple columns representing the aggregated keywords from the free text, where the keyword collection is being aggregated in the same fashion as for the Discogs crawler keywords (see equation 3.1).

3.2.4 Interim label mapping

This step is evaluating the gathered additional information from the Wikipedia crawler, focusing on if a link to the independent label page of Wikipedia appeared during the crawling on a low-level record label and the aggregated keywords from Wikipedia during the same run.

The mapping is happening in two steps, where the entry is the current low-level record label:

1. An entry is classified as *Independent* if the sum of the Wikipedia keyword aggregate is under a defined threshold α. This is motivated by assuming that if not many keywords were counted during the crawler run for this label, then this label might belong to an independent record label.
2. An entry is classified to the record label which holds the maximum value in the aggregated keyword collection, if the sum of the aggregate is over a defined threshold β. If there is a tie no classification happens. The motivation for this is that with the previous step, the keyword collection already has some significance and if no parent label was found on either Discogs or Wikipedia, the aggregate of the Wikipedia keywords is a valid classification source.

3.2.5 Copyright classification

This step is evaluating the gathered copyright information from the preprocessing step where the Spotify-API was used to get the low-level record label as well as the stored copyright information.

The idea of this step is that analysing the copyright information of the already classified low-level record labels should give an insight on which tokens are the most relevant for each major label and further classification should be deductible.

The copyright information of each album is tokenized and both types of copyright (P and C) are merged, whereby full duplicates are avoided (e.g., if P and C copyright hold exactly the same tokens only one is chosen). For each low-level record label, the copyright tokens of all its albums are aggregated by counting their occurrences and sort them descending.

All copyright tokens are grouped for the major labels and sorted descending, removing trivial filler words with an extended stopword collection. The top tokens should now portray meaningful names of big sub labels of the major labels and after verifying them manually another alias dictionary is created like the first step.

This alias dictionary is used to iterate over all the low-level record labels including the labels which are already classified to a major label. For each label the copyright tokens, sorted descending by occurrences, are analysed. If a token appears in the alias dictionary of the major labels assign the low-level record label to this major label.

The motivation for overwriting already classified entries is that because the tokens in the alias dictionaries are manually checked they hold a great truth value, making it justifiable to overwrite classifications of the crawlers. Another reason is the previously mentioned assumption A2 that the major labels have a high interest to keep the copyright information on Spotify as correct as possible.

3.2.6 Final label mapping

This step is the final touch of the complete classification process. A dictionary of corrections is applied where single low-level record labels can be assigned to a major label. This is useful for special cases like big low-level record labels which are independent or manually researched classifications of bigger record labels which are still not classified.

In this final step also the additional information of the Discogs crawler is being utilized. Like the interim mapping step, the major label with the highest value of the Discogs keyword aggregate is chosen if the sum of the aggregate is over a given threshold γ.

Everything that is not assigned to *Universal Music Group*, *Sony Music Entertainment* or *Warner Records* at this point is set to *Independent*. This is motivated by assuming that all low-level record labels for which no significant information for a classification was found on Discogs, Wikipedia or in their copyright information might just be from an independent label.

3.3 Postprocessing

In the postprocessing, the classification output from low-level record label to major record label is mapped back to the album URIs or track URIs, depending on how the preprocessing was started.

Besides the major label also the low-level record label and the copyright information is being included in the output as this seems to contain valuable information for future recommender system experiments. The additional information can be excluded easily if only the major label is needed.

CHAPTER 4

Implementation

This chapter covers the more technical aspects of the crawler like implementation details, performance, and reusability.

All available sources and the final label classification maps for used datsets as well as a README file with instructions on how to use the crawler can be found in the repository.[1] To explore the interactive graphs from this project, a public google colab notebook is provided.[2]

4.1 URI Definition

The URIs described in this thesis refer to Spotify's own unique resource identifiers (URIs). A Spotify URI consist of a fixed *'spotify:'* prefix, a category from a defined set (e.g., *'track:'*, *'album:'*, *'artist:'*) followed by a base-62 number of length 22 (example: *'0wwPcA6wtMf6HUMpIRdeP7'*). The final Spotify URI (*'spotify:track:0wwPcA6wtMf6HUMpIRdeP7'*) is used for requests to the Spotify-API and is also reflected in the Spotify URL (https://open.spotify.com/track/0wwPcA6wtMf6HUMpIRdeP7) which opens the track, album or artist in a Spotify client.[3]

4.2 Crawler Usage

The detailed usage of the label crawler with all the input paths, credentials, configuration, and parameters is described in the README file of the project.

[1] Github repository: github.com/nostromo7/MT_label_crawler
[2] Google colab notebook: colab.research.google.com/drive/1OKIO9n5BNeNHWl5JF-7REkP5mYo9fVql?usp=sharing
[3] For a detailed explanation of Spotify URIs, IDs and URls, see: spotipy.readthedocs.io/en/master/

The large parts of the project (*preprocessing*, *label_cralwer*, *analysis* and *recsys18_experiments*) are separated in different folders in *src*. The *main* script encapsulates all functionality of the crawler, calling the single scripts from subfolders in the right order with the given parameters. For the used datasets, scripts are provided to generate input lists for the crawler and then to enrich the datasets with the crawler output (*generate_input_list* and *generate_output_list*).

4.3 Crawler Resources

All used packages and dependencies can be found in a conventional *requirements.txt* file. The packages described in the following subsections deserve special attention as they represent the cornerstones of the label crawler.

All packages which need credentials (*Spotipy* and *discogs-client*) use individual credential files, which are filled blank in the repository.

Spotipy

To gather data from the Spotify-API, *Spotipy*[4] is being used. *Spotipy* is a lightweight python library which is also recommended for python environments by Spotify itself. A spotify developer account must be created to use *Spotipy*. On the Spotify developer dashboard[5] an application has to be created, from which the client-id and client-secret are used for authentication with *Spotipy*. For this research project a free spotify account was created, for which the project application was registered.

The Spotify-API is providing the possibility to bundle requests for tracks and albums in bulks to save bandwith. The tested maximum number of URIs in such a bulk was found to be 20. For failed bulk requests the process is repeated with single requests. Additionally, a restriction of a maximum amount requests per day per registered application exists. It was necessary to span the gathering of all information from Spotify over multiple days for this research project.

Discogs client

To access the Discogs-API, the *discogs-client*[6] package was used.

The label crawler relies heavily on the Discogs search engine, which has the characteristics of being very precise, not allowing typos or aliases for label names. While this strict string matching is a minor drawdown - the label name must match perfectly - Discogs offers the possibility to restrict the search on just record labels, which is a very hand feature and missing from other platforms like Wikipedia.

[4] Spotipy docs
[5] Spotify developer dashboard
[6] github.com/joalla/discogs_client

Wikipedia client

As client for the Wikipedia-API *wikipedia*[7] was used, a wrapper of the MediaWiki-API. No credentials are needed for this package.

The *wikipedia* client offers the possibility to set a defined language for which all search results are retrieved. It is suggested to set the language to English ('en') as we found that infoboxes with structured information of music related articles are more likely to be found in the English version of Wikipedia, compared to e.g., the German version.

4.4 Crawler Implementation Details

These are the chosen values for the parameters which act as thresholds for the crawler. Their functioning is described in chapters 3.2.4 and 3.2.6 resp.:

- $\alpha = 2$, *Wikipedia keyword aggregate under*: The defined threshold which the sum of aggregated Wikipedia keywords must be under to classify a low-level record label as *Independent* if a link to the Wikipedia page of independent record labels was found. The chosen value of 2 is motivated by representing a significantly small number of found keywords. It seems reasonable that two keywords found on the first level, four on the second level and so on, should prevent a classification to *Independent* if such a link was found.

- $\beta = 0.25$, *Wikipedia keyword aggregate over*: Threshold which the sum of aggregated Wikipedia keywords must exceed to classify a low-level label to the major label holding the maximum count in the keyword aggregate. The motivation for choosing 0.25 as value for this threshold stems from the number of top-level classes being 4 so the threshold represents a quarter of this number.

- $\gamma = 0.2$, *Discogs keyword aggregate over*: Similar to β but for the Discogs keyword collection in the final classification step. γ was chosen lower than β as less keywords were found on Discogs because of the limited length of label descriptions.

- $\phi = 6$, *Maximum depth Discogs crawler*: The crawler is passing through layers of Discogs articles, looking for links to parent labels. The current depth is persisted, and the lookup ends if the depth exceeds the defined max depth and no assignment was possible before. Both depth thresholds were chosen as a compromise of precision and performance, resulting from multiple test runs with different values (e.g., 4 and 10).

- $\varphi = 6$, *Maximum depth Wikipedia crawler*: Similar to ϕ but for Wikipedia crawler.

[7] Wikipedia client docs

Preprocessing for MPD

The MPD is organized in 1000 slices with 1000 playlists each. a single track can appear multiple times in the MPD and the therefore a reduction of all tracks to their unique album URIs is reasonable compression, assuming that that there are no albums with multiple labels for different tracks. The list of albums is sorted descending by the sum of occurrences of tracks in the MPD including repeated occurrences.

The resulting list of album URIs with their occurrences is the starting point for the label crawler.

Postprocessing MPD

In the postprocessing step the resulting classification of low-level record labels to major labels or *Independent* is mapped back into the original dataset.

To assign the record labels to the individual tracks of the MPD, first the mapping of album URIs to low-level record labels is used which is merged with the final label map. The album URIs then can be used as identifiers for the single tracks of the MPD resulting in a dataset augmented with two levels of record label information: Low-level record label and major classification.

Preprocessing and postprocessing LFM-2b

The LFM-2b Spotify dataset comes in a single *.tsv* file where each track has an *id* identifying it in the LFM-2b dataset and a Spotify *URI*. The process to create the list of track URIs as input for the crawler and to map back the resulting enriched list is not so interesting. It is worth mentioning though, that only 2.1 million URIs of the 2.4 *track ids* are unique so it makes sense to count the occurrences and get rid of duplicates when creating the input list.

4.5 Performance

The crawler was run mainly on a Thinkpad T480s with an i7-855OU CPU and 16GB RAM on Windows 10.

There are no exactly measured runtimes but especially the Discogs and Wikipedia crawling steps took quite long as they depend on the available bandwidth and the number of low-level record labels to crawl through.

The estimated runtime of the label crawler for the MPD from start to finish was around 3 weeks, running the Discogs and Wikipedia crawler in parallel. Decoupling the dependency of running the Discogs crawler before the Wikipedia crawler, needed a separate script to combine all findings of the latter crawler with the first one. This script can be found under *utils/fast_forward_mapping.py*.

The LFM-2b only took 10 days, explainable by its smaller size and because the results of the MPD were recycled to save resources.

For both datasets the Discogs crawler took longer than the Wikipedia crawler, which stems from the better accessibility and speed of the Wikipedia-API.

4.6 Reusability

While the label crawler was originally developed with the MPD dataset in mind, a level of generality was kept to reuse it on other datasets. Running it on another dataset (LFM-2b Spotify dataset) resulted in satisfactory result, which can be seen as proof of concept.

Especially the possibility to reuse the final label map from previous crawler runs, which maps low-level record labels to major record labels, increases reusability highly. Any label map can be reused in the postprocessing step of the spotify crawler, where all low-level record label information is being gathered. The only restriction is to follow the format of having two columns with headers *record_label_low* and *record_label_major* in a csv file.

The decision to reuse the label map from a previous crawler run underlies the assumption that in the given time span of 1-2 months no big changes in the record label market happened. But it is noteworthy to remember that the music market is changing constantly. It is possible that the current domination of the three major labels will change by getting more diverse or even more monopolistic. This must be checked before reusing the crawler and its results, making even the defined set of major record labels questionable.

CHAPTER 5

Data Analytics and Bias Analysis

This chapter covers the output of the described setup of the label crawler and machine learning experiments, starting with a description of the used datasets.

5.1 Description of Datasets

This section lists the key characteristics to describe the two datasets used: MPD and LFM-2b.

5.1.1 Dataset 1: Million Playlist Dataset (MPD)

The Million Playlist Dataset (MPD) from Spotify was originally released for the RecSys Challenge 2018 [CLSZ18] but it was re-released for an open-ended challenge on AIcrowd[1] in September 2020.

The MPD consists of one million user-generated playlists, which were created during the period of January 2010 to October 2017. From the billion available user-generated playlists, the chosen million were selected by meeting specific criteria which include:

- The user is from the US and older than 13 years.
- The number of tracks in the playlist is greater than 5 and smaller than 250.
- The playlist contains at least 3 unique artists and 2 unique albums.
- The playlist has at least one follower, other than the creator.

The full list of criteria can be taken from the README file of the MPD.

[1]Link to the AIcrowd MPD challenge homepage.

Structure

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

- Info: General information about this slice of the dataset.
  - Slice (string): Number of playlists ids (PIDs) which are in the file.
  - Version (string): Version of the MPD, which is for the RecSys 2018 and the AIcrowd challenge just 'v1'.
  - Generated_on (timestamp): Creation date of the dataset: December 2017.
- Playlists (array): An array of playlists, where each playlist contains the following items which all realate to the time of the playlist-creation:
  - PID (integer): Unique identifier of the playlist.
  - Name (string): The name the user gave the playlist.
  - Description (string): Optional description of the playlist. This was at the time of the MPD creation a rather new feature so not many playlists of the dataset have a description.
  - Collaborative (boolean): If the playlist is a collaborative playlist, for which multiple users can add tracks.
  - 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.
  - Num_followers (integer): Numbers of followers, not including the creator.
  - Num_edits (integer): Number of separate editing sessions, where a separate session is defined by a break of at least 2 hours.
  - Num_duration_ms (integer): Sum of duration of all tracks in the playlist in milliseconds.
  - Tracks (array): An array of tracks, where each track holds:
    - Track_name (string): The name of the track.
    - Track_uri (string): Unique resource identifier (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.
- ∗ Duration_ms (integer): Duration of the track in milliseconds.
- ∗ Pos (integer): Position of track in playlist.

Additional information like gender and age distribution of the users which created the MPD playlists can be found in the README of the dataset.

Statistics

The dataset also comes with some key numbers, which can be found in the stats.txt, these include:

- There are more than 6.6 million tracks in the MPD, of which 2.2 million are unique.
- There are more than 734 thousand unique albums from nearly 300 thousand unique arists.
- On average a playlist contains 66 tracks, with a median of 49 tracks per playlist.

When counting the occurrences of tracks, albums and artists in the MPD a very dense distribution can be seen. Sorting the tracks descending by their occurrences the first 8.87% of all tracks cover 90% of all track occurrences in the dataset (Fig 5.1a). This is even more dense for albums with 5.51% (Fig 5.1b) and artists with 2.74% (Fig 5.1c).

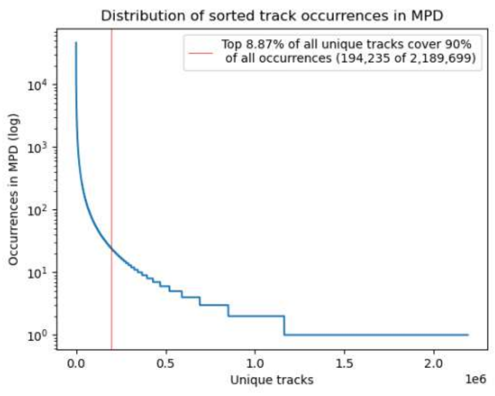
In the preprocessing step the 734 thousand unique albums could be reduced to 170 thousand unique low-level record labels. This results in an even more dense representation of the dataset, with only 1.44% covering 90% of all label occurrences in the MPD (Fig 5.1d).

5.1.2 Dataset 2: Last.fm 2billion (LFM-2b)

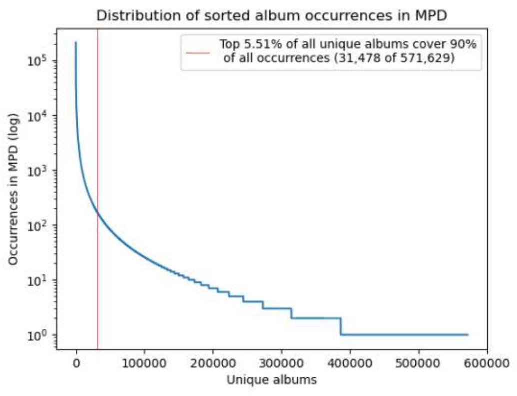
The LFM-2b(illion) dataset is a novel large-scale real-world dataset of music listening records, based on real listening events from the online music platform Last.fm[2], including user related information on gender and demographic [MRPC+21, SBL+22]. It was created in 2021 by team of researchers from the Johannes Kepler University (JKU) of Linz, Austria, the dataset and the corresponding research paper can be downloaded on the LFM-2b web page[3].

The full LFM-2b dataset contains over two billion listening events from over 120 thousand users on roughly 50 million tracks from 5.2 million artists, collected over a span of 15 years (from 2005 until 2020). It is noteworthy to mention that Last.fm decided to discontinue its radio streaming service from April 2014, making it impossible for users to stream

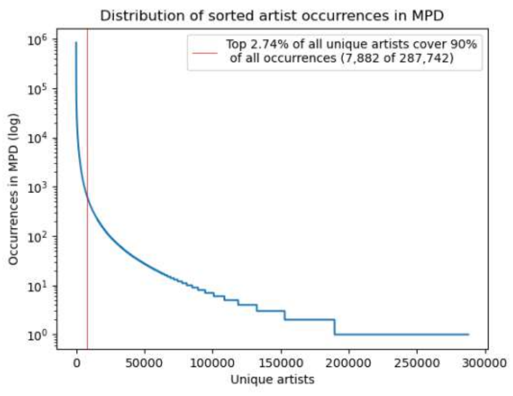
[2] https://www.last.fm/
[3] LFM-2b web page



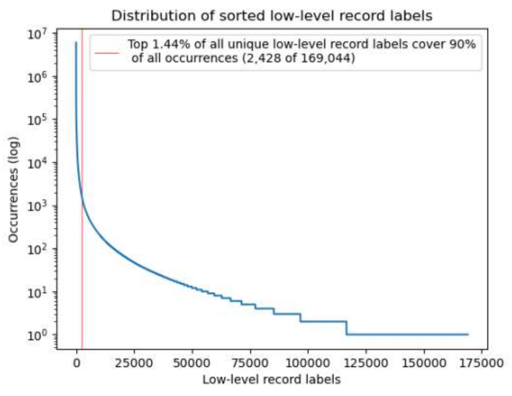
(a) Track density



(b) Album density



(c) Artist density



(d) Low-level record label density

Figure 5.1: Density of tracks (a), albums (b) and artists (c) in MPD, sorted by their occurrences descending in the dataset and with low-level record label density after Spotify preprocessing step (d).

music on the platform directly but relying on Youtube and Spotify for music streaming services.[4] The steps of becoming a sole music recommendation platform therefore might be reflected in the user listening profiles, gathered over this time period.

As the described setup of this work is using Spotify in the preprocessing step, we decided to select just the tracks of the LFM-2b for which already Sptofiy URIs existed, which the team of the JKU thankfully provided in a separate file. It is important to note that this is just a subset of the complete 50 million tracks of the LFM-2b dataset, where the subset contains just 2.4 million tracks which are also included in Spotify's music catalog.

Although Spotify's catalogue is subject constant change, we decided to go with the

[4] Archived article on TechCrunch: "Last.fm Shuts Down Its Streaming Service To Focus On Scrobbling", March 2014

described subset of tracks with URIs to circumvent the challenges of mapping track and artist names to Spotify URIs, like choosing from multiple search results. Although the used set is just a subset of the LFM-2b we will call it LFM-2b (without 'subset') as it represents all tracks with available Spotify URIs at the time of creation. Only when comparing the results to the full LFM-2b we will differentiate between the subset and the full dataset to improve clarity.

Statistics

The following key numbers describe the LFM-2b:

- There are 2.4 million total tracks in the dataset
- There are 2.1 million unique tracks.
- There are 543 thousand unique albums.
- There are 188 thousand unique artists.
- There are 110 thousand unique record labels.

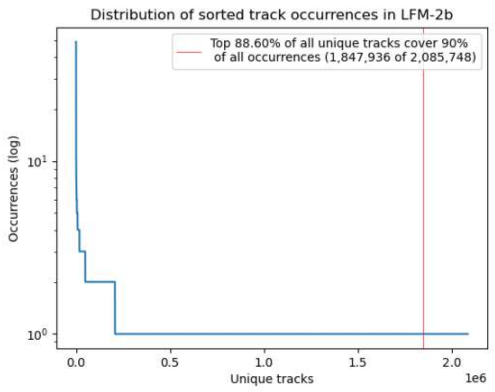
Similar to the MPD we will analyse the density of tracks, albums, artists and low-level record labels (Fig. 5.2). These entities are sorted descending by their occurrences on track level in the dataset. We can see an extremely less dense distribution in the LFM-2b compared to the MPD. Nearly all the tracks only appear once in the dataset, with the top track appearing 49 times (Fig. 5.2a). We can observe an increase of density going from tracks (88.6% cover 90% of occurrences) to albums (56.41%) to artists (33.29%) to low-level record labels (22.44%). This trend is identical for the MPD.

5.2 Descriptive Analysis

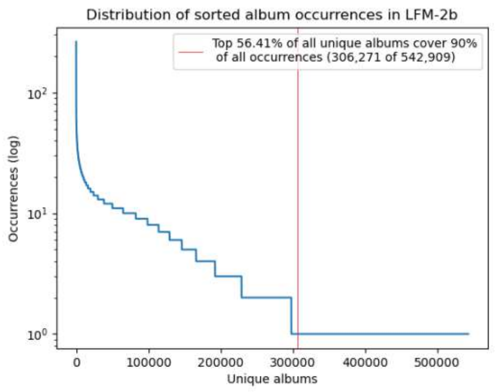
For each classification step the crawler is saving the current progress to analyse not only the final distribution of classified major labels but also the classification gain after completing each step. In the comparison (Fig. 5.3) we can see the different profiles of the MPD and LFM-2b. While the crawler starts from a blank label map with the MPD, the LFM-2b is reusing the resulting label-map from the MPD, explaining the much higher classification level from start.

Reusing the MPD label-map for the LFM-2b resulted in classifying 50.4% of the 110,209 low-level record labels. When taking the number of occurrences of the low-level record labels in account, 85.7% of the 2.4 million tracks could be assigned straight away. Removing *Independent* as top-level class and just focusing on major record labels we end with a classification rate of 51.9% on track level (Fig. 5.3b).

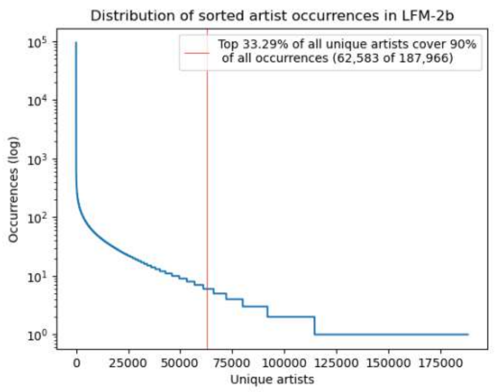
The difference between *Unknown* and *Unclassified* in the stepwise gain chart, can be explained by entries with no valid low-level record label name (e.g., "N/A", nan) are



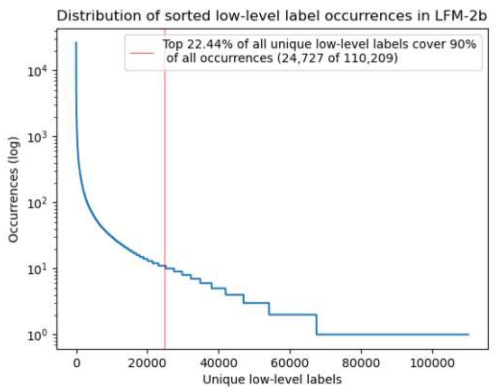
(a) Track density



(b) Album density



(c) Artist density



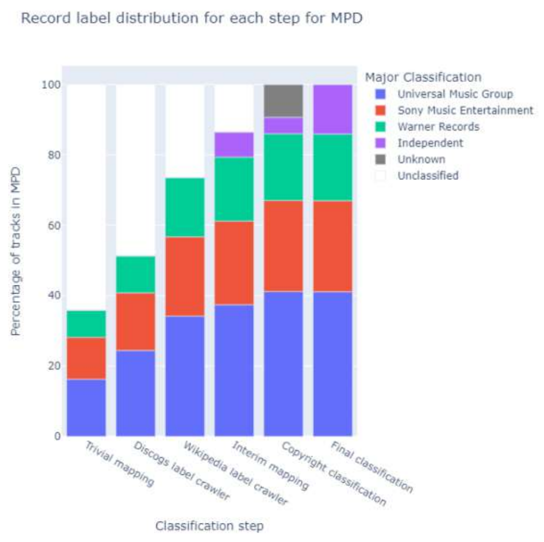
(d) Low-level record label density

Figure 5.2: Density of tracks (a), albums (b) and artists (c) and low-level record labels (d) in LFM-2b, sorted by their occurrences descending.

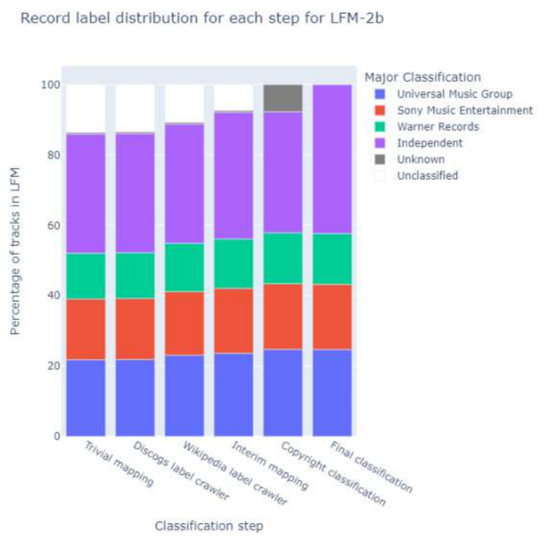
marked as *Unknown*, while *Unclassified* covers all the low-level record labels which were either not touched by the current classification step or for which no classification was possible.

Tables 5.1 and 5.2 list the distribution of majors for each classification step. The full interactive analysis of Fig. 5.3, including the number of set flags for each crawler (e.g., 'Exceeded max depth' or 'Discogs: No Parent found'), can be found in the Google colab notebook[5].

[5]colab.research.google.com/drive/1OKIO9n5BNeNHWl5JF-7REkP5mYo9fVql?usp=sharing



(a) MPD



(b) LFM-2b

Figure 5.3: Development of major label assignment over across individual steps of the crawler.

Major-label	Trivial mapping	Discogs crawler	Wikipedia crawler	Interim mapping	Copyright mapping	Final mapping
Universal	16.24%	24.44%	34.20%	37.43%	41.18%	41.11%
Sony	11.93%	16.38%	22.53%	23.78%	25.85%	25.87%
Warner	7.62%	10.44%	16.79%	18.16%	18.97%	18.97%
Independent	0.11%	0.11%	0.11%	7.11%	4.66%	14.05%
Unknown	0.08%	0.08%	0.08%	0.08%	9.34%	0.00%
Sum	35.98%	51.45%	73.70%	86.55%	100.00%	100.00%

Table 5.1: Step wise change of low-level record labels being classified to majors by the crawler for MPD.

5.3 Bias Analysis

First 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 two datasets. This will be followed by a more in-depth analysis of top-level class distribution and diversity for subsets (e.g., playlist) using the Simpson Index.

General overview

The starting point of the crawler as well as the bias analysis is the list of low-level record label names. To capture an overview of their distribution and possible biases in the two

Major-label	Trivial mapping	Discogs crawler	Wikipedia crawler	Interim mapping	Copyright mapping	Final mapping
Universal	21.81%	21.89%	23.07%	23.66%	24.73%	24.70%
Sony	17.27%	17.33%	18.08%	18.43%	18.67%	18.53%
Warner	12.98%	13.01%	13.75%	14.05%	14.48%	14.43%
Independent	33.82%	33.82%	33.82%	35.97%	34.38%	42.34%
Unknown	0.49%	0.49%	0.49%	0.49%	7.73%	0.00%
Sum	86.38%	86.54%	89.22%	92.60%	100.00%	100.00%

Table 5.2: Step wise change of low-level record labels being classified to majors by the crawler for LFM-2b.

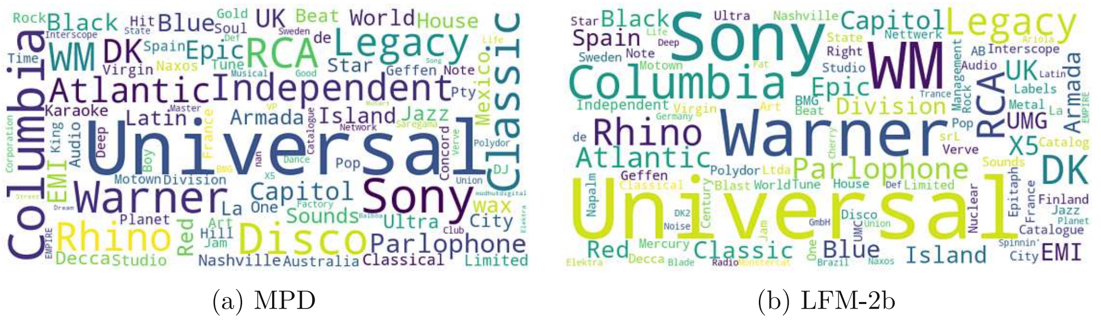

(a) MPD (b) LFM-2b

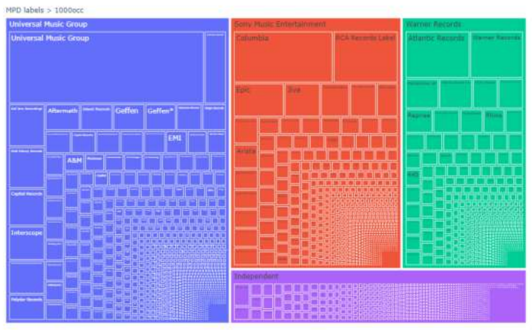
Figure 5.4: Comparison of word clouds of top 100 tokens in names of low-level record labels.

datasets, word clouds help us to grade the importance of specific terms in this soup of words. In Fig. 5.4 we can see the word clouds of the top 100 tokens from the list of low-level record labels, weighted by their occurrences on track level. For the sake of finding the most prominent tokens over all low-level record names, bigrams (combination of two tokens) are disabled and the minimum token length is set to 2. Note that no color highlighting by top-level class is possible as the tokens can be shared by low-level record labels from different majors. This is a simplified overview with a focus on label names, after removing stopwords and general tokens like 'Music', 'Group' or 'Records. Also, there is no handling for resolving abbreviation which would lead to an increased weight for specific tokens (e.g., 'WM' stands for 'Warner Music').

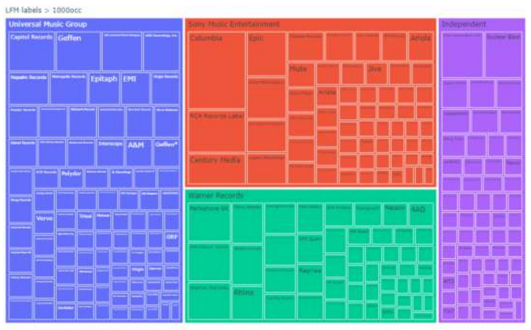
Compared to this, Fig. 5.5 shows an overview on how the low-level record labels are distributed after being classified to a final class. Although the plot is restricted to show only low-level labels with more than 1000 occurrences, it is visually too complex. Therefore, an interactive version where pinning and zooming is possible, can be found in the google colab notebook[6].

Fig. 5.6 shows the distribution of low-level record labels, classified to a major label class, for both datasets. The difference in coverage compared to the treemaps stems from the

[6] google colab notebook

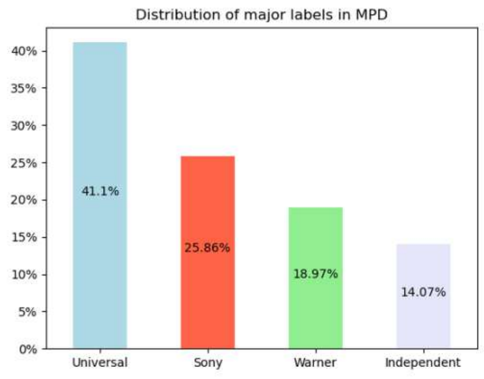


(a) MPD

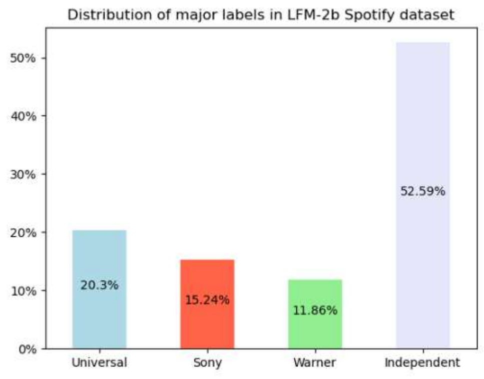


(b) LFM-2b

Figure 5.5: Comparison of treemaps of classified low-level record labels with more than 1000 occurrences.



(a) MPD



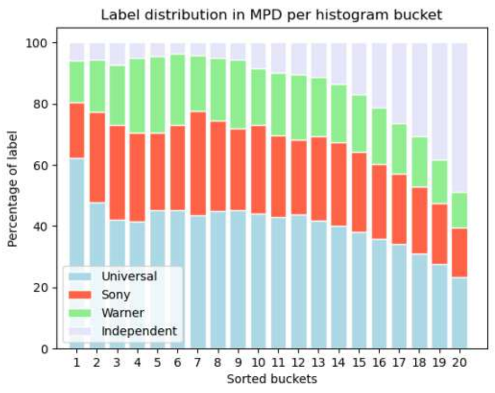
(b) LFM-2b

Figure 5.6: Comparison of top-level class coverage in enriched datasets.

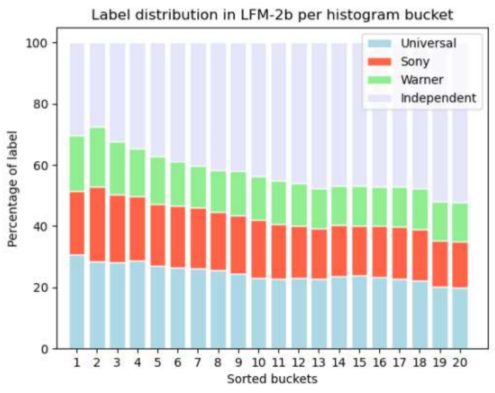
constraint of only including the top low-level record labels in the treemap to prevent cluttering.

To show the distribution of top-level classes in the enriched datasets in more detail, the list of albums, sorted by their occurrences in the dataset, is interesting. Figure 5.7 displays the albums in 20 buckets, where each bucket represents the same number of occurrences of the album in the dataset. Therefore, each bucket contains a higher number of albums than the one before.

For the MPD (Fig. 5.7a) the first bucket consists only of 67 albums (0.009% of all albums), the second bucket of 93 albums (0.012%) and the last one of 631,500 albums (85.9%). For the LFM-2b (Fig. 5.7b) the first bucket contains 2,384 albums (0.44% of all albums), the second 4,626 albums (0.85%) and the last 118,789 albums (21.88%).



(a) MPD



(b) LFM-2b

Figure 5.7: Distribution of top-level classes per album, sorted descending by occurrences in the dataset.

Playlist and Profile Analysis

To study the diversity of top-level classes in subsets of the dataset we will use the Simpson Index [Sim49]. This metric measures the probability that two tracks from a subset belong to the same top-level class. The Simpson Index is calculated with:

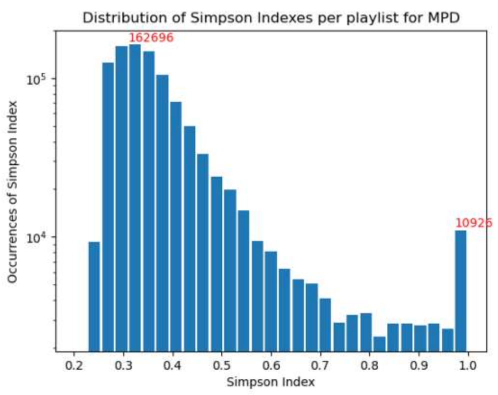
$$\lambda = \sum_{i=1}^{R} p_i^2 \tag{5.1}$$

where R describes the richness of the classes, in our case this is the number of top-level classes: 4. p_i is the probability for a class i that a randomly drawn track belongs to this class. A low λ therefore stands for high diversity and vice versa.

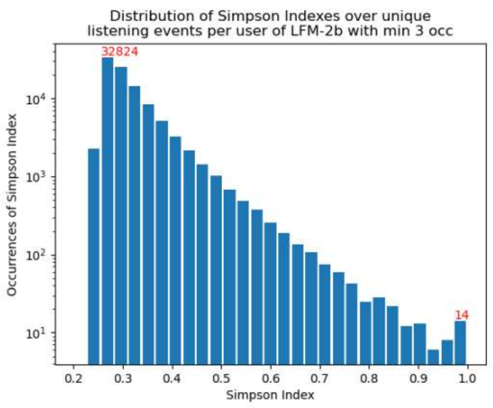
In Fig. 5.8a we can see the distribution of Simpson Indexes of top-level classes per playlist for the MPD. Note that the last histogram bucket includes not only playlists with an Simpson Index equal to 1 but all playlists with an Simpson Index between 0.97 and 1.

For the LFM-2b no playlists as such exist, but the listening histories of the users are available. There are over 120,000 users, with over 2 billion listening events in the dataset. The average number of listening events per users is nearly at 18,000 with a median of 4,200, which shows quite a different picture compared to the playlists of the MPD which were created with the rule to have a length between 5 and 250.

The following filters follow the purpose of shaping the user listening histories to something more comparable to the MPD's playlist. We understand the fundamentally different characteristics of a user listening history and a regular playlist, where the latter marks a much more conscious music selection, more like a personal music collection. Investigating how often a user of the LFM-2b dataset listened to a track will provide interesting insights



(a) MPD (playlists)



(b) LFM-2b (user profiles)

Figure 5.8: Distribution of Simpson Index of major label diversity per playlist (MPD) and user profile (LFM-2b).

in the dataset nevertheless and should help to understand possible record label biases connected to this.

Description	All profiles			Profiles with SI1	
	Number	Avg. length	Med. length	Num. (indi)	Avg. length
With dup.	119,186	12,339.59	3,056	15 (13)	191.2
No dup.	118,325	3,225.05	1,184	2 (2)	33.5
Min 2 occ	108,572	1,691.92	616	1 (1)	35.0
Min 3 occ	97,913	1,217.25	500	13 (13)	38.8

Table 5.3: Overview of LFM-2b listening profiles with different filters, with an overall minimum profile length restriction of 30.

To analyse the Simpson Index distribution over the LFM-2b user profiles, a restriction to for a minimum profile length can be motivated. Very short profiles have a naturally high probability of low major label diversity and would distort the Simpson Index distribution. This can be additionally motivated by the previously stated rule for the MPD creation, which also included a minimum length restriction. To introduce an arbitrary maximum length restriction was not found to be reasonable.

The chosen minimum length threshold is 30, which is between the MPDs minimum restriction of 5 and its average playlist length of 66. This length restriction will apply to all numbers of user profile analysis to follow. The first row of Table 5.3 shows the application of the length restriction on all user profiles, reducing the average length to around 12,300.

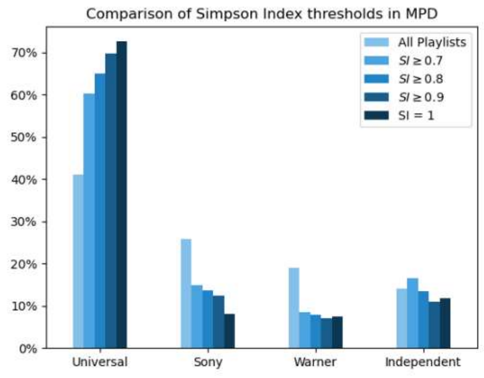
In a next step duplicate tracks from the user listening histories are removed. Although Spotify allows duplicate tracks in playlists, the platform is not promoting this behaviour. The current handling for this is a notification-dialog, popping up if an added track is already in a playlist. The dialog requires confirmation by the user to add the track anyways, with the option to not add the track is being highlighted. After reducing the listening history for each user to only the unique tracks, this results in around 118,300 user-playlists, with an average length of around 3,200 and a median of 1,184. Only 2 of these user profiles have an Simpson Index of 1, with both of them consisting exclusively of *Independent* tracks.

To move the research on the LFM-2b even further in the direction of listening profile analysis instead of listening behaviour analysis, an additional filter is introduced for how often a track must appear in a user listening history. Over all users the average number of occurrences of a track is 3.5 with a median of 1. Just taking tracks in account which appear at least two times in a user listening profile, results in nearly 108,500 user profiles, with an average length of around 1,700. Only one profile has an Simpson Index of 1, consisting only of *Independent* tracks with a length of 35. For the minimum occurrence threshold of 3 (around 98,000 profiles), only 13 of the profiles have an SI of 1, all from *Independent* with an average length of 38.8. The last two rows of Table 5.3 outline these results.

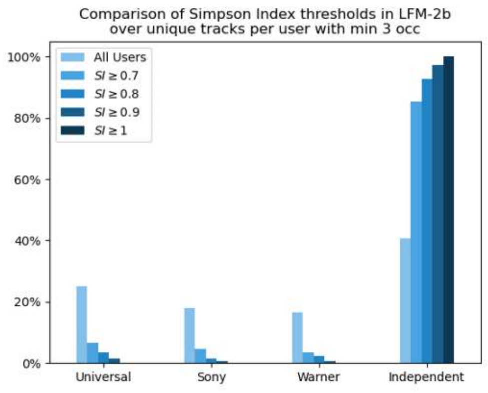
The final setup of user profile analysis for the LFM-2b to compare it to the MPD was defined as user profiles with a minimum of 30 tracks and where each track appears at least 3 times in the profile. This is motivated as seeing playlist as conscious music collections and an occurrence threshold of 3 should exclude random or unintentional interactions. Figure 5.8b shows the distribution of Simpson Indexes per user with the described configuration. The difference of 157 profiles in the rightmost bucket to the 102 in Table 5.3 is again linked to the bucket size, including profiles with an SI from 0.97 to 1.

To understand the biases related to major labels better, it is interesting to study the distribution of majors in playlists and profiles with a high Simpson Index, which are playlists with a low diversity of major labels. In Fig. 5.9 we increase the threshold δ for the Simpson Index from 0.7 to 1 in four steps, analysing how many tracks belong to a major label in the selected subgroup of playlists. These distributions are compared to the initial situation of analysing the distribution of majors in all playlists/profiles (Fig. 5.6).

In Table 5.4 we can see a more detailed analysis of the different subgroups of playlists of the MPD where each subgroup is defined by the Simpson Index exceeding a defined threshold δ. Note that these groups are no intersections but subsets of each other with the lowest δ of 0.7 including all other subsets and $\delta = 1$ being the strictest one. Therefore, the number of playlists and tracks decreases with higher threshold values. pl_x defines the group of playlists in the subset where x is the dominant major label. There is no defined dominant top-level class in playlists with an Simpson Index below 0.5, which is why there are no pl_x available for all playlists.



(a) MPD



(b) LFM-2b

Figure 5.9: Comparison of major distribution for different Simpson Index (SI) thresholds for the MPD and LFM-2b.

	All Playlists	SI $\geq$ 0.7	SI $\geq$ 0.8	SI $\geq$ 0.9	SI = 1
Tracks total	66,346,428	1,955,746	1,423,122	957,690	356,696
Playlists total	1,000,000	40,470	28,232	18,328	9,822
Coverage Universal	41.10%	60.15%	64.98%	69.72%	72.57%
Coverage Sony	25.86%	14.92%	13.70%	12.35%	8.13%
Coverage Warner	18.97%	8.41%	7.81%	6.99%	7.43%
Coverage Independent	14.07%	16.52%	13.52%	10.94%	11.87%
Number of pl_{univ}	-	24,793	18,440	12,884	7,126
Number of pl_{sony}	-	6,023	3,818	2,107	846
Number of pl_{warn}	-	3,147	2,104	1,251	621
Number of pl_{indi}	-	6,507	3,870	2,086	1,229
pl_{univ} avg. length	-	49.08	51.31	52.47	36.33
pl_{sony} avg. length	-	44.79	48.73	54.93	34.26
pl_{warn} avg. length	-	46.55	49.33	51.49	42.67
pl_{indi} avg. length	-	49.60	48.34	48.64	34.45
Avg. pl length overall	66.35	48.33	50.41	52.25	36.32

Table 5.4: MPD playlists analysed, grouped by their Simpson Index of major labels.

CHAPTER 6

Methodology: Recommender System Experiments

This chapter covers the structure and setup of the systematic machine learning experiments using the enriched datasets, resulting from the previously explained data gathering.

6.1 Offline Recommender System Performance Comparison

The setup of experimenting with offline recommender systems to improve their performance is structured around the ACM recommender system challenge 2018 (RecSys18). The basic idea for the proposed experiment is to reproduce a submission for the challenge and re-rank the tracks of the submission with the gathered label information and compare the evaluation results.

Recommender System Challenge 2018

The RecSys18 was sponsored by Spotify and is based on its one million playlist dataset (MPD). The challenge was set up to improve automatic playlist continuation, which has been identified as one of the key challenges in music recommender system research [SZC+18].

The full MPD was used as training set and an additional challenge set of 10,000 incomplete playlists was used for evaluation. The 10,000 playlists were covering 10 scenarios with 1000 playlists each:

1. Title only
2. Title and first track

3. Title and first 5 tracks
4. First 5 tracks only
5. Title and first 10 tracks
6. First 10 tracks only
7. Title and first 25 tracks
8. Title and 25 random tracks
9. Title and first 100 tracks
10. Title and 100 random tracks

Each submission would have to include a recommendation of exactly 500 tracks per playlist of the challenge set, ordered by their relevance with the top recommendation coming first. Recommending duplicate tracks per playlist would resolve in a failed submission. The challenge was split in two lanes: main and creative track. The main track would only allow the MPD as training source while the creative track also allowed other information sources.

The RecSys18 challenge was re-opened for public participation on AIcrowd[1] in September 2020. They scrapped the difference between main and creative track, allowing the use of any resource for the challenge.

The submissions would be evaluated with the following metrics, using the withheld tracks of the challenge set as ground truth:

- *R-Precision (R-prec)*: This metric describes how relevant the submitted tracks are compared to the withheld tracks. Also matching artists are considered for this metric, with a partial score of 0.25. Let G_T and G_A be the ground truth URIs for tracks and artists resp. and S_T and S_A the submitted track and artist URIs, then:

$$R\text{-}prec = \frac{|S_T \cap G_T| + 0.25 \cdot |S_A \cap G_A|}{|G_T|} \tag{6.1}$$

  The order of the recommended tracks is not relevant for this metric and the final *R-Prec* of a submission is the average value over all playlists.

- *Normalized Discounted Cumulative Gain (NDCG)*: This metric is rewarding relevant tracks when they are placed in a higher position. It is using the discounted

[1] https://www.aicrowd.com/challenges/spotify-million-playlist-dataset-challenge

cumulative gain (DCG), which is calculated for a list of 500 tracks, sorted descending by their recommendation score:

$$DCG = \sum_{i=1}^{500} \frac{r_i}{log_2(i+1)} \tag{6.2}$$

Where i is the position of the item in the submitted list and r_i the position of this item in the ground truth. The $NDCG$ is normalized by the ideal DCG ($IDCG$) where the recommended tracks are ranked perfectly. So finally:

$$NDCG = \frac{DCG}{IDCG} \tag{6.3}$$

- *Recommended songs clicks (clicks)*: This metric is simulating Spotify's own feature to recommend 10 songs for a playlist, with the possibility to refresh this list to get to the next 10 recommendations. Let R be the submission list and G the ground truth, then:

  $$clicks = \left\lfloor \frac{\arg\min_i\{R_i : R_i \in G\} - 1}{10} \right\rfloor \tag{6.4}$$

  If not a single relevant track is in the submission, *clicks* is set to 51. This is the maximum number of clicks + 1. Again, this metric is averaged over all playlists.

The presented description of the metrics is strongly oriented by the explanation in Appendix A of [SZC+18]. Finally, we can conclude that a high *R-prec* and *NDCG* and a low *clicks* value are desired.

KAENEN

One of the submission teams for the RecSys18 was KAENEN[LKLJ18], which achieved good results with a comparably simple algorithm.

For its track-based approach (scenario 2-10) the hybrid model uses a weighted ensemble of item-based collaborative filtering, session-based nearest neighbors, inverse document frequency extension (idf-knn) and matrix factorization. For the cold-start scenario with just the title and no or only a single track (scenario 1-2) a combination of string matching and title factorization is used. For the creative track additional meta data was crawled from the Spotify-API, including loudness, tempo and energy of a track.

The team managed to achieve the seventh place in the main track and the third in the creative track. The exact values for the main track are: 0.2091, 0.3747 and 2.0540 for *R-prec*, *NDCG* and *clicks* resp. For the main track KAENEN was only 5% to 7% behind the winner's submission in all accuracy metrics, giving evidence to the introductory statement of the team: *"While complex methods are usually able to outperform such simple methods, the performance differences are often not very large."*[LKLJ18]

The low complexity of the KAENEN model with its already good results, motivated to use the publicly available algorithm[2] for experiments on offline recommender system performance.

Experimental Setup

To experiment with different re-ranking configurations, a baseline result is needed for comparison. To generate this, the submission pipeline of the AIcrowd challenge was used, evaluating the result of the KAENEN algorithm. Although external information sources besides the MPD are allowed in the AIcrowd challenge, we excluded the step of gathering meta data from Spotify-API to reduce complexity. Instead, the gathered record label information can then be used to re-rank the KAENEN submission in a post-filter step.

The challenge dataset contains 66,234 unique track URIs from 32,369 albums. All tracks also appear in the MPD dataset and can therefore be assigned to top-level classes, using the previously crated label-map. For all playlists of the challenge set which include tracks (scenario 2-10) the Simpson Index of major label diversity can be calculated. The following parameters configure the re-ranking:

- *min_length*: The minimum length of the given tracks of a playlist to consider it for re-ranking. The lengths are distributed discrete: [0, 1, 5, 10, 25, 100].
- *SI_threshold*: Only consider challenge playlists which have a minimum Simpson Index above this threshold.
- *rerank_first_n*: How many of the recommended 500 tracks should be considered for re-ranking? Example: If this parameter is set to 10, re-ranking is only happening for the first 10 tracks of the recommendation.

The result of different configuration of these parameters with additional information (e.g., how many playlists were re-ranked) will be evaluated. We will focus on the *NDCG* and *clicks* metrics for the evaluation as *R-prec* is invariant to the order of the tracks.

6.2 Feedback Loop Evaluation

For answering H2 and investigating feedback loops concerning the distribution of record labels over multiple iterations of a recommender system the experimental setup of [FSB21] was reused. In their simulation to mimic feedback loops Ferraro et al. used recommender system to study the distribution of gender of artists. For this a matrix-factorization based collaborative filtering approach was used, which utilizes the Alternating Least Square (ALS) algorithm [HKV08]. This approach was adapted to fit the research goal studying record label distribution in recommendations.

[2] https://github.com/rn5l/rsc18/

A matrix of user-item interactions exists, where the items are music tracks enriched with record label information. To mimic feedback loops, we assume that each user accepts the top-10 of the 100 recommended tracks, increasing the interactions in the user-item matrix for these 10 tracks for each user. The matrix-factorization based ALS algorithm is retrained after each iteration on the altered user-item interaction matrix. This results in a new model after each iteration, which is used for recommendations. This process is repeated up to n iterations.

Specific metrics are introduced to understand the behaviour of the recommender system comprehensively, measuring the probability and coverage of a record label in the recommendations.

- *First*: This is the first position of a specific label in the 100 recommendations, averaged over all users.
- *Recommended*: This is the percentage of how many tracks in the 100 recommendations are covered by a specific label, again averaged over all users.

Note that different setups of feedback loop simulations exist in literature [FJK+20, FJS20] changing the selection of the recommendations (e.g., selecting a random subset instead of top-10) or how often the model is retrained (e.g., after every third iteration). The chosen configuration depends on the used dataset as well as the defined research goal.

CHAPTER 7

Evaluation in Recommender Systems

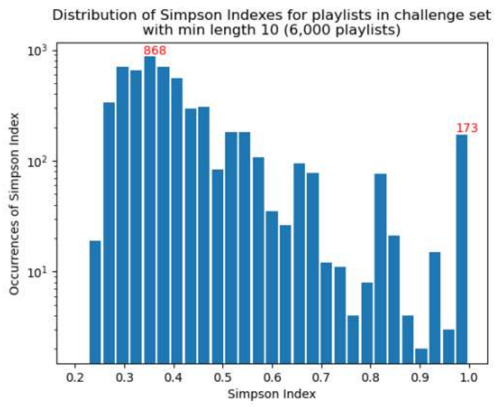
7.1 Offline Recommender System Performance Comparison

The re-ranking of submissions for the Recommender System Challenge 2018 (RecSys18) hosted on AIcrowd is based on the challenge playlist dataset, therefore we begin with an analysis of this set.

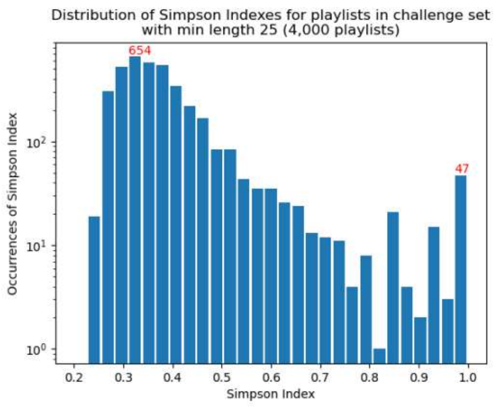
Challenge Set Analysis

The 10,000 playlists are separated in buckets of length 0, 1, 5, 10, 25 or 100. To reduce outliers in short playlists (length 0 to 5), a minimum length thresholds of 10 and 25 are used. The distribution of Simpson Indexes of top-level class diversity is therefore analysed for playlists with minimum length 10 (Fig. 7.1a) and 25 (Fig. 7.1b).

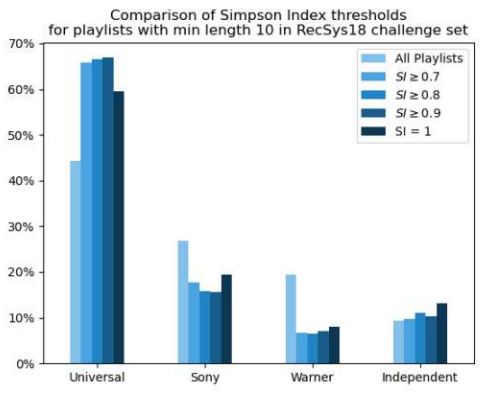
All tracks of the challenge playlist set appear in the MPD. Therefore, a complete assignment of all 66.234 unique tracks to top-level classes is possible. The analysis is again separated for playlists with minimum length 10 (Fig. 7.1c) and 25 (Fig. 7.1d). Note that 'All Playlists' includes the same number of playlists, stated in the SI distribution chart above of 6,000 and 4,000 playlists for minimum length 10 and 25 resp. Increasing the SI threshold results in a subset of 'All Playlists'. For minimum length of 10 this adds up for 329, 294, 193 and 172 playlists for SI thresholds of 0.7, 0.8, 0.9 and 1 resp. For minimum length 25 there are 128, 93, 67 and 46 playlists for the same SI thresholds.



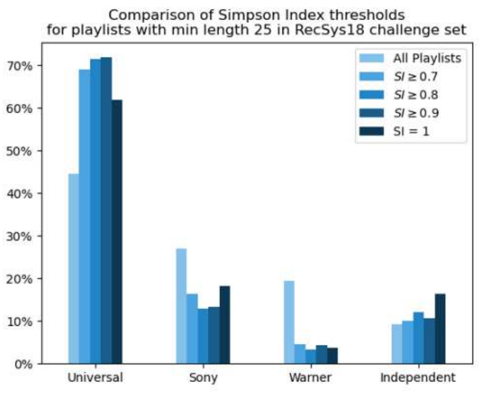
(a) SI distribution (≥ 10 tracks)



(b) SI distribution (≥ 25 tracks)



(c) Major distribution (≥ 10 tracks)



(d) Major distribution (≥ 25 tracks)

Figure 7.1: Comparison of Simpson Index (SI) distribution of playlists with minimum length 10 (a) and 25 (b) and distribution of top-classes for different SI thresholds (c,d).

Re-Ranking Evaluation

The baseline KAENEN submission was created, running the algorithm from the teams repository[1] on the full MPD without meta-data crawling. All further submissions are manipulations of this baseline submission, re-ranking the 500 recommended tracks per playlist of the challenge set.

Table 7.1 shows the results of different re-ranking configurations, using the AIcrowd submission pipeline. The columns 'min length', 'first' and 'SI' denotes the three parameters *min_length*, *rerank_first_n* and *SI_threshold*, where 'pl touched' marks the number of playlists for which re-ranking happened.

The baseline row with no parameters set (∗) marks the plain KAENEN submission, to which the NDCG of all other submissions is being compared to ('+/-NDCG%'). The

[1] https://github.com/rn5l/rsc18/

last four rows of the table (∗∗) denote submissions which do not use the Simpson Index threshold but just re-rank the tracks based on the dominant top-level class of the known tracks of the challenge playlist. The number of playlists re-ranked ('pl touched') differs from the number of playlists above the minimum length threshold (e.g., 8000 for 'min length' of 5), because no re-ranking is happening if there is a tie of dominant top-level classes.

min length	first	SI	pl touched	R-prec	clicks	NDCG	+/-NDCG%	
25	25	0.8	93	0.207436	2.0603	0.3722638	0.000266%	
25	20	0.8	93	0.207436	2.0603	0.3722635	0.000185%	
25	30	0.8	93	0.207436	2.0604	0.3722633	0.000130%	
25	10	0.8	93	0.207436	2.0603	0.3722629	0.000031%	
0	0	0	0	0.207436	2.0603	0.3722628	-	∗
25	10	0.8	86	0.207436	2.0603	0.3722625	-0.000008%	
25	10	0.7	128	0.207436	2.0603	0.3722617	-0.000297%	
25	50	0.8	93	0.207436	2.0605	0.3722609	-0.000507%	
25	10	0.9	67	0.207436	2.0603	0.3722606	-0.000597%	
25	10	0.8	105	0.207436	2.0603	0.3722595	-0.000881%	
10	10	0.9	193	0.207436	2.0603	0.3722383	-0.006596%	
10	10	0.8	294	0.207436	2.0603	0.3722238	-0.010491%	
10	10	0.7	329	0.207436	2.0603	0.3722225	-0.010818%	
10	20	0.8	294	0.207436	2.0610	0.3722019	-0.016363%	
10	50	0.5	1047	0.207264	2.0677	0.3717285	-0.143540%	
5	50	0.6	1259	0.207188	2.0680	0.3716615	-0.161524%	
25	10	-	3825	0.207436	2.0603	0.371583	-0.182549%	∗∗
10	10	-	5560	0.207436	2.0603	0.371142	-0.301124%	∗∗
5	10	-	7160	0.207399	2.0603	0.370885	-0.370120%	∗∗
5	500	-	7160	0.163617	2.2478	0.3525986	-5.282357%	∗∗

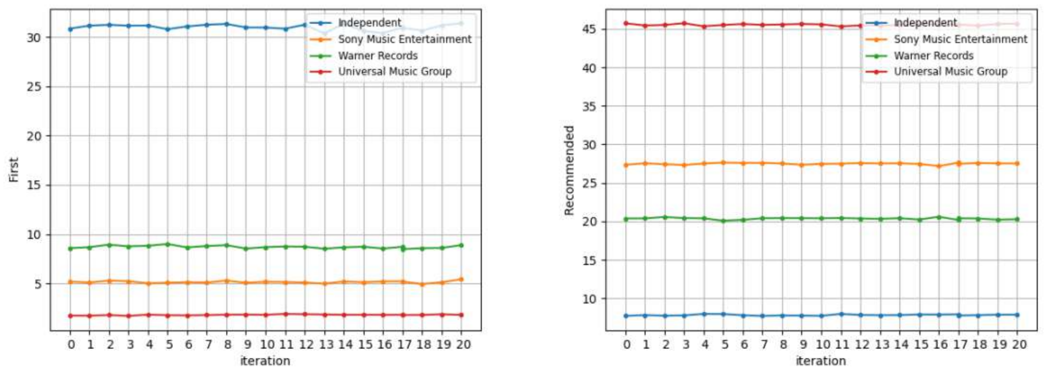
Table 7.1: Comparison of different re-ranking configurations, evaluated on AIcrowd.

7.2 Feedback Loop Evaluation

The described setup of the simulation to mimic feedback loops (chapter 6.2) was used on the MPD and the LFM-2b dataset. All experiments were run by Andrés Ferraro on a HPC cluster, using 25GB of RAM and 4 CPUs.

Results MPD

For the MPD each playlist was interpreted as a user and the tracks from the dataset as items for the user-item matrix. For the first experiment, the pool of tracks to recommend from consists of all tracks in the dataset with the possibility for tracks to appear repeatedly in a playlist. This experiment was stopped after two iterations because the run times were so extensive and made a higher number of iterations unfeasible.



(a) First position of major label (b) Coverage of major label in recommendations

Figure 7.2: Results of reduced feedback simulations with the MPD, recommending songs only for 1% of the playlists. *First* and *Recommended* are averaged over all users.

To cut complexity, the experiment on the MPD was repeated in a reduced format, just recommending tracks to 1% of the one million playlists. The 100,000 playlists were selected randomly. This setup allowed running full 20 iterations (Fig. 7.2). Table 7.2 shows the detailed output for the first two iterations.

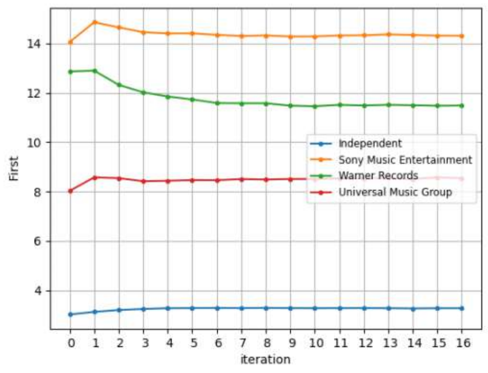
	1^{st} iteration		2^{nd} iteration	
Major-label	*Recommended*	*First*	*Recommended*	*First*
Universal	45.70	1.78	45.66	1.73
Sony	27.32	5.22	27.39	5.26
Warner	20.44	9.05	20.37	8.89
Independent	7.82	31.37	7.84	30.65
Unknown	1.29	99.28	1.27	99.27

Table 7.2: Result of feedback loop simulation reduced MPD setup, recommending tracks for 1% of the playlists.

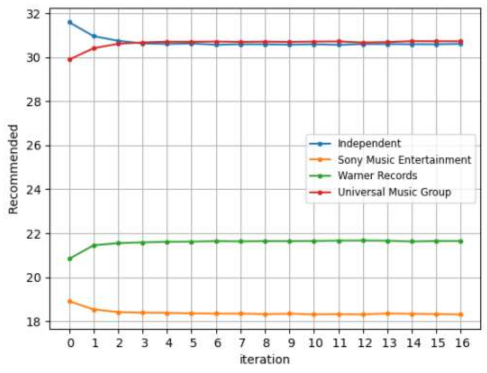
Results LFM-2b

The LFM-2b has an already defined set of users which is used in this experiment with the restriction of removing tracks that appear less than 15 times in the full dataset. Also, no tracks are recommended to a user for which he already interacted in the original dataset with.

This setup had a runtime of approximately 12 hours per iteration on the described HPC cluster and was stopped at 16 iterations (Fig. 7.3).



(a) First position of major label



(b) Coverage of major label in recommendations

Figure 7.3: Results of the feedback simulations with the LFM-2b. *First* and *Recommended* are averaged over all users.

CHAPTER 8

Discussion

In this chapter the previously presented evaluation results will be analysed and discussed with the goal of answering the posed research questions and hypotheses.

8.1 Descriptive Analysis

The discussion of the descriptive analysis of both datasets and their low-level record labels will answer RQ1 *"To which extent is it possible to identify the owner of a piece of music?"*

Record Label Distribution

When analysing the distribution of the entities of the MPD (tracks, albums, artists and low-level record labels) we can see an overall strong compression of the top entities (Fig. 5.1). All these entities, sorted descending by their occurrences on track level in the dataset, show a high density of high frequent items with an exponential drop. This trend is the strongest for low-level record labels, where only 1.44% cover 90% of occurrences in the dataset, followed by artists, albums and finally tracks. This means that we only need very few of these entities to describe the dataset sufficiently. This distribution suggests the existence of a popularity bias, where single tracks appear so frequent over all playlists, that a recommender system might adapt this bias for its recommendations.

The Spotify sub-set of the LFM-2b shows a completely different picture, with less or nearly no such compression (Fig. 5.2). Especially the distribution of tracks is interesting as most tracks only appear once in the dataset and the occurrence count maxes out at 49. This can be explained by the origin of the Spotify sub-set of the LFM-2b, which is just a collection of all tracks of the full LFM-2b dataset which could be mapped to the Spotify catalogue by the authors of the dataset. The very different structure and purpose of the

LFM-2b Spotify sub-set and the MPD suggests that direct comparisons must be taken with a grain of salt, keeping their differences in mind.

Analysing the distribution of tracks in the LFM-2b (Fig. 5.2a), poses the question of why some tracks even appear multiple times in the dataset. This means that multiple track-IDs (the identifiers in the LFM-2b) share the same Spotify track-URI. A possible reason for this is, that these track-IDs are used to describe different versions of the same track or the same track in different albums. This hypothesis is supported by manually checking the top tracks of the LFM-2b where all three tracks with more than 40 occurrences are classic music tracks from the 1980s. A deeper study of the LFM-2b composition and generation process would be necessary to answer this question thoroughly.

The LFM-2b and MPD share the same order of density, where tracks are less densely distributed than albums, which are less dense than artists and so on. The reason for this lies simply in the nature of these entities: usually a record label covers multiple artists, which produce multiple albums which contain multiple tracks. Given this pre-existing structure we want to focus on the distribution of low-level record labels, where over 22% of the low-level record labels cover 90% of the track occurrences in the LFM-2b (Fig. 5.2d). The higher number of labels with a moderate number of occurrences is in accordance with the composition of the dataset but also with the type of platform the dataset originally stems from. Last.fm was a completely free and user activity driven music streaming service up until 2014. Today Last.fm is solely focusing on music recommendation now, relying on Youtube and Spotify for music streaming services. But a difference in the user base and their music consumption patterns over the time span of gathering the LFM-2b (from 2005 to 2020) is a possible explanation for the higher number of low-level record labels with a relevant number of occurrences in the LFM-2b.

Crawler Results

When analysing the development of top-level classes for both datasets we can see a big difference in the results (Fig. 5.3). For the MPD big gains can be registered in the two crawler steps (Discogs and Wikipedia) in regard to the major labels. The Wikipedia crawler achieved a higher classification gain with 9.8%, 6.2% and 6.3% (for *Universal*, *Sony* and *Warner* resp.) compared to 8.2%, 4.4% and 2.8% of the Discogs crawler. This could be explained by the higher publicity that Wikipedia enjoys compared to Discogs, resulting in broader source of information. From this also a bigger budget for resources and therefore faster API resources could be inferred. But it is more apparent, that this difference is also reflecting the difference of platforms search engines. While the Discogs search engine only works with strict string matching, the Wikipedia search allows for fuzzy search tokens, increasing the possibility of matches. Coming back to the major label distribution per step, we can see that each individual step and additional source increases the classification rate while the distribution between the major labels remains consistent.

Comparing the crawler results on the MPD to the top-level assignment in previous work

[KH19], we are confident to confirm the found trends of major label dominance. Additionally, the number of tracks with no major label assigned (around 26% of *Unknown* and *Independent* in previous work), could be reduced significantly (only 14% of *Independent* record labels).

For the LFM-2b, the crawler starts with a much higher classification rate of around 86%. This stems from building on the already made assignments of the MPD, where derived top-level classifications for low-level record labels overlap heavily between the two datasets, resulting in a high coverage from the very beginning of the crawler process. Interestingly it can be seen that the distribution of major record labels vs. *Independent* differs significantly, with the LFM-2b having a much higher share of *Independent* labels in the beginning (ca. 34%) compared to the MPD's final assignment (ca. 14%). Differences between the policies and user bases of the two platform were already stated, but it is also possible that this bias was introduced by filtering the LFM-2b tracks to get Spotify URIs. What the exact reason for the different share of *Independent* classified labels is, cannot be answered at this point and needs further investigation.

Research Question 1

To answer RQ1, we want to recap the basic structure of the label crawler. In the methodology chapter of the data gathering (chapter 3) a deterministic approach was motivated, where each made classification is based on sound evidence (e.g., no assignment on ties or a specified listed parent company on Wikipedia as classification reason). This process was monitored by random sanity checks on samples during development to see if the resulting classifications are valid. The share of *Unknown* major labels, peaked below 10% for both datasets in the copyright mapping step: 9.34% for MPD and 7.73% for LFM-2b. Although we defined later all the low-level labels with no sound information for major label classification as *Independent*, this demonstrates the high degree of identification of ownership through the proposed process. Finally, we can answer RQ1: Based on the evaluated research, we are confident to cover more than 90% of possible identifications of the ownership of a piece of music.

It is noteworthy to remember, that no ground truth exists for this classification problem, which is why plausibility checks were used during development and evaluation. It is suggested to use such checks in combination with a general review of the hierarchy of the major labels in the current music market when rerunning these experiments.

8.2 Bias Analysis

Discussing the analysis of biases in both datasets will answer RQ2 *"What kind of biases in music datasets can we identify regarding record labels?"*

General Overview

The word clouds (Fig. 5.4) and treemaps (Fig. 5.5) present a general overview of the distribution and biases of major labels in both datasets.

In both word clouds we can see that the most prominent tokens (*'Universal'*, *'Sony'*, *'Warner'*) relate directly to the major labels, followed by the biggest distributor companies (*'Columbia'*, *'DK'*, *'Atlantic'*, *'RCA'*, *'Capitol'*, etc.). These prominent tokens are nearly identical for both datasets from which a general dominance of these distribution companies can be inferred. The strong dominance of *Universal* is probably the most significant inference from this analysis.

For the LFM-2b word cloud (Fig. 5.4b) we can see a bigger gap between the most prominent tokens and the less important ones, compared to a more linear regression of token weight for the MPD. This is interesting as the LFM-2b showed a less dense distribution of low-level record labels (Fig. 5.2d). This could be explained by a general smaller amount of low-level record labels being classified to major labels (58%) compared to the MPD (85%).

The token *Independent* is much less prominent in the LFM-2b compared to the MPD, which is on place 26 compared to place 9 for the MPD. But for the LFM-2b the share of labels being classified to *Independent* is much higher compared to the MPD. A possible explanation for this is that the music catalogue represented by the LFM-2b dataset contains much more tracks which have no label information on Spotify (e.g., no label name) and are therefore classified as *Independent* but do not show up in the analysis of label names.

From analysing the treemaps (Fig. 5.5) we can see the different coverage of top-level classes between the two datasets. The difference to the share in the full dataset (compare Fig. 5.3) stems from the restriction of showing just the low-level record labels with more than 1000 occurrences in the treemap. While for both datasets we can see the same hierarchy of majors (*Universal* > *Sony* > *Warner*), *Universal* is showing a much stronger dominance in the MPD's top low-level record labels.

We want to stress that the comparison between both datasets with the same occurrence-threshold of 1000 is a bit misleading as this is including much more low-level record labels for the MPD as for the much smaller LFM-2b. The chosen visualization is motivated by reflecting the different distribution of low-level record labels between the two datasets. Furthermore, it can be argued that a perfect comparison between the datasets is not possible because of their different characteristics.

Distribution of Major Labels

In the MPD (Fig. 5.6a) *Universal* is by far the most dominant top-level class, covering 41.1% of all tracks in the dataset. *Universal* has a difference of 15 percentage points to the next major label, *Sony* with 25.68%, which is followed by *Warner* with 18.97%. Only 14.07% of tracks of the dataset were assigned to *Independent*. We can conclude from this that the three major labels pose a dominance in the Spotify music catalogue during the creation of the MPD (2018), with a combined total coverage of the three majors labels of over 85%.

For the distribution of top-level classes in the LFM-2b (Fig. 5.6b) *Independent* is covering more than half of the tracks with 52.59%. The major labels follow in order *Universal*, *Sony* and *Warner* with 20.3%, 15.24% and 11.86% resp. Here we can see a different dominance of *Independent* record labels, which fits the previously found characteristics of the LFM-2b of being a more diverse dataset with less bias connected to popularity and major labels.

In both distributions we can see the same order of major labels with *Universal* being the most dominant one, *Sony* in the midfield and *Warner* at the end.

Top-Level Class Distribution over sorted Albums

Figure 5.7 shows the trend of albums from major record labels being more popular in both datasets, with *Independent* albums having the highest coverage in the tail of the datasets.

For the MPD (Fig. 5.7a) the high density of album occurrences (Fig. 5.1b) is reflected, where the last bucket contains over 85% of all albums. The number of major label albums is dominant around 90%, before declining exponentially after ca. 50% of all album occurrences are covered (first half of graph). This underlines again the high album density but also shows how *Independent* is most frequent in the long tail of the dataset.

The LFM-2b (Fig. 5.7b) shows a different picture, reflecting its more diverse album distribution (Fig. 5.2b). The same trend of decrease of major labels can be seen the more unpopular the albums get, but the decline is rather linear.

Simpson Index Distribution

Analysing the distribution of major labels per playlist in the MPD using the Simpson Index (SI), we can see a peak of playlists, which have an SI of 0.25 to 0.3 (Fig. 5.8a). This represents an equal distribution of the 4 top-level classes for most playlists. From there we can see an exponential decay regarding diversity with an interesting outlier peak at 1. This outlier represents all playlists, where all tracks belong to the same top-level class.

When analysing the playlists of the MPD with an SI of 0.7 or higher we can draw some interesting conclusions (Table 5.4). The dominance of *Universal* increases the smaller the diversity of the playlist set gets. Compared to the 41.1% over the full dataset, *Universal* covers 60.15% of all tracks of playlists with an SI of 0.7 or higher. This value even increases for higher SIs with a maximum coverage of 72.57% for playlists with an SI of 1. Taking the average playlist length in account, we can see that the length drops from 66.35 for the whole dataset to around 50 for SIs 0.7 to 0.9. The even shorter average length of 36.32 for playlists with an SI of 1 is still more than half the average length for the full dataset. This suggests that these playlists are long enough to derive statistically meaningful conclusions.

This is also supported by the number of playlists in these SI groups, which is shrinking from around 40,000 for an SI ≥ 0.7 to nearly 10,000 for an SI of 1. This means that 1% of the playlists of the dataset consist only of tracks from one top-level class. From the overall dominance of *Universal* in the dataset, combined with an increase of this trend in playlists with small diversity, we can conclude that there is a popularity bias for tracks from the major label *Universal* in the MPD.

An interesting outlier in the analysis of major distribution in SI groups of the SI, is the average playlist length of *Warner*. While the difference of average playlist length between the top-level classes has a rather small fluctuation range, playlists with just tracks from *Warner* (SI of 1) are on average much longer than of other major labels. Data exploration confirmed the hypothesis, that many of these playlists with above average length are collections of movie soundtracks (e.g., *Lord of the Rings*, *Harry Potter* or *Shrek*). This finding must be seen in the context of *Warner* having the smallest share of playlists in this group with 621.

Analysing the listening profiles of the LFM-2b revealed quite different results. The general high coverage of *Independent* in the dataset, with more than 52% of all tracks belonging to this top-label class, was reflected in the absolute dominance in profiles with an SI of 1 (Table 5.3). But also the distribution of Simpson Indexes over all user profiles (Fig. 5.8b) showed a more uniform distribution compared to the MPD. Both datasets share a peak around 0.25 to 0.3, but the decay from this equal distribution is almost linear. Also, the striking outlier at SI 1 is nearly missing, with just 14 profiles being in this bucket. This is in accordance with the low density of tracks in the LFM-2b, where many tracks only appear once in the dataset (Fig. 5.2a). The median of how often a track appears in a user profile with 1 means that more than half of the tracks in an average listening history only appear once.

For profiles with tracks filtered to appear at least 3 times, we can see a further decrease of major record label coverage in favour of *Independent* for increased SIs (Fig. 5.9b).

Research Question 2

RQ2 (*"What kind of biases in music datasets can we identify regarding record labels?"*) must be answered separately for both datasets:

For the MPD a strong dominance of the three major labels was found, where 85% of the tracks of the dataset belong to these labels. We can therefore conclude that there is a bias in the dataset towards major labels and away from independent record labels. There is a specific bias towards the major label *Universal* which has overall the biggest coverage in the full MPD (41%), combined with an increased coverage of this major label in playlists with a high Simpson Index.

The music catalogue covered by the LFM-2b is strongly focused on independent record labels, painting a completely different picture than the MPD. 52% of the dataset belongs to *Independent*, including all user profiles with an Simpson Index of 1. From this we can derive a bias towards *Independent* for the LFM-2b.

For both datasets the same order of major label coverage (Universal > Sony > Warner) was found, which can be seen as a shared bias of both dataset and as an insight in today's music market hierarchy.

8.3 Recommendation Simulation

The discussion of the experiments on offline recommender system performacne and feedback loop simulation will answer RQ3 *"Do the identified biases impact the recommendations of a recommender system?"*

This includes the evaluation the two hypothesis H1 (*Including the record label information will improve the offline evaluation performance of recommender systems*) and H2 (*Over iterations of recommendations a decrease in diversity of record labels can be observed.*).

8.3.1 Offline Recommender System Performance Comparison

As the experiments on performance inprovements are based on the challenge set of the Recommender System Challenge 2018 (RecSys18), first the analysis of this set will be discussed, followed by the benchmark evaluation.

Challenge Set Analysis

The distribution of Simpson Indexes (SI) over the challenge playlist set (Fig. 7.1a and 7.1b) showed a trend similar to the whole MPD (Fig. 5.8a). There is an exponential decay from the peak of top-class equality above 0.3, but there are much more outliers, especially for the playlists with a minimum of 10 tracks. The increase in outliers could be explained with the much smaller sample of playlists (6,000 or 4,000 compared to 1,000,000) and incomplete playlists. The biggest outlier peak is again around an SI of 1, which is identical to the full MPD.

The distribution of top-level classes (Fig. 7.1c and 7.1d) confirms once more the dominance of *Universal*. Increasing the SI to 1 shows a different picture with a decrease of *Universal* coverage in favour of the other top-level classes, especially *Sony* and *Independent*. This denotes a more equal distribution of top-level classes for an SI of 1, but again the low sample size must be taken in account. Only 172 playlists with minimum length 10 have an SI of 1, for a length threshold of 25 there are only 46 such playlists. Compared to the ca. 10,000 playlists with SI 1 of the full MPD (ca. 1%), the 172 playlists with minimum length 10 are a significant increase compared to the challenge set (1.7%), which can again be explained by the naturally short length of many of the challenge playlists (only 5 or 10 tracks), whereas the median playlist length of the full MPD is 49.

An interesting finding of the challenge analysis is the very low coverage of *Warner* for higher SI thresholds in the subset of playlists with minimum length 25 (Fig. 7.1d). While *Warner* has also a lower coverage than *Independent* for higher SI thresholds in the full MPD (Fig. 5.8a), the difference is much more significant in the challenge set. If this reflects decisions in the challenge set creation or if it can be explained by the low sample size cannot be answered at this point.

Re-Ranking Evaluation

The plain KAENEN submission with no re-ranking (row marked with single asterisk in Table 7.1), performed similar to the original result of RecSys18' main track: 0.2091, 2.0540 and 0.3747 compared to 0.2074, 2.0603 and 0.3723 in the experiment, for *R-prec*, *clicks* and *NDCG* resp. The plain KAENEN ranked straight away on fourth place on the AIcrowd leaderboard[1].

For the further analysis we will focus solely on the *NDCG*, as *R-prec* is invariant to the order of tracks and *clicks* is only available with 4 decimal places on AIcrowd and therefore not showing much difference.

It was possible to improve the performance slightly by re-ranking the first 10 to 30 tracks of playlists with minimum length of 25 and an SI above 0.8. The best result could be achieved by re-ranking the first 25 tracks of the submission of the described set of playlists. Only 93 of the 10,000 playlists were touched from this approach, which explains the very small increase in performance. This is equal to a maximum increase by about 1%, which could be motivated to interpret the performance gain for the best configuration (first row) as an increase of around 0.0266%.

Lowering the SI threshold or the minimum length decreased the *NDCG*. Re-ranking a higher number of tracks (50 or even 500) had the same result. Especially the approaches of just using the dominant major for each challenge playlist (last four rows) performed quite bad.

Hypothesis 1

Based on this experiment we can conclude that including the record label information will improve the offline evaluation performance of recommender systems. Although the difference is very subtle, record label information can be used to capture the distribution and biases of a ground truth of outliers with an already low diversity of record labels. But in a competition like the RecSys18 such small differences make a difference and could be decisive for winning a better place in the final leaderboard. Whether such information also results in improved recommendations with regards to user satisfaction needs to be analysed separately and is beyond the scope of this thesis.

[1] https://www.aicrowd.com/challenges/spotify-million-playlist-dataset-challenge/leaderboards

8.3.2 Feedback Loop Evaluation

MPD

The results of the feedback loop experiments on the MPD must be discussed in consideration on the limitations of the described setup, only recommending tracks to 1% of the one million playlists.

In Table 7.2 we can see that the distribution of major labels in the ALS recommendations are very similar to the overall distribution of major labels in the MPD (Fig. 5.6a) and that there are no significant changes between the two iterations.

When looking at the reduced setup the *First* position of a label in the recommended tracks (Fig. 7.2a) and the percentage of *Recommended* tracks (Fig. 7.2b) is very similar to the order for the full MPD (Table 7.2). No trends are visible over the 20 iterations, concluding in no detected feedback loops for the MPD with the described experimental setup.

However, given these limitations in the conducted experiments, we refrain from concluding that feedback loops do not exist for the MPD in this scenario.

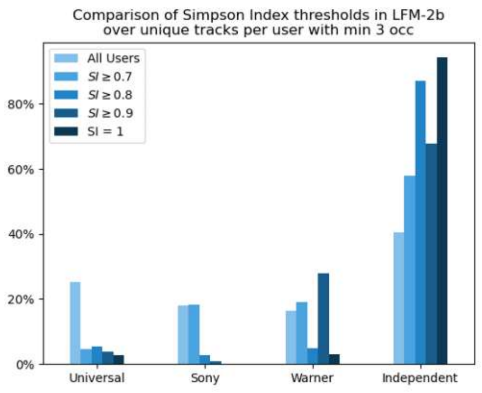
LFM-2b

For the LfM-2b the results turned out to be more interesting. When comparing the *Recommended* metric (Fig. 7.3b), presenting the coverage of a major in the first 100 recommendations, from the feedback loop experiment of the LFM-2b, we can see a different coverage than in the LFM-2b subset of tracks with Spotify URI on which the crawler ran (Fig. 5.6b).

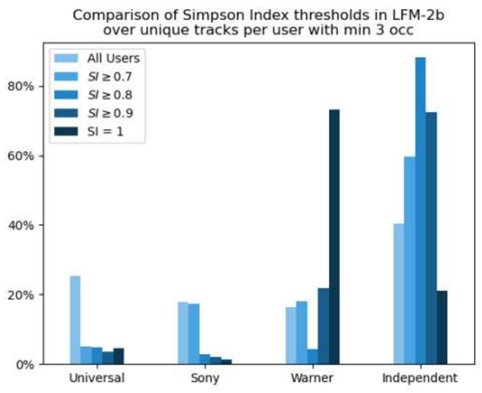
In iteration 0, *Independent* claims the highest number of tracks covered with around 31.5% while *Universal* is just under this value with 30%. This is a much higher representation in the recommendations than the share of tracks from *Universal* in the LFM-2b subset which was just 20.3% compared to 52.6% from *Independent* (compare Fig. 5.6b). Over the first three iterations the *Recommended* value of *Independent* and *Universal* aligns to just below 31%.

Like *Universal*, the *Recommended* values of *Sony* and *Warner* are also much higher than their representative values in the overall track coverage of the LFM-2b subset, but with switched positions (*Warner* over *Sony*). While *Sony* had 15.2% coverage in the LFM-2b subset, *Warner* was just covering 11.9% of the tracks. Now in iteration 0 of the feedback loop experiment they cover 19% (*Sony*) and 21%(*Warner*) respectively. Different to *Independent* and *Universal*, *Sony* and *Warner* do not converge during the first three iterations, but even distance themselves.

We can conclude from the *Recommended* metric, that while all three majors (*Universal*, *Sony* and *Warner*) are over-represented in the recommendations for the full LFM-2b compared to their coverage in the LFM-2b subset. For *Universal* and *Warner* this over-representation is quite strong with each around 10% above their coverage in the



(a) Min 1 occurrence



(b) Min 3 occurrences

Figure 8.1: LFM-2b: Distribution of top-level classes when duplicates are not removed.

subset. The playlist continuation experiment for feedback loop simulation was the only time during the research of this project when *Warner* succeeded *Sony* in coverage. A possible reason for this might be the structure of the playlists in the LFM-2b, where *Warner* seems to claim a more important position than *Sony*.

When looking at the *First* metric (Fig. 7.3a), representing the first position of the label class in the recommendations, we can see the labels in the order: *Independent*, *Universal*, *Warner* and *Sony*. Remember that a lower value in this metric is better. Like with the *Recommended* metric, we have *Warner* in a more dominant position than *Sony*. Interestingly there is a trend of *Warner* improving its position from 13 to around 11.5 in the first six iterations before stabilizing. All other three major classes (*Independent*, *Universal* and *Sony*) keep their *First* position relatively, with a minimal decline.

Overall, we can see a bias of over-representation of *Universal*, *Warner* and *Sony* in the recommendations of the feedback loop experiment. This trend is the strongest for *Warner* which is even recommended in an earlier position over iterations, marking a moderate feedback loop.

The previous analysis of the LFM-2b user profiles (Fig. 5.8b and 5.9b) only considered profiles with duplicate tracks removed. But the feedback loop simulation is not removing duplicate tracks, adopting the LFM-2b profiles just as they are.

Analysing the distribution of top-level classes in the LFM-2b when duplicates are not removed (Fig. 8.1a) showed an interesting peak for the *Warner* coverage in profiles with an Simpson Index above 0.9. There are 69 profiles with an SI ≥ 0.9 of which 57 belong to *Independent* (meaning that *Independent* is the dominant top-level class) and 4 to *Warner*. The high coverage of *Warner* can be explained by the average length of such profiles. *Warner* profiles with SI ≥ 0.9 have on average 5290 tracks, compared to 957 tracks in *Independent* profiles. For a minimum occurrence restriction of 1, only 15 profiles have an SI of 1.

Introducing a minimum occurrence restriction of 3 per tracks without removing duplicates (Fig. 8.1b) intensifies the trend of high *Warner* coverage in profiles with a high SI. There are 272 profiles with a minimum SI of 0.9 and 162 profiles with an SI of 1. Profiles with an SI of 1 have an average length of 231 tracks, 106 for *Independent* and 1490 for *Warner*.

Although it is obvious, that not removing duplicates with a threshold of multiple occurrences per track increases the number of profiles with a high SI, this finding may allow conclusions on why a feedback loop could be simulated for *Warner*. This is based, as previously explained, on the fact, that the simulation setup is also not removing duplicates. A possible explanation for the much higher average number of tracks in profiles with a high SI and *Warner* dominance, is that Last.fm has a bias of recommending *Warner* tracks if a user is on autoplay, resulting in a long listening history. Further research is necessary to answer this hypothesis thoroughly.

Hypothesis 2

For the MPD we must reject the second hypothesis that *over iterations of recommendations a decrease in diversity of record labels can be observed*, keeping the restrictions of the experimental setup for this dataset in mind. For the LFM-2b also no significant decrease in diversity could be observed. The discovered feedback loops rather lead to a change in the distribution of record labels over longitudinal recommendations.

8.3.3 Conclusion

Combining the findings from the offline recommendation experiments and the feedback simulation, we can conclude that the record label biases in both datasets have an impact on the recommendations of recommender systems.

For the MPD the record label information helped to improve the performance of the KAENEN submission slightly by simply re-ranking the recommended tracks of edge case playlists with low major label diversity. No feedback loops could be identified in the restricted setup of recommending songs for only 1% of all playlists of the dataset.

For the LFM-2b an over-representation of major labels in the recommendations was discovered with a feedback loop for *Warner*.

CHAPTER 9

Ethical Statement

This chapter covers the ethical thoughts that came up during the research project. Reflecting upon these openly extends the depth of understanding the process of developing the label crawler and the resulting machine learning experiments. Additionally, this should influence future research positively, as low-level assumptions are explicitly formulated.

Overall, there is the ethical context of transparency regarding biases, making underlying tendencies in recommendation processes visible. This research field is already a broad one and will become more and more important in the future where we will find variants of recommender systems in widespread use. There seems to be no interest of big music streaming providers to reveal their recommendation algorithms in full disclosure. The precise way of how tracks are recommended is part of the marketing strategy of these companies, used to differentiate themselves from competitors. While this makes sense from a business standpoint, it raises multiple ethical questions regarding the relation of representation of stakeholders and user base.

9.1 Risk and Benefits to Stakeholders

The three major label companies: *Universal Music Group*, *Sony Music Entertainment* and *Warner Records* form an oligopolistic block in today's music market with a dominant position worldwide. We could see this confirmed multiple times during the research project, especially when studying the MPD, the only playlist dataset ever released publicly by Spotify.

Record labels are the stakeholders in the recommendation process of music, using companies like Spotify to distribute their goods. These companies have interests of which we must assume that they are reflected in the recommendation algorithms in use [JA16, AE17]. De-biasing such algorithms regarding the distribution of major labels or even just publishing detected biases will touch the interests of these companies. We should be aware of these interests when trying to uncover such biases as they might be hidden intentionally or covered by bureaucratic layers.

Although this research proved a slight influence of label information on recommendations in regards to performance (Table 7.1) and diversity in long-term recommendation simulations (Fig. 7.3), Spotify decided not to include label or copyright information of any kind for the RecSys Challenge 2018 (RecSys18). This information is included in the standard album information when requesting an album from the Spotify-API. But it is only fair to keep in mind, that the challenge is focusing on playlist names and tracks, while label information is primary album related. Also, we have to give credit for the creative track of the RecSys18, which endorsed the use of data outside the MPD which would include label information.

9.2 Data Collection and Privacy

The MPD just includes public playlists from users from the USA, as already described in the dataset chapter. This might be explained with the different data protection laws in the USA compared to the EU. The EU's General Data Protection Regulation (GDPR) protects a playlist, published with the purpose of sharing music could, of being automatically used for different reasons - like a public coding challenge on improving playlist continuation.

While this seems to have been the easy way out for Spotify on a legal base, it introduces a very strong bias when creating the MPD. Just including the listening behaviour of US-citizens entails a possible restriction on using the algorithms of the participants of the challenge for global or e.g., European music datasets. Rightfully all details on the MPD creation were covered in the associated README file, but it would have improved the understanding of the dataset when the chosen actions would have been explained thorough.

In [PVZ22] Prey et. al compare Spotify's ever changing music catalogue to Herclitus' river. *"'Everything flows', was the basis of his philosophy. 'You cannot step twice into the same river; for fresh waters are ever flowing in upon you.'"*[Woo21] Although this metaphor is meant to describe the world in general, it is an accurate description of the platform archiving policy: *"(...) one can never access the same platform twice. Constantly in flux from one moment to the next, Spotify churns in motion as tracks, albums and playlists are added, deleted and reconfigured. This presents researchers with a conundrum: Any analysis of the platform itself will necessarily be an analysis of the present. Spotify does not provide an archive of songs that were uploaded in 2009, or playlists that disappeared in 2019."*[PVZ22].

This leads to the conclusion that all findings of the MPD and LFM-2b can just be seen as a snapshot of the platform in the period of the research. That is because the preprocessing step of the label crawler is gathering all its low-level record label and copyright information from the Spotify-API. For the sake of reproducibility, it would be necessary to keep an archive of Spotify's data stock. To the best of our knowledge no such repository exists currently. The final label assignments from low-level record label to top level class for both datasets are available in the project's repository to allow further research on this snapshot.

9.3 Other Ethical Concerns

Music streaming platforms started as sheer repositories of music items, promoting their ever-growing catalogue of music. As of 2022 Spotify and Apple Music host around 80 to 90 million tracks[1].

This sheer amount of available music makes music recommendation not only a commodity for music discovery but also a necessity. But this tendency of platforms from the stewardship of open exploration towards taste shaping of their large user bases can be seen critically. *"For some, the promise of streaming is that it frees music distribution and discovery from the stranglehold of the major labels – giving more space to independent artists and independent record labels as listeners explore the 'long tail' of a platform's catalogue."*[PVZ22] But the three major record labels *Universal*, *Sony* and *Warner* have a dominant position over independent labels and exert their power to influence and expand their shares and presence on music streaming platforms [HM15, HOSB21].

The research on targeted promotion of platform-owned playlists [PVZ22] or the over-representation of major label content in recommendations as showed in this project, extends the area of responsibility of big music streaming platforms. The increasing importance of such platforms brings an active role in potentially shaping the cultural identity of whole generations. This consequentially raises ethical questions on how such promotion and recommendations happen. We hope that the research of this project helps to increase the transparency of the functioning of these algorithms and will allow future research upon the documented findings.

[1] cf. https://newsroom.spotify.com/company-info/, https://www.apple.com/apple-music/

CHAPTER 10

Conclusion

To wrap up this research project, the most significant findings are presented, followed by suggestions for future research.

10.1 Findings

We have presented results and efforts into the direction of investigating the impact and role of major record labels in music recommendation. Starting with two datasets of different origin, namely MPD as a playlist dataset and LFM-2b as a listening dataset, we could show very different characteristics and biases wrt. record label distribution.

For the MPD a strong dominance of the three major labels *Universal*, *Sony* and *Warner* was found, which cover over 85% of the dataset. This dominance is emphasised for *Universal* which covers over 41% on its own. Also, non-diverse outliers could be identified in the MPD, where some of these playlists to contain collections of movie soundtracks, thus exhibiting diverse artists, while being published under the same major label.

Although *Independent* is predominant in the LFM-2b with 52% coverage, both datasets showed the same order of major labels with *Universal* in the front, *Sony* in the midfield and *Warner* in the end.

Using label information has shown a potential to improve the performance of recommender systems when re-ranking edge cases.

With regard to the effects of recommender systems on record label distribution (i.e., one type of item providers), we could identify first feedback loop effects. Despite the dominance of independent labels in the LFM-2b set, major labels are over-represented in the recommendation process, with Universal's and Warner's over-representation even being further amplified over iterations. Further analyses need to be also linked to effects of popularity.

10.2 Future Research

Re-using the presented label crawler algorithm on other datasets could help to broaden the understanding of record label hierarchy. From that an improved way to visualize the complex hierarchy between low-level record labels, bigger distributor companies and major labels will present an additional challenge. The potential size of this network, including cycles and multiple connection between single nodes motivates an interactive visualization, which could lead to interesting findings through ways of data exploration.

Another future research goal is to further investigate possibilities to improve recommender system performance with the gathered label information. The same order of major labels in both datasets hints towards a possible useful application of the label data. But the proposed setup of re-ranking an existing recommendation limits the usage of label information.

List of Figures

List of Tables

Bibliography

[AE17] Himan Abdollahpouri and Steve Essinger. Multiple stakeholders in music recommender systems. arXiv, 2017. doi:10.48550/ARXIV.1708.00120.

[AM20] Himan Abdollahpouri and Masoud Mansoury. Multi-sided Exposure Bias in Recommendation. arXiv, 2020. arXiv:2006.15772.

[AW21] Luis Aguiar and Joel Waldfogel. Platforms, power, and promotion: Evidence from spotify playlists. volume 69, pages 653–691. Wiley Online Library, 2021. doi:10.1111/joie.12263.

[CG18] Sam Corbett-Davies and Sharad Goel. The measure and mismeasure of fairness: A critical review of fair machine learning. volume abs/1808.00023. arXiv, 2018. arXiv:1808.00023.

[CLSZ18] Ching-Wei Chen, Paul Lamere, Markus Schedl, and Hamed Zamani. Recsys Challenge 2018: Automatic Music Playlist Continuation. In *RecSys '18: Proceedings of the 12th ACM Conference on Recommender Systems*, RecSys '18, page 527–528, New York, NY, USA, 2018. Association for Computing Machinery. doi:10.1145/3240323.3240342.

[CR18] Alexandra Chouldechova and Aaron Roth. The Frontiers of Fairness in Machine Learning. arXiv, 2018. arXiv:1810.08810.

[DB22] Karlijn Dinnissen and Christine Bauer. Fairness in music recommender systems: A stakeholder-centered mini review. In *Frontiers in Big Data*, volume 5, July 2022. doi:10.3389/fdata.2022.913608.

[EBD19] Michael D. Ekstrand, Robin Burke, and Fernando Diaz. Fairness and Discrimination in Retrieval and Recommendation. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR'19, page 1403–1404, New York, NY, USA, 2019. Association for Computing Machinery. doi:10.1145/3331184.3331380.

[FBSY19] A. Ferraro, D. Bogdanov, X. Serra, and J Yoon. Artist and Style Exposure Bias in Collaborative Filtering Based Music Recommendations. arXiv, November 2019. doi:10.48550/ARXIV.1911.04827.

[FJK+20] A. Ferraro, J. Jeon, B. Kim, X. Serra, and D. Bogdanov. Artist biases in collaborative filtering for music recommendation. In *Proceedings of the 37 th International Conference on Machine Learning*, Vienna, Austria, July 2020. URL: https://repositori.upf.edu/handle/10230/45185.

[FJS20] A. Ferraro, D. Jennach, and X. Serra. Exploring Longitudinal Effects of Session-based Recommendations. ACM, September 2020. doi:10.1145/3383313.3412213.

[FSB21] A. Ferraro, X. Serra, and C. Bauer. Break the Loop: Gender Imbalance in Music Recommenders. In *CHIIR '21: Proceedings of the 2021 Conference on Human Information Interaction and Retrieval*, Canberra ACT, Australia, March 2021. Association for Computing Machinery. doi:10.1145/3406522.3446033.

[GF16] Clare Garvie and Jonathan Frankle. Facial-recognition software might have a racial bias problem. Atlantic, April 2016. Retrieved June 9 2022 from https://www.theatlantic.com/technology/archive/2016/04/the-underlying-bias-of-facial-recognition-systems/476991/.

[HKV08] Yifan Hu, Yehuda Koren, and Chris Volinsky. Collaborative Filtering for Implicit Feedback Datasets. In *2008 Eighth IEEE International Conference on Data Mining*, pages 263–272, 2008. doi:10.1109/ICDM.2008.22.

[HM15] David Hesmondhalgh and Leslie Meier. Popular music, independence and the concept of the alternative in contemporary capitalism. In *Bennett, J, (ed.) Media Independence: Working with Freedom or Working for Free?*, Routledge Research in Cultural and Media Studies, January 2015.

[HOSB21] David Hesmondhalgh, Richard Osborne, Hyojung Sun, and Kenny Barr. Music Creators' Earnings in the Digital Era. Intellectual Property Office Research Paper Forthcoming, September 2021. doi:10.2139/ssrn.4089749.

[IFP22] IFPI. 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-music-revenues-grew-18-5-in-2021/.

[JA16] Dietmar Jannach and Gediminas Adomavicius. Recommendations with a purpose. In *Proceedings of the 10th ACM Conference on Recommender Systems*, RecSys '16, page 7–10, New York, NY, USA, 2016. Association for Computing Machinery. doi:10.1145/2959100.2959186.

[KH19] Peter Knees and Moritz Hübler. Towards Uncovering Dataset Biases: Investigating Record Label Diversity in Music Playlists. In *Proceedings of the 1st Workshop on Designing Human-Centric Music Information Research*

Systems, pages 19–22. Zenodo, November 2019. doi:10.5281/zenodo.3547417.

[KM11] Peter Knees and Schedl Markus. Towards semantic music information extraction from the Web using rule patterns and supervised learning. In *Workshop on Music Recommendation and Discovery*, pages 18–25. ACM RecSys, October 2011.

[KO76] B.J. Kostrewski and Charles Oppenheim. Some ethical and political implications of theoretical research in information science. In *Proceedings of the ASIS Annual Meeting*, volume 13, October 1976. doi:10.1177/016555157900100505.

[KSL19] Dominik Kowald, Markus Schedl, and Elisabeth Lex. The unfairness of popularity bias in music recommendation: A reproducibility study. arXiv, 2019. doi:10.48550/ARXIV.1912.04696.

[LCK16] Jin Ha Lee, Hyerim Cho, and Yea-Seul Kim. Users' music information needs and behaviors: Design implications for music information retrieval systems. In *Journal of the Association for Information Science and Technology*, volume 67, pages 1301–1330, 2016. doi:https://doi.org/10.1002/asi.23471.

[LGHT+20] Min Kyung Lee, Nina Grgić-Hlača, Michael Carl Tschantz, Reuben Binns, Adrian Weller, Michelle Carney, and Kori Inkpen. Human-Centered Approaches to Fair and Responsible AI. In *Extended Abstracts of the 2020 CHI Conference on Human Factors in Computing Systems*, CHI EA '20, page 1–8, New York, NY, USA, 2020. Association for Computing Machinery. doi:10.1145/3334480.3375158.

[LKLJ18] Malte Ludewig, Iman Kamehkhosh, Nick Landia, and Dietmar Jannach. Effective Nearest-Neighbor Music Recommendations. In *Proceedings of the ACM Recommender Systems Challenge 2018*, RecSys'18, New York, NY, USA, 2018. Association for Computing Machinery. doi:10.1145/3267471.3267474.

[MAP+20] 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 Knowledge Management*, CIKM '20, page 2145–2148, New York, NY, USA, 2020. Association for Computing Machinery. doi:10.1145/3340531.3412152.

[MMB+18] Rishabh Mehrotra, James McInerney, Hugues Bouchard, Mounia Lalmas, and Fernando Diaz. Towards a Fair Marketplace: Counterfactual Evaluation of the Trade-off between Relevance, Fairness 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. doi:10.1145/3269206.3272027.

[MP15] Jeremy Wade Morris and Devon Powers. Control, curation and musical experience in streaming music services. *Creative Industries Journal*, 8(2):106–122, 2015. doi:10.1080/17510694.2015.1090222.

[MRPC+21] 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. doi:https://doi.org/10.1016/j.ipm.2021.102666.

[Naj09] M. Najork. Web crawler architecture. pages 3462–3465. Springer, 2009. doi:10.1007/978-1-4899-7993-3_457-3.

[Nob18] Safiya Umoja Noble. *Algorithms of Oppression: How Search Engines Reinforce Racism*. NYU Press, 2018.

[Ora14] S. Oramas. Harvesting and Structuring Social Data in Music Information Retrieval. In *The Semantic Web: Trends and Challenges. ESWC*, pages 817–826, Cham, 2014. doi:10.1007/978-3-319-07443-6_55.

[PT08] Yoon-Joo Park and Alexander Tuzhilin. The long tail of recommender systems and how to leverage it. RecSys '08, page 11–18, New York, NY, USA, 2008. Association for Computing Machinery. doi:10.1145/1454008.1454012.

[PVZ22] 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. doi:10.1080/1369118X.2020.1761859.

[PZS15] Martin Pichl, Eva Zangerle, and Günther Specht. Towards a context-aware music recommendation approach: What is hidden in the playlist name? In *2015 IEEE International Conference on Data Mining Workshop (ICDMW)*, pages 1360–1365, 2015. doi:10.1109/ICDMW.2015.145.

[SBL+22] Markus Schedl, Stefan Brandl, Oleg Lesota, Emilia Parada-Cabaleiro, David Penz, and Navid Rekabsaz. Lfm-2b: A dataset of enriched music listening events for recommender systems research and fairness analysis. In *ACM SIGIR Conference on Human Information Interaction and Retrieval*, CHIIR '22, page 337–341, New York, NY, USA, 2022. Association for Computing Machinery. doi:10.1145/3498366.3505791.

[SC18] United States Securities and Exchange Commission. Form f-1 registration statement, spotify technology, s.a., March 2018. Retrieved May 17 2022 from https://www.sec.gov/Archives/edgar/data/1639920/000119312518092759/d494294df1a.htm.

[Sim49] Edward H. Simpson. Measurement of diversity. *Nature*, 163:688, 1949. doi:10.1038/163688a0.

[SKM+15] Markus Schedl, Peter Knees, Brian McFee, Dmitry Bogdanov, and Marius Kaminskas. *Music Recommender Systems*, pages 453–492. Springer US, Boston, MA, 2015. doi:10.1007/978-1-4899-7637-6_13.

[SKMB22] 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. doi:10.1007/978-1-0716-2197-4_24.

[SPGC20] Dougal Shakespeare, Lorenzo Porcaro, Emilia G'omez, and Carlos Castillo. Exploring artist gender bias in music recommendation. *ArXiv*, 2020. arXiv:2009.01715.

[Ste11] Harald Steck. Item popularity and recommendation accuracy. In *Proceedings of the Fifth ACM Conference on Recommender Systems*, RecSys '11, page 125–132, New York, NY, USA, 2011. Association for Computing Machinery. doi:10.1145/2043932.2043957.

[SZC+18] Markus Schedl, Hamed Zamani, Ching-Wei Chen, Yashar Deldjoo, and Mehdi Elahi. Current challenges and visions in music recommender systems research. In *International Journal of Multimedia Information Retrieval*. Springer Science and Business Media LLC, apr 2018. arXiv:1710.03208.

[Woo21] Alan Woods. *The History of Philosophy: A Marxist Perspective*, page 64. Wellred Books, September 2021.