2. Company and Context

To understand the steps needed to conduct music tagging at scale and more specifically why this music tagging needs to be conducted, context must be provided, about both the company, Groover, and the need that the internship aims to fill. This sections focuses on the presentation of Groover, the team in which the intern was included, and how music tagging aims to improve the existing recommender system.

2.1. Company

2.1.1. Groover's mission and goals

Groover, founded in 2018, is a company which aims to provide the service to artist of sending their music to music industry professionals. By influencers, we mean : Spotify playlist curators, sound engineers, Youtube channels, music experts, soundcloud curators, and many more. The main mission of Groover is to **Empower all independent artist to get their music heard and accelerate their careers**, with the key rule to serve artists everyday.

Recently, Groover has been putting efforts into R&D and diversifying their services :

- Providing a curated list of groover-favorite artists with **Groover Obsessions**, with new releases being promoted and artists being followed by the Groover team
- Further helping artists accelerate their careers with **Groover Masterclass**, which aims to instruct artists about the music industry and give them insights into the workings of getting their careers off the ground

Groover has recently been gaining a lot of traction with important partnerships, prevalent ads, and artists kickstarting their career thanks to their services. Their growing market share in the US and globally, with influencers, artists and influencer ambassadors in over 20 countries, is key to their goal of being the go-to platform for artists to develop their careers.

Business Model

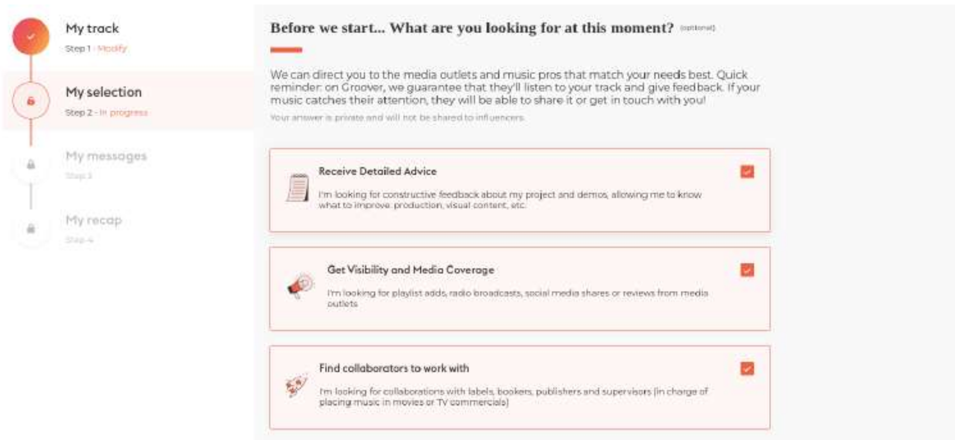
On the Groover Website, artists can start what are called "campaigns" and send one of their tracks to recommended curators within an alloted budget. Each curator has a certain cost to send the track to. The following figure (2.1) shows the services proposed by Groover for an artist after providing basic information about their track, in line with their goals : **Receive advice, get media coverage and exposure, and find collaborators**.

Following this step, the recommender algorithm, which is at the core of the provided service here, selects a list of relevant curators for the artist based on the artist's music, the curators's taste, and past behaviours of both the artist and the curator **if the artist is not new to the platform**. Figure 2.2 shows the recommendations for a test track.

Of course, the better the recomendations, the better the chance of the artist getting something back from his campaign, be it consistent and relevant feedback, getting posted to a spotify playlist, or scoring a collaboration. And the better artist satisfaction, the closer Groover is to its mission and the faster the growth. So, the recommender system is an important piece of the ecosystem at Groover, and this internship focuses on exploiting audio data and signal processing to improve it further.

2.1.2. Organization of the Groover team

To understand how the data team, which I joined, articulates around the rest of the Groover Team, the following figure (2.3) shows the basic organization of the team.



My track
Step 1 · Modify

My selection
Step 2 - In progress

My messages
Step 3

My recap
Step 4

Before we start... What are you looking for at this moment? (optional)

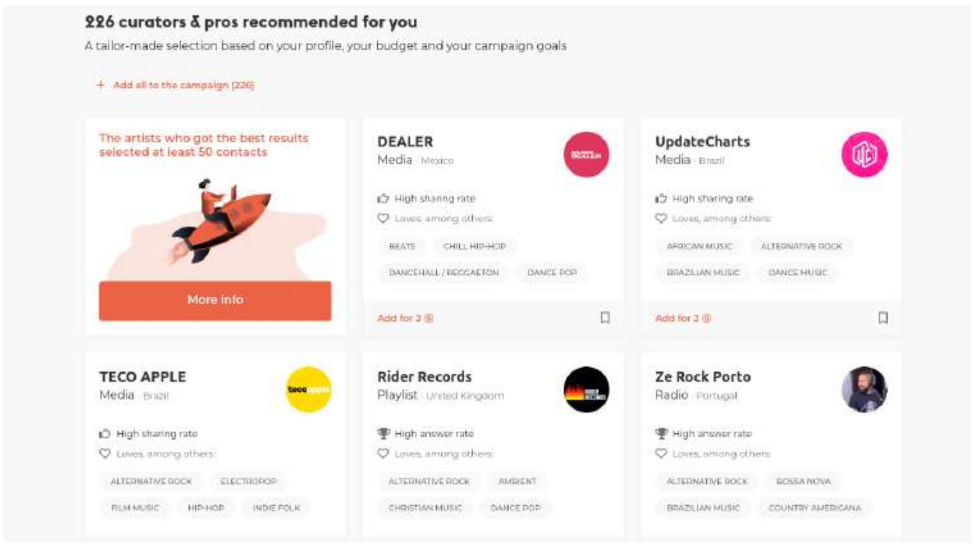
We can direct you to the media outlets and music pros that match your needs best. Quick reminder: on Groover, we guarantee that they'll listen to your track and give feedback. If your music catches their attention, they will be able to share it or get in touch with you!
Your answer is private and will not be shared to influencers.

Receive Detailed Advice ✓
I'm looking for constructive feedback about my project and demos, allowing me to know what to improve: production, visual content, etc.

Get Visibility and Media Coverage ✓
I'm looking for playlist adds, radio broadcasts, social media shares or reviews from media outlets

Find collaborators to work with ✓
I'm looking for collaborations with labels, bookers, publishers and supervisors (in charge of placing music in movies or TV commercials)

Figure 2.1.: Choosing campaign goals on Groover.co



226 curators & pros recommended for you
A tailor-made selection based on your profile, your budget and your campaign goals

+ Add all to the campaign (226)

The artists who got the best results selected at least 50 contacts
More info

DEALER
Media · Mexico
High sharing rate
Loves, among others:
BEATS · CHILL HIP-HOP · DANCEHALL / REGGAETON · DANCE POP
Add for 2

UpdateCharts
Media · Brazil
High sharing rate
Loves, among others:
AFRICAN MUSIC · ALTERNATIVE ROCK · BRAZILIAN MUSIC · DANCE MUSIC
Add for 2

TECO APPLE
Media · Brazil
High sharing rate
Loves, among others:
ALTERNATIVE ROCK · ELECTROPOP · FILM MUSIC · HIP-HOP · INDIE FOLK

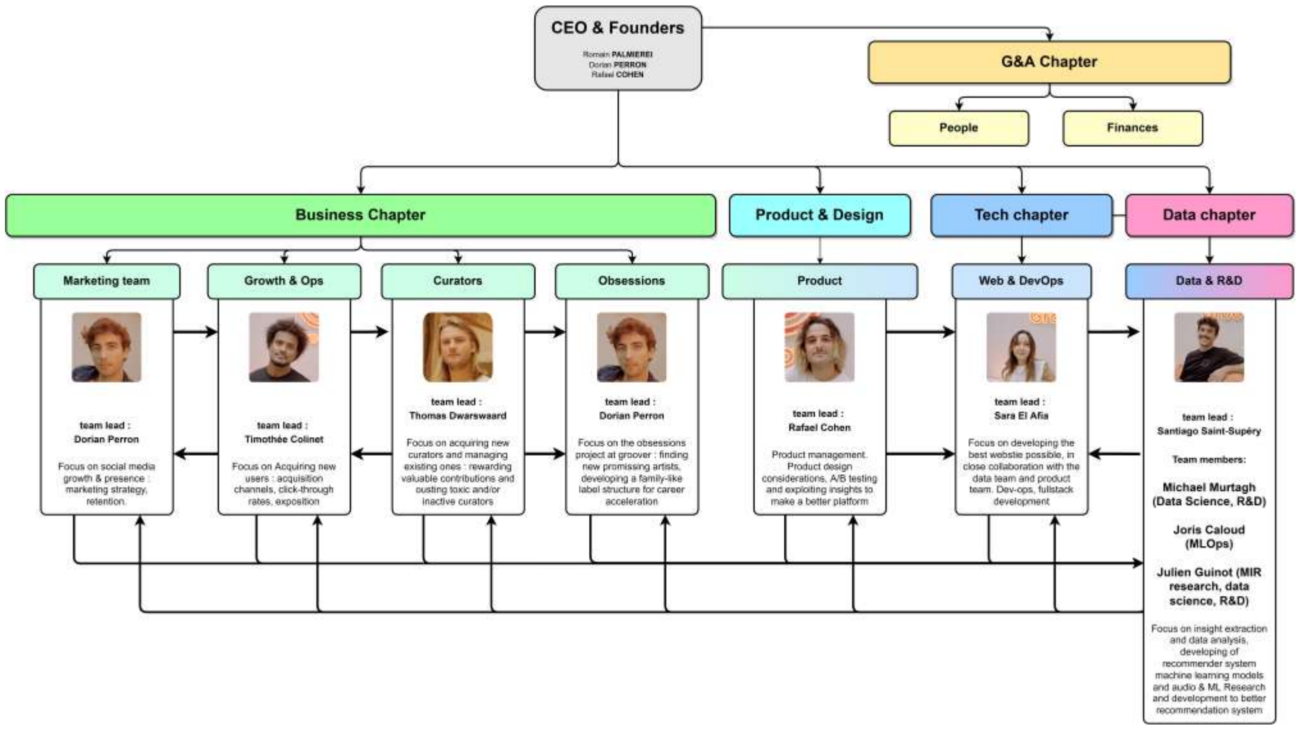
Rider Records
Playlist · United Kingdom
High answer rate
Loves, among others:
ALTERNATIVE ROCK · AMBIENT · CHRISTIAN MUSIC · DANCE POP

Ze Rock Porto
Radio · Portugal
High answer rate
Loves, among others:
ALTERNATIVE ROCK · BOSSA NOVA · BRAZILIAN MUSIC · COUNTRY AMERICANA

Figure 2.2.: Recommended influencers for a test track on Groover.co

The team is divided into five main areas : **Business, Tech, Product design, Data & G&A**. We do not go into detail for members of each team except for ours, the data team. Of course, Groover is a growing company, so transversal interaction between roles is highly possible, and no structure is frozen (One of the reasons this company seemed attractive to me). However it is important to understand how all of these teams relate to the data team, so we specify each of the teams' global roles.

- **The marketing team** focuses on promoting Groover on social media and to grow the presence and platform of the company. *The data team ingests data and provides insight to this team, such as ad click-through rate, user trends and growth, and campaign efficacy, per their request*
- **The growth team** focuses on growing the user base of the Groover platform through various media. Again, per their request, the data team provides insight on all the ingested data by creating visualizations.
- **The curators team** manages the base of curators on the platform (curators is the vernacular term for influencers). Operating on a per-geographic region basis, they recruit new curators and manage the existing ones. They work in close collaboration with the data team to not only request metrics and insights on curators, but also to create ML models to flag unsatisfactory curators, who provide bad or abusive feedback, or do not deliver on provided feedback.



Focus on social media growth & presence : marketing strategy, retention.

Focus on Acquiring new users : acquisition channels, click-through rates, exposition

Focus on acquiring new curators and managing existing ones : rewarding valuable contributions and ousting toxic and/or inactive curators

Focus on the obsessions project at groover : finding new promising artists, developing a family-like label structure for career acceleration

Product management. Product design considerations, A/B testing and exploiting insights to make a better platform

Focus on developing the best webstie possible, in close collaboration with the data team and product team. Dev-ops, fullstack development

Focus on insight extraction and data analysis, developing of recommender system machine learning models and audio & ML Research and development to better recommendation system

Figure 2.3.: Organization of the groover team

- **Obsessions team** focuses on recruiting new artists for Groover's label-like acceleration service, and dealing with events which showcase these artists. They interact with the data team to request recommendations for artists applying to these events (same as the online recommendation pipeline), and, as always, to request data insights and visualization

Moving on to the technical chapter, the teams are:

- **The product team** is halfway between marketing and tech. While it focuses on such metrics as click-through and activation funnel to analyse the components and steps in the website, it also delves into product design and management to create the best platform possible. Interactions with the data team include insights on A/B testing, and other metrics such as the activation funnel.
- **The web & devOps team** works in close collaboration with the data team on data ingestion and database structure. New data acquisition goes through the back-end team within the web&devops team. They focus on deployment of the website, front-end development in line with the product teams' asks, and continuous development/integration pipelines.

The data team

Finally, the data team, in which this internship took place, is in charge of creating data pipelines to generate visualization for all of the aforementioned teams (Data analysis function). It also develops state of the art machine learning models to better the recommendation pipeline and conduct other tasks (curator flagging, Natural Language Processing for better recommendations and automatic biography analysis...) (Data science function). Tight collaboration with the Web&Devops team happens on the data ingestion and infrastructure for the platform : This is the MLOps responsiblity of the team.

Finally, and this is where the internship takes place, the R&D facet of the data team focuses on research and development in Machine Learning and in the case of this internship, audio signal processing. In the scope of this facet, state of the art is explored to produce new relevant results towards implementing new models in the scope of the data science responsibility of the data team. The members of the data team are described here:

- **Santiago Saint-Supéry**, data team lead, participates in all facets of the data team
- **Michael Murtagh**, data scientist, aids in data analysis and data science matters, as well as R&D
- **Joris Caloud**, MLOps engineer, focuses mostly on the MLOps part of the team's responsiblities
- **Myself**, mostly focused on R&D (almost exclusively), though I participate sometimes in data analysis matters when help is needed, which has allowed me to broaden my horizons in data analysis frameworks and tools.

2.2. Context of the internship

To understand the role of audio processing in the internship, we must understand the technical context that goes behind the recommender system at Groover. to do that, we have to understand the current recommendation pipeline and how the intuition that signal-based music tagging can benefit it.

2.2.1. Data-driven artist-to-curator recommender system

What is a recommender system?

A recommender system is a type of machine learning algorithm, the goal of which is to make recommendations to a user based on a catalog of kown objects in a database. Use cases of such algorithms could be online shopping (**Amazon, Ebay, Asos or any online retail store**), Music recommendation (**Spotify, Apple music, Soundcloud...** , Video recommendation (**Youtube, Spotify**), Book recommendations... The use cases are numerous. Recommender systems as of late have become a high-value space in the machine learning landscape due to the existence of more and more complex and diverse models and concepts to apply to the idea of a recommender system. Recommender systems can be divided into two types : Collaborative recommender systems and content-based recommender systems.

Collaborative recommender systems

Collaborative recommender systems rely on the interactions of other users with items in the catalog to deduce which item to best recommend. Consider a user-item interaction matrix where each interaction between a user and an item (an interaction can be a like, a purchase, a view, a rating, an early skip, a full listen in the case of songs...), is given an interaction score. Figure 2.4 shows the global appearance of a user-item interaction matrix. For our use case we consider the score is a 1-5 rating:

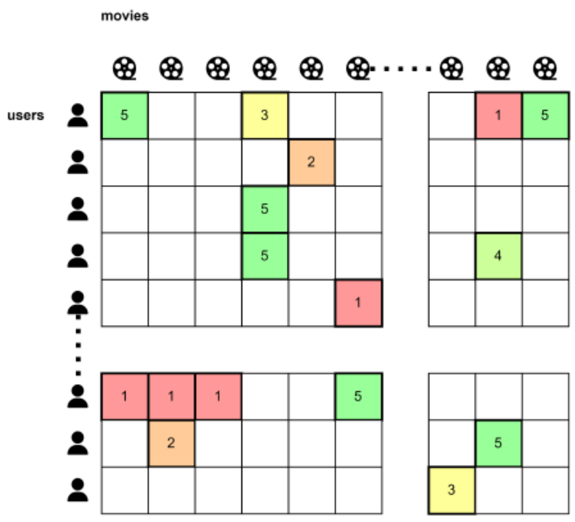


Figure 2.4.: Basic User-Item Interaction Grid for recommender systems

The intuition here is that by relying upon past user interactions, it can be possible to intuite what the user will choose next. This is possible by two methods : **Method-based inference** and **memory-based inference**. In memory-based inference, items and users are represented by large relatively sparse vectors, and recommendations are done based on nearest neighbours. **Users similar to each other are likely to Want the same items**. With memory-based inference however, an attempt to exhibit a latent model is done, model which can capture the relation between users and items. Users and items interactions are distilled into learned denser vector representations and a model is built to try to predict the best interaction in all the catalog.

However, this method suffers from the **cold start problem**: How to recommend anything to a user that has never interacted with anything before? This prompted the need for the other type of recommender systems :

Content-based recommender systems

Content based recommendation, on the other hands, focus rather on the features of the users and the items. For instance, the intuition that young men would prefer action movies than elderly women, is relevant. So is the inuition that French people would prefer French-spoken podcasts than spanish citizens. This intuition can yield a simple method that allows for recommending catalog items to users that have never purchased anything insofar that their features are available. Consider a model in which the respective features of the user and the item are used as input features for the model, which learns on pre-existing interactions without consideration for user trends and history. Figure 2.5 shows the basic working of a content-based recommender system, with feature vectors for the item and the user.

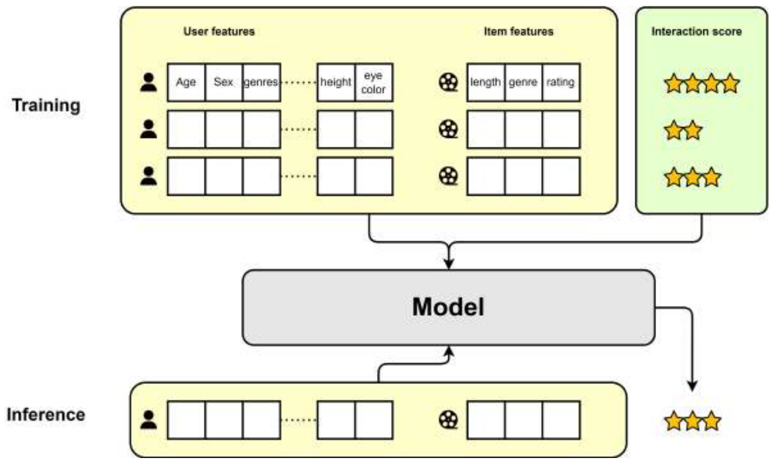


Figure 2.5.: Content-based recommendation example

So the model learns from the scores and codified features of the user and items and this learning is generalized at inference time by attempting to predict an interaction score for each item in the catalog (summarily, as some catalogs are very large and it is not possible to allow such computation times. However, the methods used to deal with very large catalog search in recommender systems are outside the scope of the internship and this report, so will not be covered though very interesting). Predictions are then ranked according to best score and the top items by predicted score are presented first to the user.

Hybrid models and a quick summary of the evolution of recommender systems.

These methods are somewhat mutually exclusive, as content-based recommendation does not take into account past behaviour of users. Collaborative filtering, on the other hand, does not take into account item features. So, most current recommender systems are what are called hybrid recommender systems : a mix of the two previous methods. Figure 2.6 summarizes the difference between the previous sytems and shows how a hybrid compromise is beneficial to the complementarity of the previous methods.

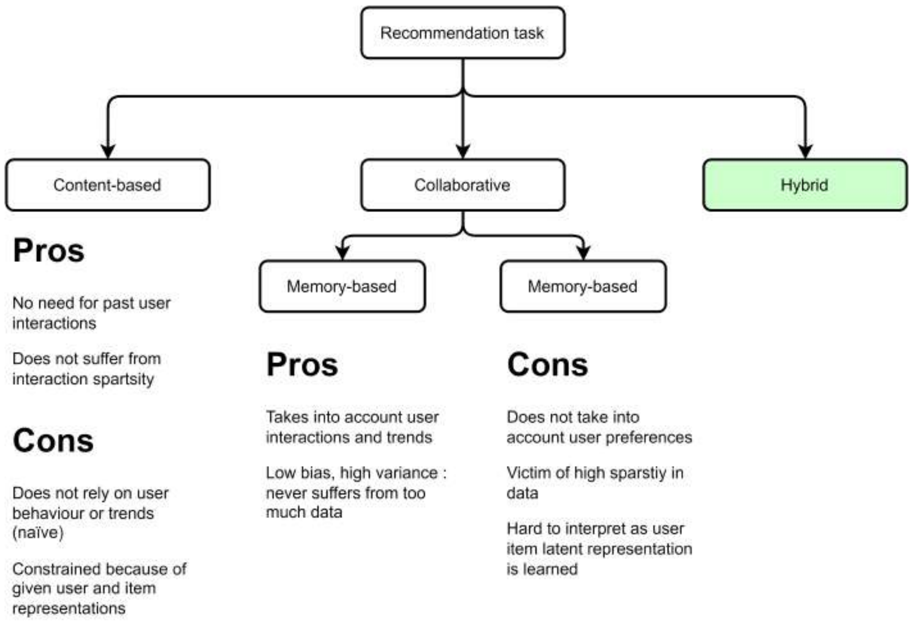

Recommendation task
Content-based
Collaborative
Hybrid
Memory-based
Memory-based
Pros
No need for past user interactions
Does not suffer from interaction spartsity
Cons
Does not rely on user behaviour or trends (naïve)
Constrained because of given user and item representations
Pros
Takes into account user interactions and trends
Low bias, high variance : never suffers from too much data
Cons
Does not take into account user preferences
Victim of high sparstiy in data
Hard to interpret as user item latent representation is learned

Figure 2.6.: Summary of the types of recommender systems

A deep dive into the recent development of recommender systems is outside of the scope of this internship. However, it is important to at least conduct an overview of the recent advances and a quick history of recommender systems to understand how the recommender system at Groover fits into this timeline.

A quick history of recommender systems

Collaborative item-item approaches and user-user approaches come first in the history of recommender systems. Essentially, this approach considers items in the item space and users in the user space and performs K nearest neighbours on the other vectors taken from the interaction matrix. Figure 2.7 below shows the working principle of item-item and user-user approaches:

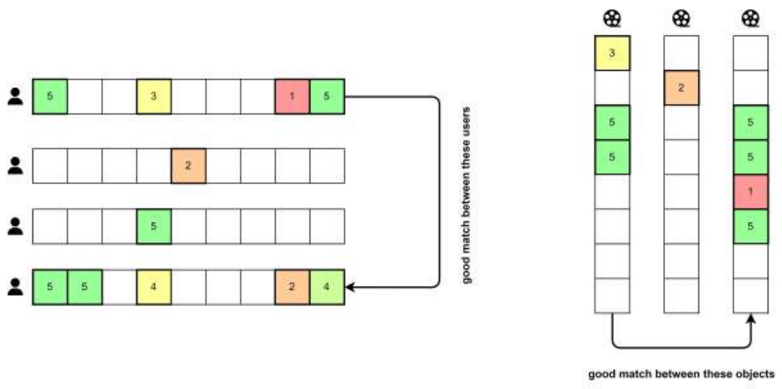

good match between these users
good match between these objects

Figure 2.7.: item-item and user-user approaches for recommendation

To bring precision to this difference, in the item-item approach, the user is recommended items similar to his preferred items. In the user-user approach, the user is recommended items preferred by users with similar tastes. A few issues arise here: for large catalogs and large user bases, the search algorithm does not scale well at all ($KNN\ is\ in\ O(nmk)$). Furthermore, successful items will always be recommended in this fashion. To palliate this, Interaction Matrix Factorization was conceptualized.

The intuition is as follows : the user-item interaction matrix is a representation of interactions between users and items that is the result of latent descriptors that are either known or unknown. For content based filtering, we can provide these features and let the model learn the interaction between these features. With matrix factorization, we let the model learn the representations themselves. We assume the matrix M can be decomposed into the product of a User matrix U and a item matrix I with a residual error reconstruction matrix E, as shown in Figure 2.8 [33] [16]:

$$M = UI + E \tag{2.1}$$

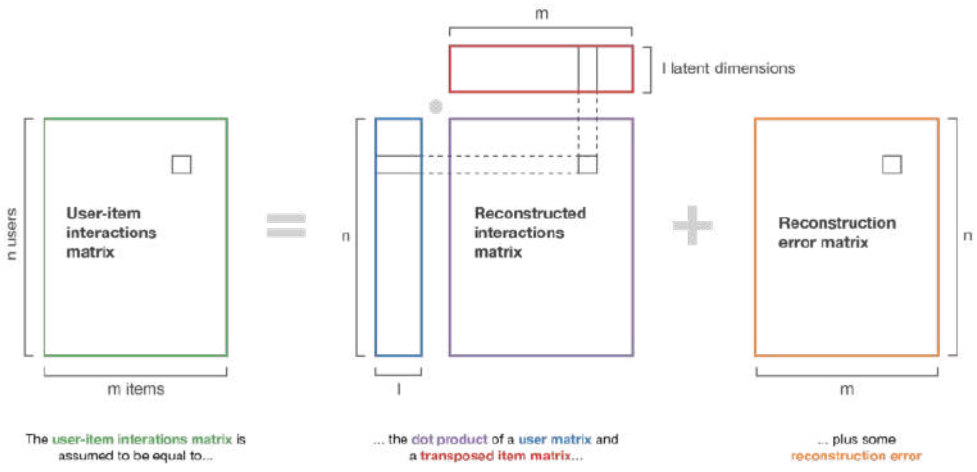

m
l latent dimensions
n users
User-item interactions matrix
m items
n
l
Reconstructed interactions matrix
Reconstruction error matrix
n
m
The user-item interations matrix is assumed to be equal to...
... the dot product of a user matrix and a transposed item matrix...
... plus some reconstruction error

Figure 2.8.: Matrix factorization working principle (Image from[33])

Finding the best latent representation for users and idem corresponds to a minimization problem defined as follows [33]:

$$(U, I) = argmin_{UI} \sum_{i,j \in E} [U_i I_j^T - M_{ij}]^2$$

Which can be learned by gradient descent backpropagation classically as with with any machine learning algorithm. This allows to compute the recommendation score for a user and an item by calculating the associated learned representation vecors, which deals with the issue of scaling. However, a new issue arises, which is that of Matrix Sparsity. These interaction matrices are often practically empty for large catalogs (sparsity at Groover is 99.5%), which makes learning latent representations a high variance problem, requiring large latent representations.

A few solutions were proposed to paliate this over which we will not go into detail on, as the state of the art rapidly moved on to newer and more problem-appropriate models:

Sequential and session-based recommenders, modern recommender architectures.

Following these initial models, newer and deeper models which aim to encompass more inherent characteristics of what a recommendation problem is made their appearance. Deep recommender systems started combining collaborative and content-based recommender systems by applying learnable models to vocabulary embeddings of both users and items, to combine both user behaviour representations and item feature representation [45] [14]. The general structure of such a network is shown in figure 2.9:

After combining user-behaviour and content into one model the notion of sequentiality came about : user behaviour while making choices based on recommendations is naturally sequential. The user interacts with a certain amount of items and the order of interaction is important. For instance, if the user listens to episode 1 of a podcast, recommending episode 2 next seems like a fair intuition. Without going into details. From 2014, which is when the original study for collaborative & content based recommendations featured [45], recommender systems aimed to tackle sequentiality:

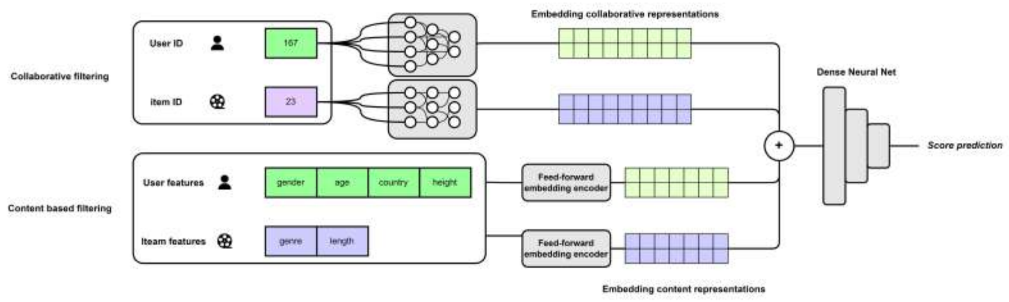


Figure 2.9.: A collaborative-content Neural Recommender System much like that at Groover

- Recurrent Neural networks tackled the problem of sequentiality first due to their appropriate architecture for handling sequences [35] [18] [13].
- Convolutional Neural networks followed suit [39] as well as graph neural networks due to their ability to model relationships between entities [53].

Sessionality is another aspect of recommendation that should not be neglected : between sessions a user will not necessarily have the same needs. Consider a user logging in to a electronics retail website to purchase a computer one day and a phone the next. If a computer was purchased the previous day, though It might make sense to recommend a computer based on the sequence, it does not based on the session.

Recurrent Neural Networks, again, tackled this problem first [15], and the session-based recommender system has since become state of the art [46]. Many domains have branched out from the exploration of these sequential and sessional models, including transformers, which have established themselves as *de facto* state of the art on recommendation problems [38], [5]. Recently, approaches in self-supervised learning [23] [54] and reinforcement learning [28] , [1] have shown the value of autonomous learning and trial-and-error algorithms for the recommendation problem. **Contextual bandits** for instance is an algorithm in which individual agents called bandits attempt to propose new recommendations using environment-based decision making using an evolving reward mechanism. This model has shown promise in recommender systems by introducing a stochastic dimension to recommendation and 'mixing up' the recommendations, learning when the recommended item is or is not interacted with, and converging upon human-like recommender behaviour that is not purely representation-driven.

This, of course, is only an overview of the possibilites linked to recommender systems and many more paths have been explored as of late with deeper and deeper architectures. The recommender system is important in the scope of this internship as it represents the core of R&D at Groover, and the objectives of this internship all converge towards building upon the recommender system and making it more performand using audio processing and deep learning applied to musical signals.

The Groover recommendation pipeline

This section focuses on taking a look at the particularities of the Groover recommendation pipeline and the architecture at the beginning of the internship. We define the intricacies of the Groover recommendation task in relation to other more classic recommendation tasks, as well as exhibit the current architecture for the model and the business-educated decisions acted upon to design the data preprocessing pipeline and model itself.

Recall that the objective of recommendation at Groover is to recommend relevant curators to independent artists based on their objectives for the track they have submitted for the campaign. This relevancy can be based on user - which are also referred to as bands - and curator features, but also on the sequence of interactions of bands with curators and curators with bands. The Groover recommendation task is a tricky one. The goal is not purely to recommend curators based as items - curators are humans or organizations of humans, so a purely catalog-like recommendation task, such as **Amazon, Ebay, HM or any retail store** would tackle, does not suffice to encapsulate the problem. Neither is it a purely network-driven task such as Facebook or Linkedin, where users are recommended to users, and (most of the time) the interaction between these users are equal. It is similar in that the band will send a request to a curator to listen to, review or share their track, and the curator will either accept or deny - as he would a connection request on Linkedin - however as the curator delivers feedback and a service to the band, it cannot be only seen as a user-to-user interaction.

The curator here, which is the recommended (to the *recommendee*, which is the band) acts as both a purchasable item and an interactable user. This difference with most common recommendation tasks makes the Groover recommendation pipeline an interesting task to tackle. As the pipeline is in constant evolution, the following explored the recommender architecture at the beginning of the internship.

The database at Groover stores information for both bands and curators, some of which can be both. This information, for bands, can be **The country of origin, a vector of relevant genres and subgenres, the goal of the campaign, relevant information extracted from the biography**. The same can be applied to curators with some extra features : curators are required at sign up to specify a list of **liked, neutral and hated** subgenres. Furthermore, artists and curators are specified by BandID and InfluencerIDs, which pertain to the collaborative aspect of the recommendation system. All interactions between bands and curators are annotated with a score from 0 to 1:

- 0 means the track was not accepted by the influencer : it is a negative interaction, and the model should learn to attribute a score close to 0 to interactions that will probably not end in an acceptation from the influencer
- 0.25 means the curator has provided feedback on the track but has not decided to share it
- 0.5 means the curator has agreed to share the track but has not yet done it
- 1 means the curator has shared the track

So interactions are scored in ascending order : better interactions are scored closer to one (or just have a higher score) and worse interactions score lower. To summarize this, the following tables show the database tables available for use by the recommender system :

Interaction table		
bandID	*InfluencerID*	*Score*
12	2	1
43	3	0
661	2	0.25

Table 2.1.: Interaction table example

Band table					
bandID	*Subgenres*	*Country*	...	*Bio information*	*Date joined*
1	[rock, pop]	FR		[1999, ...]	10/1/2021
2	[drumstep, hardcore]	US		[Michigan ...]	12/5/2020

Table 2.2.: Band Table example

Band Table						
BandID	*Country*	...	*Language*	*liked subgenres*	*hated subgenres*	*neutral subgenres*
1	FR		FR	[trance, techno]	[ballad, folk]	[acid]
2	USA		EN	...	...	...

Table 2.3.: Curator table examples

The task is similar to what was described before : the interaction table can be reformulated as an interaction matrix, and the band and influencer tables provide the features needed for content-based recommendation. The task can be formulated as a gression task with the objective to predict the score for a given interaction between a user and a curator. At inference time, the user who is asking for recommended curators is screened through the whole catalog of curators and the results are arg-ranked by score, providing in theory most relevant curators at the top.

Based on [45] [18], the recommender system at the start of the internship had the following general architecture (Figure 2.10) *Nota : for confidentiality purposes, the exact architecture of the recommender system shall not be divulged in this report. However, it is not relevant to know the exact blocks and parameters of each of these blocks in our case - nor is it relevant to the scope of the internship*:

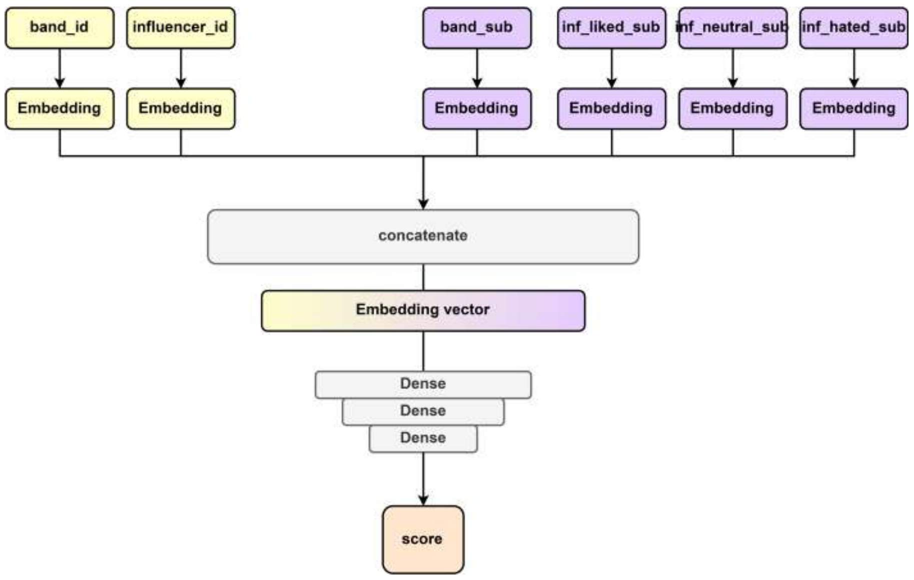


Figure 2.10.: Base recommender system architecture at Groover

Note that here only subgenre content features are shown, but others are taken into account. This a relatively basic collaborative-content neural model shown previously. However, as we have discussed in the previous section, curators are not catalog objects : they have behaviours just as bands do that might bias the recommender system. Business-motivated solutions are proposed to these issues to make the recommender system more prevalent to the considered use case.

- **Influencer that always accept tracks sent by bands will be strongly positively biased by the algorithm**. This is due to the collaborative filtering aspect of the recommender system. Intuitively, highly selective curators are more desirable and should be more highly ranked in the recommendations. The adopted solution is to penalize the score of influencers by a pickyness rate variable. The higher the pickyness rate of the influencer, the more the score of the interaction is boosted during training. The lower the pickyness, the more the score is penalized.

- **Influencers can bias the recommendation by saying they are partial to all subgenres**. The standard interaction is on average more positive between bands and curators that have similar liked genres. This is a relationship that the model will *a priori learn and take into account at inference time.* So, by saying that they love all subgenres, an influencer can be ranked higher for all artists, which is undesirable. **Again, the score is peanlized at training time for influencers who fill up the whole subgenre list in their liked subgenres section**

Finally, though the model learns relevant correlations between both content features (namely subgenre correspondence) and past interactions between users and influencers, we want it to be somewhat grounded in an intuitive use-case truth : **Bands have a higher chance of being accepted by curators with similar tastes in music.** And though the model is expected to learn this, it is not explicitely provided (recall that collaborative content filtering relies on learned and not provided representations). The solution to this is adjust the score at training time in accordance to the cosine similarity between vector representations of the subgenres for the band and liked and hated subgenres for the curators. The simplest vector representation to use that provides a known representation and is not learned is **one-hot encoding**. Given a vocabulary V of cardinality n;, the one-hot representation of a list of items of that vocabulary is the vector:

$$OH = \begin{bmatrix} \delta_0 \\ \delta_1 \\ \vdots \\ \delta_n \end{bmatrix} \tag{2.2}$$

Where δ_i is the kronecker delta equal to 1 if the ith item of the vocabulary is in the list of items. To compute a ground truth of perceived similarity between the subgenres of the band and the liked and hated subgenres of the curator, we use cosine similarity, which is defined for two vectors A and B as :

$$Cosim(A, B) = \frac{A \cdot B}{|A||B|} \tag{2.3}$$

Which is a measure of the similarity of n-dimensional vectors. We then mitigate the score at training according to this obtained similarity. Including all score mitigation to reflect curator behaviour and business use case applicability, the final training score expression can be given as follows:

$$s_{new} = s_{original} - p_{subgenres} - p_{pickyness} + cosim(OH_{band}, OH_{liked}) - cosim(OH_{band}, OH_{hated}) \tag{2.4}$$

Where s denotes score and p denotes penalization. This is the final regression problem the recommender system aims to tackle. The following figure (Figure 2.11) shows the adapted architecture from previously with the inclusion of the mitigations

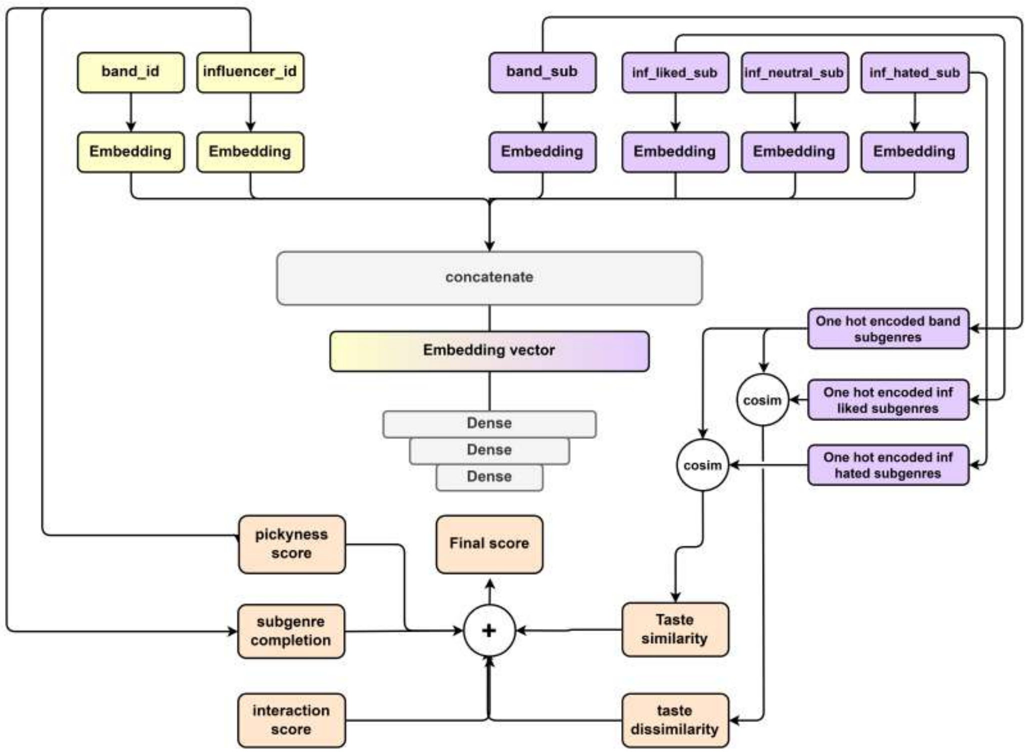


Figure 2.11.: Current recommender system

This is the architecture as it stands at the start of the internship. As mentioned previously, the recommender system is the subject of most R&D efforts from the data team at Groover. And as was stated previously, many directions have been explored by state of the art to better recommender systems. Here are the current directions that are being explored at Groover - without going into much detail as the specifics of these are outside the scope of the internship:

- **Contextual bandits**, which were described previously, have already been implemented and deployed on part of the user base. Using the recommender-specific and business-specific metrics which will be elaborated on in the next section, the effectiveness of the algorithm on the recommendation system will be evaluated and generalized to the whole user base of needed
- Efforts in natural language processing to extract relevant features from artist biographies
- Efforts in sequential and session-sequential recommender systems to model user sessions (campaigns)

And of course, the field of R&D which this internship is focused on is the incorporation of audio feature extraction models at scale in the pipeline to derive audio features from artist-uploaded. We will elaborate on the exact goal and potential value of these audio features in a following section but the intuition is simple : Groover is an audio-centered platform that conducts user-to-user recommendation. There must be some usable feature in the audio files provided by the artists to make better recommendation. This sets the context for the internship and explains the importance of signal processing, audio machine learning and latent representation learning in this context.

2.2.2. Music tagging, signal-based recommendation

An important domain of R&D at Groover is Audio, and more specifically automatic music tagging. This section focuses on explaining the task of music tagging as well as the potential applications at Groover. We also discuss the metrics, both business-oriented and ML-oriented, used to evaluate a recommendation system. As the goal of the internship is at term to develop a model and put it into production, we need measurable indicators that the boost in recommendation quality brought by our model is substantial.

What is music tagging?

Music tagging is a sub-task of the more general music classification task, part of a canonical and previously explored set of tasks in the domain of Music Information Retrieval, and more Specifically Discriminatory Music Information Retrieval.

Music is a subjective art and qualifying it objectively is debatable even by human metrics. However, it is possible to attribute to a musical track a set of **tags** in an attempt to describe it as concisely as possible. These tags are used to describe high-level information about a given piece of music and are used by music streaming platforms (**Spotify, Deezer, Apple Music, Youtube Music**) to construct playlist or make recommendations. Tags can be seen as a set of descriptors of music, which can be of many types : **Genres, Subgenres, Acoustic Features such as danceability, energy, Moods, themes, instruments, vocal registers...**. The following figure (Figure 2.12) shows a set of example tags for the song *South of the river*, by **tom misch**:

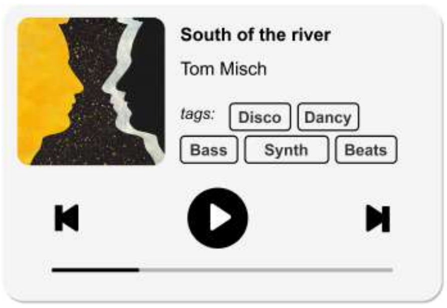


Figure 2.12.: An example of tags on a music track

These tags were manually annotated and thus subject to high variance and subjectivity until [40] aggregated hand-crafted acoustic and signal processing descriptors of frame-level samples of music pieces to predict the musical genre of the audio sample. This prompted the exploration of Deep Neural Networks for use as music classifiers, starting with [6] in 2016. Deen Neural Networks provide the advantage of learning relevant harmonic and temporal representations of audio signals through their downstream task objective and thus sidestep the need for hand-crafted audio features [6].

So the goal of music tagging is to predict a set of labels for a given audio track. This makes it a multi class classification task per the multiple labels that can be assigned to the tracks. However, there can be multiple labels - tags - attributed to a single track, which makes this task a **multilabel multiclass classification task**. The objective of the task becomes the following : given an audio signal, model a predictive score for each label the model is trained on based on the audio features of the track and rank these scores to predict the relevant tags for the track. The following figure (Figure 2.13) shows the inference pipeline for the general task of automatic music tagging:

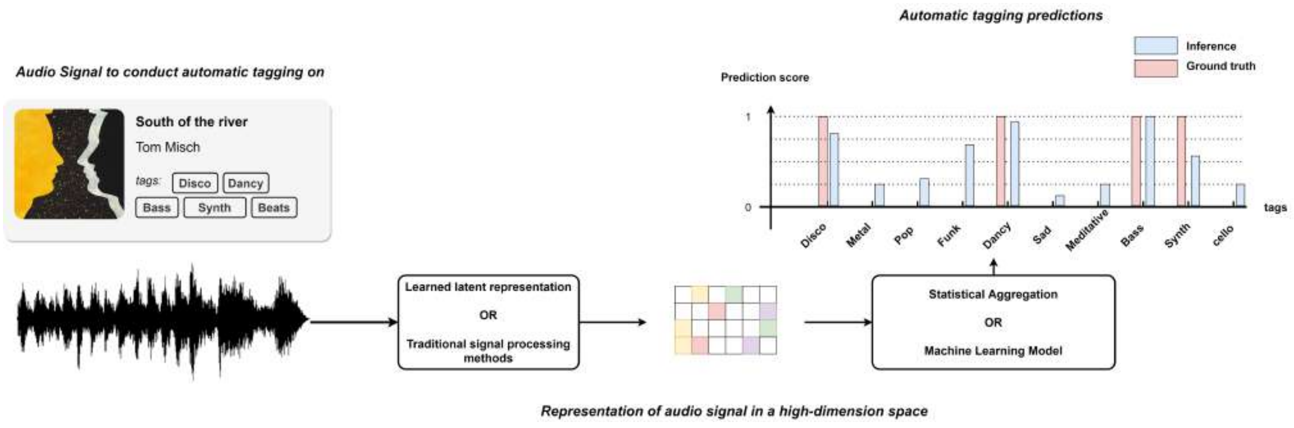


Figure 2.13.: End to end music tagging

In the above figure, ideally inference predicted scores would be as close as possible to the ground truth scores. However, music tagging is tricky, per subjectivity of the task itself, which induces noisy datasets, and the complexity of attaining high accuracy on such a task.

Metrics for automatic music tagging

We want to evaluate the accuracy of the music tagging algorithm we implement before incorporating it into the recommendation pipeline to ensure the tag representations we use are coherent (in other words, we want our tagging algo to be the best as possible to provide meaningful tags to the recommender system). To do this, we use three metrics. First, recall the format of the model output versus the ground truth. The model output is a vector of n floating point values from 0 to 1 that represent the relevancy of the n tags. The ground truth value is a vector of same length with binary values of 0 and 1 (See figure 2.14)

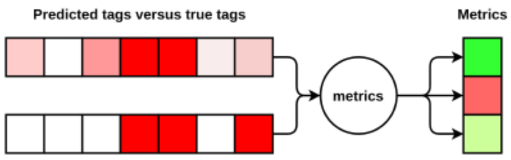


Figure 2.14.: Metric evaluation for music tagging

Area Under Receiver-Operator-Characteristic Curve The first metric is the area under the receiver operator characteristic curve. Consider first TP true positive predictions - where the prediction is close to 1 and the actual value was 1, FP false positives where the predicted value was close to 1 and the actual value close to 0, and their equivalents FN false negatives and TN true negatives. We define TPR true positive rate and $FPRthefalsepositiverateas$:

$$TPR = \frac{TP}{TP + FN} \quad \text{and} \quad FPR = \frac{FP}{FP + TN} \tag{2.5}$$

The receiver operator characteristic curve is defined as the couple (FPR,TPR) at one decision threshold for the model selection. For instance, if we choose the first decision threshold as 0.5 such that all values predicted above 0.5 are determined to be 1, this will yield an evaluation of (FPR, TPR) for this threshold. The ROC curve is the plotting of all these (FPR,TPR) tuples for many decision thresholds (figure 2.15):

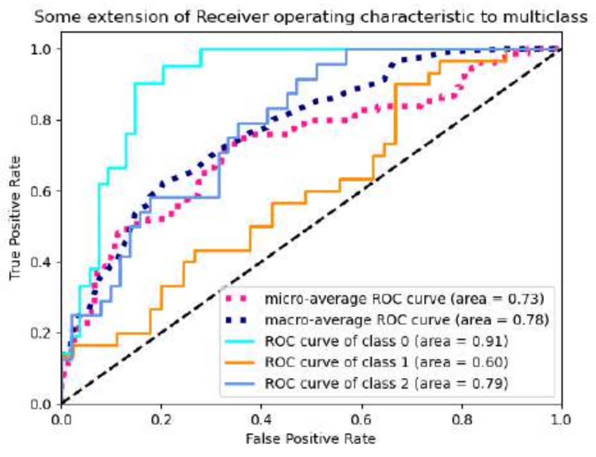


Figure 2.15.: ROC curve for a general ML application

The Area Under the ROC Curve (ROC-AUC) is one of the metrics used to determine multilabel multiclass classification performance as it is threshold invariant and shows a performance scale from 0 to 1 - A perfect classifier having 1 TPR for all FPRs.

Area under Precision-Recall Curve We define the Precision-Recall (PR) Curve as the curve (Pr,Re) Where Pr and Re are Precision and recall, defined as:

$$Pr = \frac{TP}{TP + TN} \quad \text{and} \quad Re = \frac{TP}{TP + FN} \tag{2.6}$$

And the same as before, the better the model, the higher the area under the (Pr,Re) tuples curve for various decision thresholds will be. PR-AUC can also be called average precision (AP) and denotes how many of the retrieved tags are relevant.

Cosine Similarity Finally, although PR-AUC and ROC-AUC are the main metrics for the music tagging task [8] [6] [7], [4] [37], we also use average cosine similarity over the predicted tags (as defined in equation 2.7) to have a raw evaluation of the similarity of the predictions - essentially, the better the model, the better the average cosine similarity.

signal and tag-based recommendations at Groover

How can music automatic tagging help at Groover? Two main contributions could be brought to the recommendation pipeline by implementing automatic music tagging:

Subgenre suggestion at signup: Currently, when Artists sign up to Groover to launch a campaign, they define themselves by a set of subgenres. Likewise, curators define their liked and disliked sugenres themselves. While it is important to maintain personnalized input from both parties, human inputting of genres is both subjective and noisy, which introduces high variance in the recommender system training.

- **Subjectivity** entails that an artist that is, by a given curator's standards, closer to indie rock than alternative rock, but by the artists' own standard, closer to alt than indie, will define himself as being an alt artist over an indie artist, while he could reasonably be called both. This will bias the recommender system towards curators with an affinity for alt more than indie for this curator, meaning that 1) they might miss out on great curators for them, and 2) they might be recommended some curators that do NOT like their music as much as they would expect, thus leading to higher rejection rates.
- **Noisiness** is caused by the fact that users and curators alike might define themselves by many subgenres to boost their "matchability", or might put only one subgenre because they do not really know how to define their music. Suggesting subgenres to artists at signup based on the analysis of their music by auto-tagging might reduce the variance of the distribution of number of subgenres and thus lead to a less noisy dataset for training the recommender system.

So, the first application of music automatic tagging is to present artists with machine-based recommendations regarding subgenres at signup based on their music to reduce the noisiness and subjectivity of the data fed to the recommender system, which intuitively would lead to better recommendations.

The other main contribution that can be made is contributing directly to the content-based features in the recommender system. By storing the musical tags of tracks for each artist and accepted and rejected tracks for each influencer, one can build an average/majority vote/softmax tag-score identity of the user (which can be either a curator or a band). At training time and inference time, these consitute valuable content features that are neither subjective nor noisy. They are not subjective because these "tag vectors" are learned representations from a model that is deterministic once trained, and they are not noisy because we can implement a fixed number of top-tags to consider for the tag representation, and if we do not even then the vocablary size of the tags is fixed.

With regards to what can be seen in figure 2.11, these tag representations of artist and curator tastes can also be used as another "ground truth booster", as with the cosine similarity ground truth in the aforementioned figure. Recall that these tags can be moods, themes, languages, instruments, which includes much more diversity of taste representation than a higher-level subgenre to determine taste. Cosine similarity in the case of tags would also take into account if a curator is in the habit of recommending more uplifting tracks, more energetic tracks, tracks with good bass lines. The new recommender system, integrating tag representations, would then look something like this (Figure 2.16):

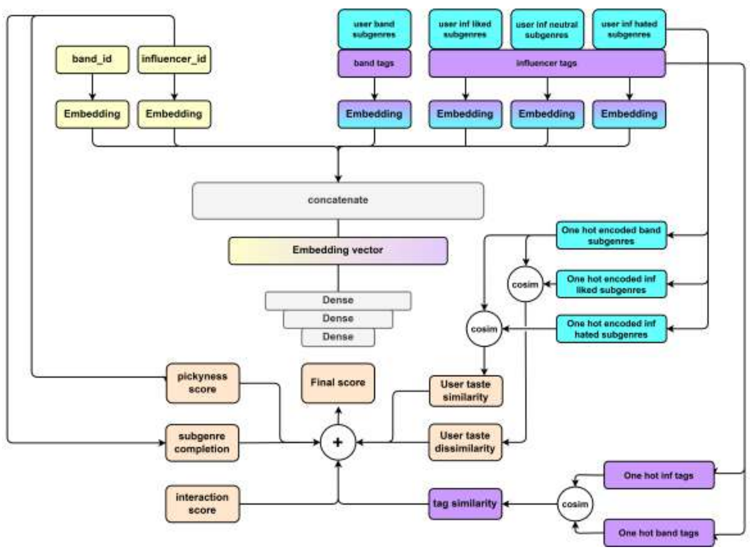


Figure 2.16.: Automatic music tagging contribution to the recommender system

Of course, these tagging representations can also be used as features in any other model that takes content-based representations as an input. So should the architecture change in the future, implementing these tagging models at scale will serve for the next version of the recommendation pipeline as well.

Measurable results of better recommendation

it is important to monitor the influence of modifications upon the recommendation pipeline through quantitative metrics : these metrics can be separated into two main categories : **business related categories** and **recommender system metrics**. In all this section, *preds* denotes model predictions (scores) and *gt* denotes ground truth scores:

Recommender system metrics

- **(R)MSE or Root Mean Squared Error** is a common metric for evaluating regression tasks. Since our task is to predict a continuous score between 0 and 1, this metric is adapted to the evaluating the quality of the score predictions.

$$RMSE(preds, gt) = \sqrt{\sum_i \frac{(pred_i - gt_i)^2}{n}} \tag{2.7}$$

  RMSE is robust against 0-valued ground truth and distribuion shift in predictive data, which makes it an adequate loss function for training our model.

- **MA(P)E** Mean Absolute (Percentage) Error makes more sense from an interpretability standpoint, as it is used to evaluate the percentile or absolute difference between ground truth score and score prediction. However, it is weak to 0-valued ground truth and thus is not used as our loss metric but rather as an interpretable metric.

$$MAPE(preds, gt) = \frac{1}{n}\sum_i |pred_i - gt_i| \tag{2.8}$$

- **Cosine similarity**, described previously, is used to describe the subgenre similarities between the user and the predicted influencer. Intuition gives that the higher the similarity between top predictions, the better the recommender system. We define Average Cosine Similarity at k (ACS_k) as the average cosine similarity between the artists' subgenres and the curators' liked subgenres for the top k ranked curators for a given artist (this is averaged over all recommendations in the test set)

$$ACS_k(preds, gt) = \frac{1}{nk}\sum_n \sum_{i<k} Cosim(subgenres_{artist}, subgenres^i_{curator}) \tag{2.9}$$

- **Personalisation index** is a measure of the variability of recommendations for different users. For instance, for a database of two users and six curators A,B,C,D,E,F, two recommendations (A,B,C) would yield a personnality index of 0. (A,B,C) and (A,E,F) would yield 0.66. The personalisation index is a measure of the capacity of a recommender system to adapt to different tastes.

- **Mean Average Precision (MAP) and Normalized Discounted Cumulative Gain**: Both are a measure of the capacity of the model to target the correct curators for a given band in the top k recommendations, and NDCG specifically is a measure of their order of relevancy [16]. If NDCG is high at 5 and low at 10 then the 5 first recommendations are correctly ordered by order of relevance and recommendations 5-10 are poorly ordered. MAP does not take into account ordering, however. It is better to strike a balance between these two metrics to consider order of recommendation as well as hit rate.

Should the developed recommender system including audio be put into prediction before the end of the internship, these metrics will serve as a benchmark comparing the newer recommender system to the previous one.

Business metrics for recommender system monitoring The following business metrics are used to determine whether or not a recommender system is beneficial from the user point of view at Groover:

- **Number of recommendations tracks were sent too** In the first page looking at how many of the top k recommendations were selected by bands when building their campaigns is crucial to knowing whether the recommender system is doing a good job at finding relevant curators
- **Missed shot count**: The proportion of recommendations that were sent a track and refused it (predicted high score and truth zero score). The lower the Missed Shot Count (MSC) or "Sadness rate", the better the recommender system.
- **Time spent on platform and time spent looking at influencers**. The lower the time spent looking at influencers and the lower the time spent on the platform, the less the recommendations of the recommender system have caught the eye of the user
- **Feedback rating & number of active contacts**: On the platform, users are able to rate influencers for their feedbacks (e.g a 5 star feedback is a good inlfuencer match, a 2 star is a bad influencer match). This and the amount of influencers the bands are still in contact with after a campaign are good metrics to evaluate whether or not the recommender system is providing good recommendations *a posteriori*

2.3. Objectives for the internship, timelines

This section focuses on defining clear objectives for the internship as well as expected timelines from the point of view of Groover. The macroscopic and single most important objective of the internship is **Implementing a music tagging model into the current recommendation pipeline**, based on what the previous section covered. However, this task can be decomposed into multiple phases: **A research phase, a development phase, and a production - quantiy control phase**. Furthermore, the internship will also have subtasks to support the data team in other endavours, which will be briefly presented here.

2.3.1. Research on the state of the art of audio processing for and by music tagging applications

The first phase will be a state of the art research phase to establish all the necessary information about music auto tagging. What it is, what are the main models to conduct it, what is the history of the task, the contributions over the years, as well as the necessary preprocessing.

Audio preprocessing is an important bottleneck for most audio machine learning applications as the audio representation derived from the signal processing methods selected for the task will dictate the representation used as input for the model. Spectral resolution, sampling rate, sampling algorithms, Spectral Weighting, mel scales and many other potential representations will be thoroughly and comprehensively explored through this step in the research process, which is an important one and will fully exacerbate the need for a clear understanding of audio preprocessing techniques to deal with audio machine learning problems.

Audio Augmentation techniques will also be explored. The state of the art of audio models for tagging will be discussed, and as a last step the available data to train our models on will be explored : open source datasets, the data they contain, as well as their strengths and limitations, will be discussed.

Overall, the Research phase for this project is expected to last about 2 months, with all the previous steps being tackled in that order. These steps are highly interconnected however, so it will be necessary to tackle them in order of best comprehension, while conducting back-and-forth checks between the various aspects of audio tagging (e.g model choice is highly dependent on both audio preprocessing steps and available data)

2.3.2. R & D on implementation and benchmarking of audio preprocessing techniques and music tagging models

Once the state of the art for these models has been established, the available models and methods will be selected based on time constraints, material constraints, data constraints, and task appropriacy. Furthermore, the full preprocessing, data wrangling and selection pipeline as well as the implementation of the various models described in the previous section and their evaluation will be undertaken. The implementation of this code base as a modular pipeline is expected to take about 3 months, leading to the near end of the internship. This means :

- fully analyzing the preprocessing needs for the task based on the models we have selected based on the available data and preprocessing techniques
- building the preprocessing pipeline including all signal preprocessing steps
- implementing the relevant and selected models
- building and executing the training pipelines on all the models and the selected tags
- Evaluating the various models on the evaluation split to benchmark the best ones
- finetuning the best models for our task
- going back on the preprocessing, training and evaluation pipeline when needed.

2.3.3. At-scale implementation of the designed music tagging pipeline

With an estimated 1 month left in the internship, remains to productionize the code for the music tagging model at scale on the Groover artist database and all the stored music clips. This includes setting up the model to be trained. This will be a distributed task over most of the tech team but the model will still need to be monitored during this time. So, the overall Estimated timeline for the project is shown in the following figure (Figure 2.17)

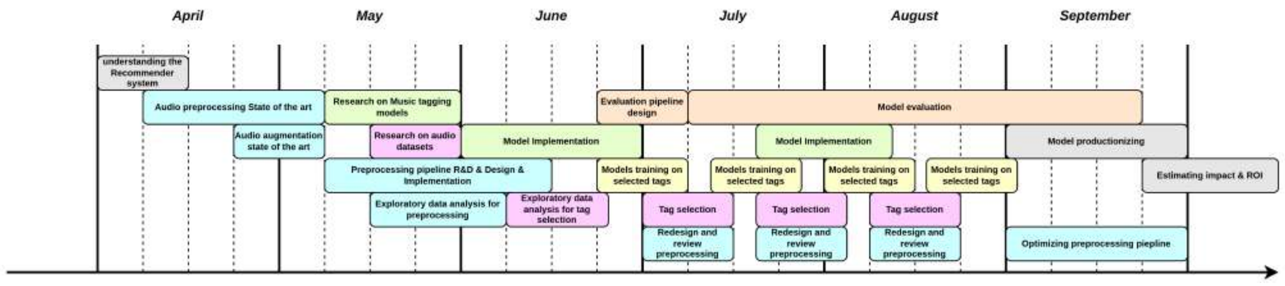


Figure 2.17.: Estimated timeline of the project

As can be seen, this will be far from a linear project. Mutliple reconsideration based on model performance, current model implementations, future model implementations, and model training will have to be undertaken, and back-and-forths between tasks will be inevitable. One of the objectives of the internship will also be to minimize these back and forths.

2.3.4. Developing the research culture at Groover

Another sub-objective for this internship will be to develop the research culture and presence at Groover, by writing articles and pursuing continuing research presentations throughout the internship. These are covered succintly is this section, as they are not a main part of the internship.

Paper talks: Audio and Deep Learning biweekly paper reviews

One of the goals of the internship was to start and maintain biweekly paper presentations on state of the art of machine learning and audio tagging. These presentations were cycled between the members of the data team and their goal was to maintain state of the art knowledge amongst the team members. These papers would also be rewritten as short vulgarization articles for medium, an open blogging platform where the machine learning community is quite developed. Below the links for the presentations that were created by myself:

- **Diffusion models presentation**
- **Codified audio language models presentation**
- **Contrastive learning of musical representations presentation**

Writing articles for a deeper research culture at Groover

To develop the online research presence of Groover, both in-depth articles and vulgarzation articles are written on state of the art evolutions in machine learning and the relevant tasks at groover (Recommender systems and Audio processing Machine Learning algorithms). A non-negligible part of the relationship was spent on writing an in-depth article on the role of the transformer architecture [44] on recent machine learning applications. Not only is it revolutionary and fundamental in today's machine learning landscape, but also relevant to some of the models presented in Chapter 4 4. The article can be found below:

Transformer in-depth article