Hit Helper

By: Henry Blue, Mikael Hokkanen, and Pritom Kumar Das

1. Introduction

Small Finnish independent musicians face a daunting challenge when trying to promote their music. With the rise of streaming platforms such as Spotify and Spotify's pivot to allow all independent musicians access to upload their songs, hundreds of thousands of songs are released onto Spotify alone every day (Willman). With such a variety of music, music streamers now rely on recommendation algorithms to find new music that suits their tastes.

Unfortunately, this presents a problem for small independent streamers. Music recommendation algorithms, such as the Spotify algorithm, use metrics like popularity and Collaborative filtering (a metric that describes a song's connection with other tracks on the platform by studying ties such as playlists) (Dmitry). Thus, the music from smaller musicians is often not recommended because they do not have many views and are not in many playlists.

One of the most recommended methods to solve this problem is to get your music playlisted on Spotify official playlists. Spotify official playlists are handpicked playlists made by Spotify curators. To get your song on a Spotify official playlist, you must Pitch your song to Spotify for each playlist you want to get on. Unfortunately, Spotify's curators must go through thousands of playlist submissions daily. To stand a chance against the competition, each pitch must be highly specialized to the requested playlist. To make matters worse, there are over 10000 Spotify official playlists that take applications. This makes it difficult to find which playlist you should spend your time on.

In this mini project, we solve the following problem: How can you find what playlists have the most commonalities with your songs?

To solve this problem, we created a web application that uses the URL of an input song to recommend five Spotify official playlists with similar music to that song. Our project uses Spotify's open-source API data to train a machine-learning classification model that finds playlists with similar characteristics to a given song. This way, we can help small independent musicians find relevant playlists to promote their songs.

1.1 Motivation:

The topic of the investigation changed from our original mini-project canvas. we always wanted to find a way to help small musicians in Finland by creating something that would help them promote their music. Our first idea was to quantify what made a Finnish song popular, using the characteristics of the song and data visualizations. We knew this was possible because we had found two data sets that held music with popularity metrics: The Spotify API and the Muff open-source database. Finding the characteristics that popular songs have in common was a good starting point and helped us learn about the Spotify API but was too easy and not useful. Once we had found those characteristics, we realized that it did not address any needs of our end user. Because of this, we decided to do more research into finding a service that would generate value for our end user.

After deciding to change the scope of our project we re-aligned our objectives with our target audience. Then we set out to help solve a more practical problem small musicians have; how can I get my music to the people who would most enjoy my music? This question is one of the most

fundamental questions of music production. No matter how good your music is, it is difficult for one person to find their target audience. From here, we discovered the problem discussed in the introduction and found that this was a problem we could solve that would provide an actionable outcome to our end user.

1.2 A Note on Data Privacy and Ethical Considerations

All the data we gathered in this project came from the open-source Spotify API. Thus, for ethical considerations, we made sure our project fulfils the Spotify Developer Terms and agreements. By fulfilling this agreement, we understand that we do not have nor claim the rights to license the songs we process. Furthermore, we have no for-profit intentions with this project. (Spotify Legal Team) Outside of those terms, there were no ethical impacts we saw fit to consider.

On the other hand, as this project used open-source data, we did not take any extra data privacy precautions. Through our machine learning platform, we only used open-source Spotify API data to train and test the classifier. Furthermore, when building the website, we used an open-source framework and do not store any user information. Because of this, we did not take extra steps toward data privacy as we were not handling any sensitive information.

1.3 Structure of the report.

This report is structured in the following way: First, we discuss our 2. Data Collection, Pre-processing, and Exploratory data analysis. Secondly, we will discuss our Learning Task and Learning Approach. Finally, we will go over our deliverables and conclusions of the project.

2. Data Collection, Pre-processing, and Exploratory data analysis.

From the beginning of our project, we understood that the scope of our project is dependent on the availability of the data we could collect. When deciding on our topic, we chose to use either the Spotify API or an open-source database called Muff. In the end, using two separate databases provided too much data. The amount of data made the data collection process difficult. For this reason, we decided to choose to focus on Spotify.

After finding the data source, we planned on finding all Finnish playlists with more than 10000 average monthly listeners. However, this proved more difficult than expected. The Spotify API has a limited number of inflexible functions. There was no way to classify whether a regular public playlist was Finnish or not, and there were far too many regular public playlists to cross-reference to see if they contained Finnish songs. These problems required us to shift our expectations of which playlists we could use. We decided to go to the Spotify API documentation to find what playlists we could collect.

After examining the Spotify API documentation, we decided that using Spotify's official playlists would fit best within the scope of the project. Spotify official playlists are handpicked playlists made by Spotify employees. Using the "Get Featured Playlists" function from the Spotify API allows us to fetch the names and IDs of Spotify official playlists depending on the target country (such as Finland). These playlists fit our needs well because Spotify's official playlists are curated playlists that get recommended to users with similar interests.

Unfortunately, the featured playlists were only a subset of all Finnish Spotify playlists. Calling get featured playlists would only give you 15 of the 1000s of official Spotify Finnish playlists, and there were no other functions to find the Spotify official playlists. To get around this, we implemented a method that would check for new featured playlists every day and add the playlist names and Ids to a dataframe as shown in figure 1. We then store that dataframe as a CVS.

	playlist_name	id
0	Ei tässä ole kiire mihinkään	37i9dQZF1DXbcCTrbdHmWr
1	Pehmeät klassikot	37i9dQZF1DWVIKZXJLByBS
2	Akustista tunnelmointia	37i9dQZF1DWYCTsyhNdNao
3	Herkimmät suomalaiset biisit	37i9dQZF1DWW1FLvEAcimu
4	#vainsuomihitit	37i9dQZF1DWUvzPS8uIABd
5	Poppia työpäivään	37i9dQZF1DX8JRb0iafpW2

Figure 12: Playlist Name example dataframe

	playlist_name	song_name	id	artist	popularity	danceability	energy	key	loudness	speechiness	mode	acousticness	instrumentalness	liveness	valence	tempo	duration_ms	time_signature
0	#vainsuomihitit	PLAYA	1vQoEeHAKr8OIGIMFY2yAj	Jami Faltin	51	0.859	0.565	8	-5.485	0.1560	1	0.35100	0.000005	0.1090	0.9570	142.065	128258	4
1	#vainsuomihitit	Ikävä meitä	10G4sUXO33FT5SyRTdE4GM	Miri	39	0.559	0.729	2	-6.657	0.1150	0	0.52800	0.000854	0.1090	0.4650	191.782	165342	4
2	#vainsuomihitit	Voimaa ja valoa	3F2Sm1D7m7uz1i0Ytrh19c	Yona	41	0.552	0.403	7	-9.076	0.0850	1	0.56600	0.000021	0.1020	0.4930	148.030	217854	4
3	#vainsuomihitit	Vapaa (Mestarit)	6xUyD4l2ToLbUuazyY1aHV	Ellinoora	39	0.294	0.726	10	-6.615	0.0472	0	0.00537	0.000049	0.1260	0.0387	104.926	365038	3
4	#vainsuomihitit	Kohta sataa - Vain elämää kausi 13	2FIcD6o29XKTcHCJpvTTiO	Jyrki 69	37	0.662	0.794	4	-7.423	0.0326	0	0.00554	0.742000	0.1680	0.6310	108.049	191200	4

Figure 21: un-Cleaned data frame example

After a few weeks of playlist collection, we used the IDs of the playlists to fetch all the songs in that playlist and compile a pre-processed data frame. To do this, we had to break the process into three steps: First, we fetched the IDs of each song in a playlist and added them to a dictionary. In the dictionary, the playlist name is the key, and the list of song IDs is the value. Second, we iterated through the dictionary and extracted metadata for each song. Finally, the metadata is added to our pre-processed data frame as shown in figure 2.

For data cleaning, the two biggest challenges were finding the values that are useful for classification and filling in the few null values. To get the metadata of a song, you need to make a get request to the Spotify API, and when making that request, you can specify which metadata you require. By default, the API would give you all the metadata as shown in figure 3. For our data, when requesting the song metadata from Spotify, we excluded retrieving characteristics such as track_href, analysis_url, and URI. Thus, the data was almost fully clean when it entered our pre-processing pipeline. Unfortunately, there were still some missing values we needed to fill in.


```
('Come Mi Guardi (con Madame, Coez & Bresh)',
 'Night Skinny',
 39,
 [{'danceability': 0.759,
   'energy': 0.555,
   'key': 7,
   'loudness': -8.065,
   'mode': 1,
   'speechiness': 0.174,
   'acousticness': 0.572,
   'instrumentalness': 0,
   'liveness': 0.109,
   'valence': 0.295,
   'tempo': 122.023,
   'type': 'audio_features',
   'id': '1IRllh4V0h8MUGEHROZvhe',
   'uri': 'spotify:track:1IRllh4V0h8MUGEHROZvhe',
   'track_href': 'https://api.spotify.com/v1/tracks/1IRllh4V0h8MUGEHROZvhe',
   'analysis_url': 'https://api.spotify.com/v1/audio-analysis/1IRllh4V0h8MUGEHROZvhe',
   'duration_ms': 213496,
   'time_signature': 4}],
 10000)
```


Figure 3: Song metadata example

To clean the missing values in the data, we first found the average song for each playlist and used those values to replace missing values. To do this, we went through the pre-processed data frame to find the average song for each playlist and stored those average songs in CSV format as shown in figure 4. At this point, we iterated through the processed data and replaced all missing values with the corresponding average value for their respective playlist. This data is now clean and ready to train a classifier to predict playlists.

	playlist_name	popularity	danceability	energy	key	loudness	speechiness	mode	acousticness	instrumentalness	liveness	valence	tempo	duration_ms
0	#vainsuomihitit	41.122449	0.635867	0.707776	5.367347	-6.577602	0.060626	0	0.130015	0.026540	0.186852	0.522670	127.142857	193065.102041
1	100 Suomi	39.428571	0.737429	0.661543	5.514286	-6.457971	0.131009	1	0.199008	0.029648	0.168846	0.582000	123.079586	174754.828571
2	Aitoa suomiräppiä	30.408163	0.740510	0.628816	5.500000	-7.342510	0.167695	1	0.192310	0.030923	0.139526	0.550743	111.945939	180812.928571
3	Best New Pop	55.698795	0.614855	0.535805	5.987952	-7.701169	0.072594	1	0.379486	0.023302	0.157684	0.422153	112.834795	201871.819277
4	Big Country	70.040000	0.579290	0.711130	5.440000	-5.548760	0.046439	1	0.165173	0.000799	0.162385	0.567040	130.064580	195850.250000
5	Bileräppiä	73.397727	0.767034	0.625057	5.193182	-6.679432	0.208866	1	0.152363	0.015006	0.173718	0.440665	125.418136	190304.329545

Figure 4: Average song for each playlist example

3. Learning Task and Learning Approach

In this project, we approach the learning task of making a classification model to find the playlist a song most suited for. In section two, we outlined how we compiled a CSV file that holds all the songs for each finnish Spotify official playlist. With this data, we knew our problem would be a classification problem. However, we did not know what classification algorithm would work best.

Next, we needed to find what classification model best fits our needs. Our data has over ten features, and each data point corresponds to a specific class, so we knew to use the supervised learning method for our model. Furthermore, after testing many different supervised classification models, we found neural networks seemed to fit the best to our needs. However, before we could figure out which neural network to use, we had to do more data processing.

Neural networks weigh each feature equally, so we needed to process and normalize the data features with integer values outside the range [0,1]. To normalize the numerical variables, we used a Min-Max Scaler that scales the value between 0 and 1. With these normalized values, we were now able to start testing which neural network would work best for our data.

After testing a few different neural networks, we found that multi-layer perceptron (MLP) had the best results. MLP works well for approximation, this made it a good fit for our classification problem. For our categorical data variables, key and mode were one-hot encoded and multiplied by 0.5 to be consistent with the continuous variables. In the end, each data point consisted of 13 features. Next, we started to tweak the parameters of the MLP model. We found that increasing the maximum iterations for the model to 2500 was the sweet spot between a good amount of training without overfitting. In the end, we achieved an accuracy of about 30%. This accuracy is good for our model because each playlist has a small number of data points, and many playlists have similar songs (which confused the classifiers).

Now that we had the working model, we needed to add some functions so that the model would fetch meaningful results for the end user in an easily callable way. First, we decided that the best thing for the end user would be for us to give them a range of playlists to pick from. To get that range, we made a function to return the top 5 predictions when given a song name. When testing this function, we found that even though the model's accuracy was 30%, the playlist the test data came from was in the top 5 playlists almost every time.

In hindsight, even though we did get our model working and outputting actionable data, there are some things that we would have liked to do differently. Firstly, the model would work better if we used a larger database or a different method for fetching the playlist data. Secondly, many of the playlists were similar. We could have grouped similar playlists into one class which would have made the training process much more effective. If we did that, the model would reach the correct playlist group more often rather than finding a wrong similar playlist; this would have helped with positive reinforcement learning for the neural network. Even though there were some things we would have liked to do differently, our model was now ready to be embedded into a website for our end user to use.

In appendix 1, you can find the GitHub repository where we have all our commented code and stored data.

4. Communication of results and added Value

To best communicate our results, we originally decided just using a GitHub repository coupled with a blog post would be enough. However, after further deliberation, we decided that this was not the best method of reaching our target audience. Our target audience is small-time Finnish

musicians, so they would likely turn away from having to run programs themselves, even with a blog post to help them. We decided that a website embedded with our model would be the most helpful for our target audience. You can find the website URL in appendix 2.

The website is easy to use and navigate. The home page has a Search bar that takes a Song URL from Spotify. After inputting the URL to the website, the site sends the URL to our backend. In the backend, a function converts this URL into a piece of data that works with our model. Then it sends the results from the model back to the Prediction page. The Predictions page then shows five recommendations based on the ML analysis. By clicking on the recommendations, users are sent to the playlists on the Spotify website. With this website, Finnish artists can easily find related Spotify official playlists for their new soon-to-be hit songs.

Unfortunately, the creation of the website was more difficult than anticipated. We made our website with Django, an open-source framework to create websites for python projects. Django works for all lightweight or medium-level applications and can scale to larger projects in the future. As none of our group mates have worked with website development, designing the website was challenging but fun to learn. Another difficulty we had was implementing a method to collect some song Identifier and convert it into data our back end could use. In the end, we decided to use Spotify song URLs as those are not hard for the end users to get and make processing the song data easy.

We encourage you (the reader) to try out our website found in appendix 2. For an example song, Amatimies by JVG works well (found at https://open.spotify.com/track/5KvlbPXF13g14tPnPuW93m?autoplay=true).

5. Conclusion and future work

Making the website and the playlist recommendation classifier was a fun and valuable learning experience. We debated a lot about what to do in this project as we wanted to make something actually useful to someone. In the end, we think we achieved that with have we did. In the future, we want to work on the website a bit more to make it more refined. We also want to make it more accessible to a general audience. Right now, because of the data we collected, the model only shows playlists for Finnish songs, but in the future, we want to expand on this idea to include playlists from other countries too. This way, we can have our idea help more people make their start in the music industry globally.

6. Bibliography

Dmitry, Pastukhov. “How Spotify's Algorithm Works? A Complete Guide to Spotify Recommendation System [2022]: Music Tomorrow Blog.” How Spotify's Algorithm Works? A Complete Guide to Spotify Recommendation System [2022] | Music Tomorrow Blog, 9 Feb. 2022, https://www.music-tomorrow.com/blog/how-spotify-recommendation-system-works-a-complete-guide-2022.

Spotify Legal Team. “Spotify Developer Terms.” Spotify for Developers, 27 Sept. 2021, https://developer.spotify.com/terms/.

Willman, Chris. “Music Streaming Hits Major Milestone as 100,000 Songs Are Uploaded Daily to Spotify and Other Dsps.” Variety, Variety, 7 Oct. 2022, https://variety.com/2022/music/news/new-songs-100000-being-released-every-day-dsps-1235395788/.

7. Appendix 1: link to project repository:

https://github.com/schmaigul/Predicting-Hits

8. Appendix 2: link to website:

https://powerful-journey-20089.herokuapp.com/