FINAL PROJECT

CAN WE PREDICT THE POPULARITY OF A SONG ON SPOTIFY?

PROJECT CONTENT

INTRODUCTION AND PROJECT SCOPE

As a final project, we wanted to learn and apply deep learning and AI. As fans of music, my project partner and I wanted to go for something we are passionate. Being around artists, we though about how the tech industry could help them. Spotify being the leader in term of music streaming platform, and having a well documented API, the decision was fast taken.

PROCESS AND ALGORITHMS

The popularity on Spotify is a number between 0 and 100 where 100 is the most popular, calculated on different factors like amount of recent streams for example.

Our first steps with Tensorflow and OpenCV, our first thoughts on the topic and how did we end up with this model? How our model is performing?

CONCLUSION AND NEXT STEPS

What conclusion and what results did we get? We did had time to finish our project on time, but improvement can be implemented. What did we learn, and how overall was the project? What blockers did we face?

INTRODUCTION

Project Scope

This is our final Project at our Data Sciences Bootcamp at WBS Coding School. It is a two and half weeks project with an end presentation to non technical audience. The presentation can be found in the Github Repo. This presentation is way more technical and explain the process and steps we used, and the issues we faced.

Introduction

Spotify, being the leader of music streaming platform, has a score called 'Popularity' for each song, artist or album. On top of it, the platform provide also audio features such as instrumentalness, energy, tempo, etc and a 30 secondes audio preview. With all of it, we wanted to see if a computer could analyse all these information and predict the popularity.

PROCESS AND ALGORITHMS

Data Collection

From where and which data did we decided to work with?

Audio Features

What are Audio Features and what did we do with them?

Audio Previews

How did we process audio files in our model?

Full Model

2 inputs models, what kind of algorithm?

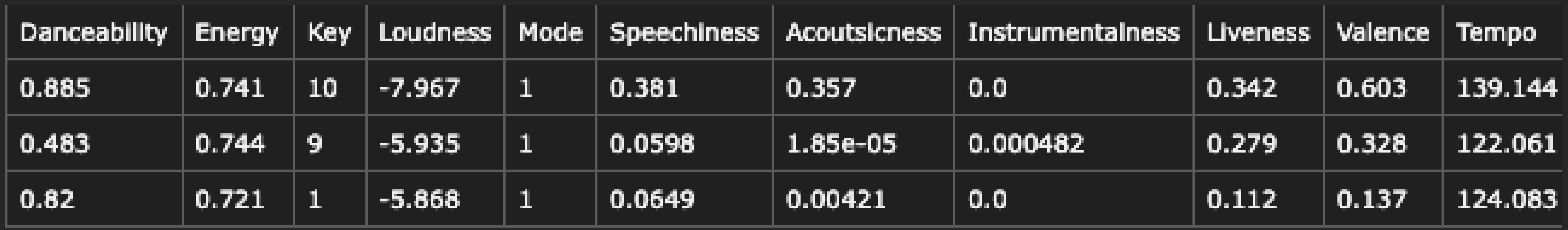
Data Collection

Thanks for Spotify's API, we could get all information for tracks. We started with thinking about how could we gather a good mix of popular, non-popular and different genres songs. Being myself a big Spotify listener, I know they have lot of playlists and some of them like 'Friday Discover' would be songs that are still not popular but promising. We then thought it would be pretty interesting to see the archive from this playlist, and we found it on Github from someone that keep track of it.
We could get a list of more than 16 000 ids of songs. With this, after creating an App in Spotify for Developper and getting credentials, we could create a dataframe with all information for each track like Artist, Title, the url preview for the audio and all audio features.

After this, we processed data cleaning and ended up with around 11 000 songs.

Audio Features

The audio features provided by Spotify are numbers and represent attributes of a song such as Energy, Danceability, etc... (More information here)

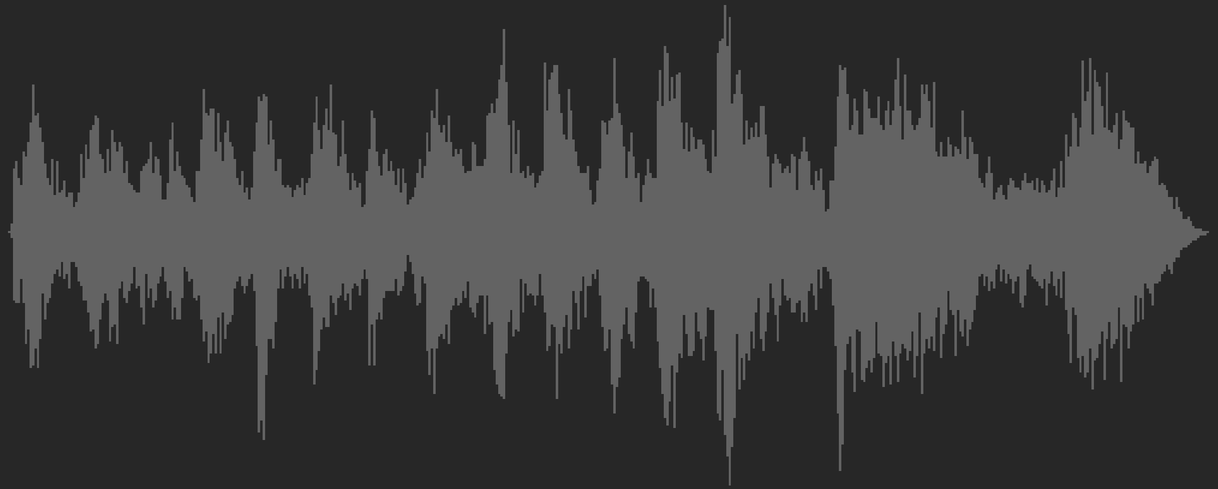


Danceability	Energy	Key	Loudness	Mode	Speechiness	Acoutsicness	Instrumentalness	Liveness	Valence	Tempo
0.885	0.741	10	-7.967	1	0.381	0.357	0.0	0.342	0.603	139.144
0.483	0.744	9	-5.935	1	0.0598	1.85e-05	0.000482	0.279	0.328	122.061
0.82	0.721	1	-5.868	1	0.0649	0.00421	0.0	0.112	0.137	124.083

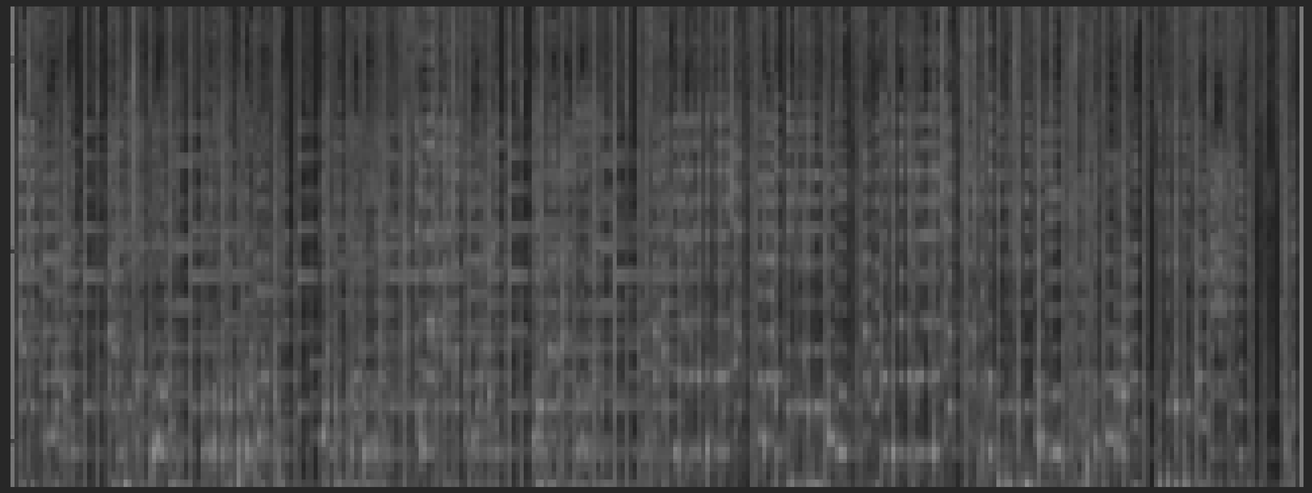
To start, we decided to build a model that will predict the popularity with only the audio features and see what results we get. We have tried to take few of my liked song to test the model on complete unknown songs from the model, and we had an average of 25 difference between the predictions and the real popularity.

Audio Previews

Audio Previews are given from Spotify. It is a link that we could download from with **Request** library in Python. This comes as an MP3, we converted the files to WAV, and from **Librosa** Library, converted them to Waveforms.



From Waveform, we created Spectrograms



With these images, we could then create Numpy arrays and give them to the model.

Full Model

We then had all what we needed to build our model, and to give different type of input.
We did a **Keras** model with two input streams - one for pictures and one for stats - and one output for a regression task. The picture stream uses a MobileNetV2 pre-trained CNN to extract features from an input image, followed by some convolutional and fully connected layers. The stats stream uses dense layers to extract relevant information from an input vector. Both streams are concatenated and passed through a fully connected layer, before producing the final regression output.

AUDIO FEATURES + AUDIO PREVIEWS

(numbers) (mp3)

CONCLUSION AND NEXT STEPS

Conclusion

We were actually surprise that the Audio Previews actually help the model to get better. This means that somehow, the audio features collected on Spotify are not complete.

While trying the model on random songs, we also have seen that the model perform on recent music. When a pretty old song is being predicted, predictions are bad.

Next Steps

We would have liked to develop a tool for artists to be able to have predictions on their own songs, like an application.

Also, it would be interesting to get the code to get audio features, to get a way to predict the popularity of a song even before to release it.

The model needs also to be trained with more songs, in particular old ones.

THANK YOU!

Github Repo

LinkedIn