Analysis

Introduction

My project aims to find a solution to being unable to find songs that are similar to those that a user already listens to. My program is aimed at a user base of teenagers who are attempting to broaden their music tastes.

Problem Identification

*“3.1.1 Problem identification*

*(a) Describe and justify the features that make the problem solvable by computational methods.”*

My user base may currently get recommendations from friends and family members who have a wider range of music knowledge. As such the music that they get recommended does not stem from a full music database and will not be as accurate as a fully functioning computational solution with a wider access to comparison material that others may never have heard of.

In order to make this solution compete with just asking some friends to share preferences it should be able to match, if not be better, the ability to recognise songs that are similar of a human.

To do this I can take a variety of approaches which are currently used in existing systems. I can either design it to base its recommendations off songs that users input as liking both of, effectively building a map of songs. The alternative is to use a numerical break down of the music to make numerical comparisons.

Features that allow the solution to be executed computationally:

Stakeholders of the Project

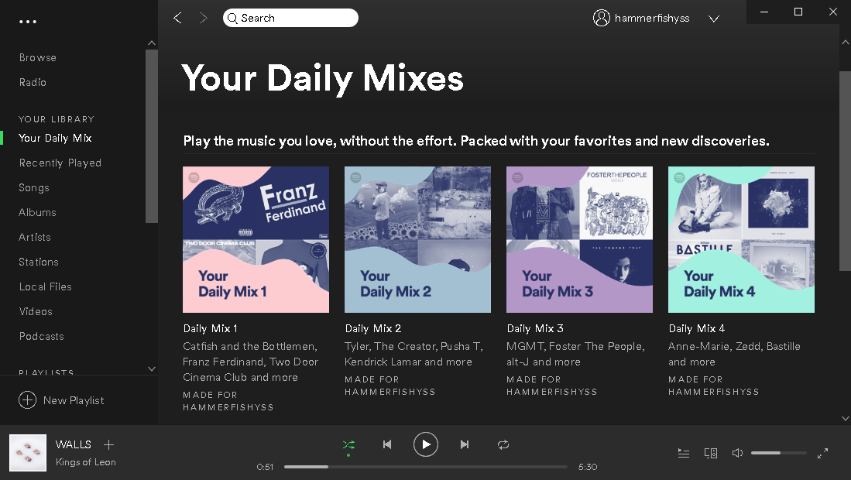
Research

*“3.1.3 Research the problem*

*(a) Research the problem and solutions to similar problems to identify and justify suitable approaches to a solution.”*

My project is not the first of its nature and as such many systems that provide similar experiences are available. Many music streaming services make use of algorithms to help listeners broaden their listening tastes. By analysing the ways that different programs present their data and how they reach their recommendations I will be able to make the best of previous designs and make a very accessible and flexible program.

1 – Spotify:

One of the major programs that makes use of music recommendation algorithms. As a very popular music streaming platform it attempts to broaden user’s music listening activity. One of their features is an automatic playlist generation that will match the user’s genre tastes, the “Daily Mix” playlists are auto generated, and such are powered by a powerful algorithm that can find songs that may not be related by artist but have a similar sound. They also provide a “Discover Weekly” playlist which as the name suggests is generated weekly. These suggest songs by artists that the user hasn’t listened to before, based on artists that they do listen to regularly.

Platform:

Spotify has been developed for a large range of different platforms due to its large user base. It has applications on the Google Play Store, Apple App Store and Microsoft Windows Store, on top of this it has a web player and applications for Windows, Mac OS, Linux and Chromebooks.

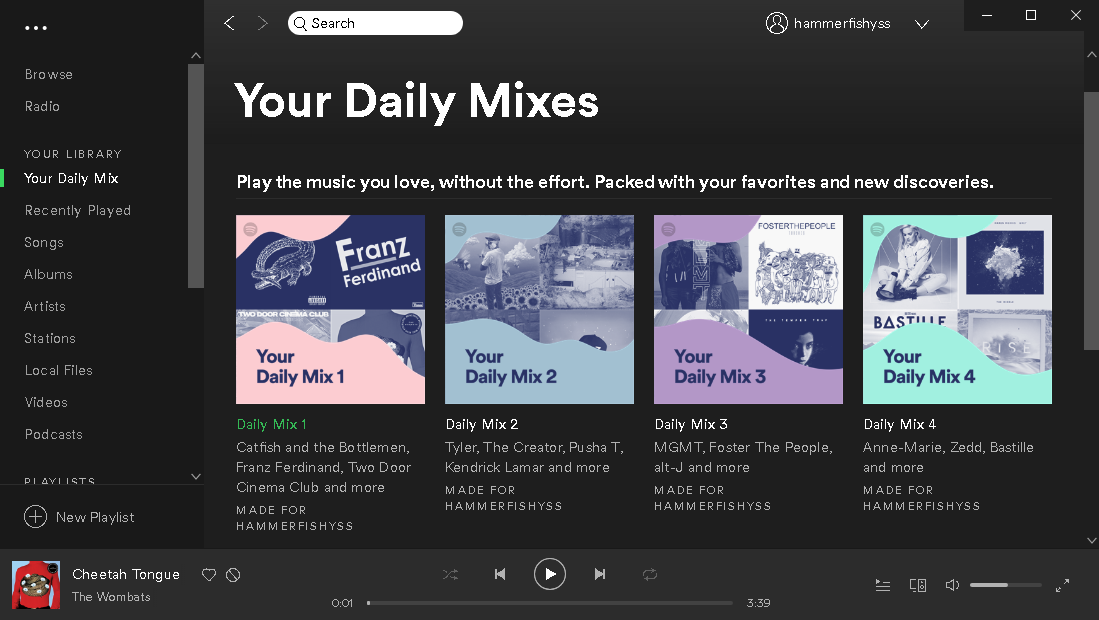
Price and Versions:

All these applications are free to download and use, but Spotify offer a Premium subscription-based service as well which costs £9.99/month. On top of the basic streaming functionality Premium allows users to listen un-interrupted by advertisements and skip an unlimited number of songs. The basic version of Spotify allows users to stream their music at 96kbps on mobile devices and 160kbps on desktop programs, this is upgraded by the premium version allowing users to stream at 320kbps.

These versions do not impact the recommendations the user gets, both versions have access to the personalised playlists.

Interface:

I will be primarily looking at the interface of the windows desktop application as that will be the platform my project will be developed for.

The user doesn’t get any choice in the songs that their recommendations are based on other than a like/dislike system by the song name when they are listening to a ‘daily mix’ or ‘Discover Weekly’ playlist.

The recommendations are presented in sets of playlists that are accessed by a side bar menu, this keeps the recommendation aspect out of the way of the main use (streaming music) but allows users who want to branch out an easy to use and dependable system.

Methods:

Spotify have a very advanced method of recommendation that considers multiple different ways of matching songs and then converge these into a verdict on whether the user will like them.

One of the main ways of recommending artists is based on a massive array of users and songs with Boolean values for each based on a like/dislike system for songs. This allows cross comparison of huge numbers of users to detect trends in shared taste.

The other method uses a neural network to decompose the auditory features of songs allowing songs to be recommended based on the similarity of the sound. This is a lot more advanced than simple trend comparison, allowing for different songs to be recommended outside of songs by artists that are linked to current favourites.

Sources:

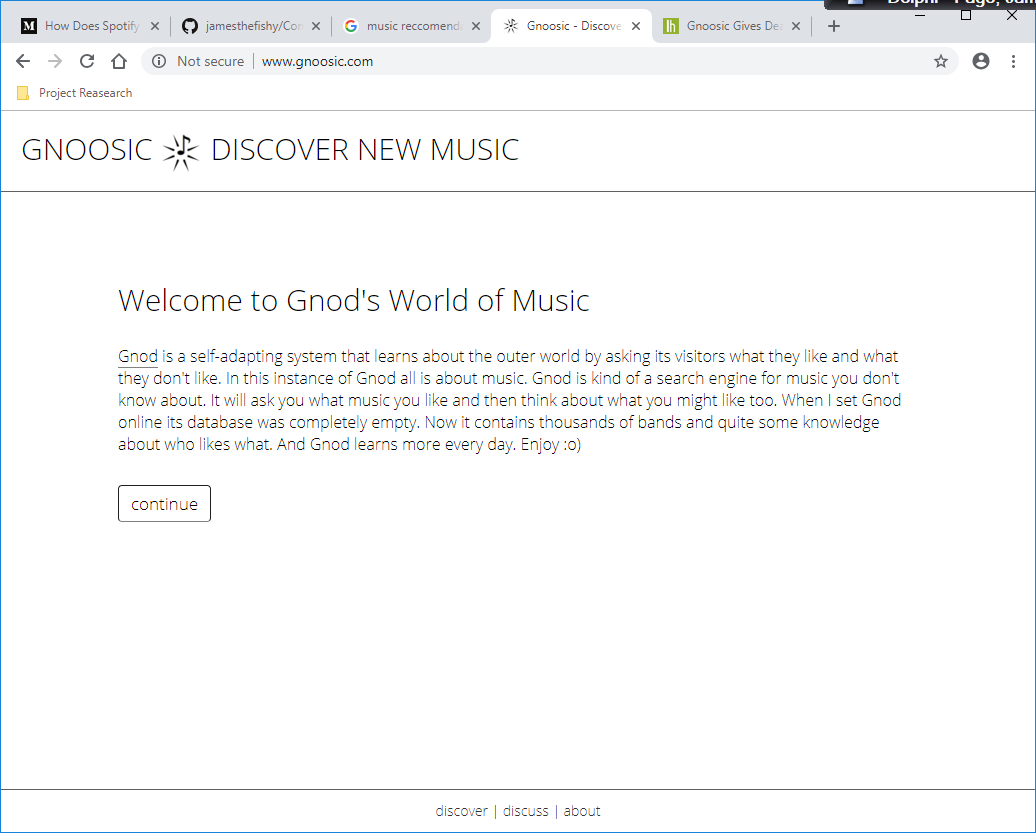
<https://medium.com/s/story/spotifys-discover-weekly-how-machine-learning-finds-your-new-music-19a41ab76efe>

<https://support.spotify.com/is/using_spotify/the_basics/what-is-spotify/>

2 – Gnoosic:

Gnoosic is web-based algorithm that takes in 3 artists from the user and recommends a different artist which the user can either select 'like', 'dislike' or 'Don't know'.

In comparison to Spotify it does not have as many features due to it being a simple recommendation site and not having the streaming functionality.



Platform:

Gnoosic is part of a collection of recommendation algorithms called The Global Network of Discovery, all of these services are web-based so available on all web connected devices.

Price and Versions:

As it’s a web-based program, and anyone can access it online, it is completely free for anyone to use.

It only has one version but as part of the Gnod group of programs there is a connected website called music-map which shows how the recommendations are linked.

Interface:

Gnoosic uses a simple white form user interface that allows the user to enter 3 artist choices and then shows you the name of a different artist that is linked to a the 3 entered by the user.

The connected site ‘music-map.com’ allows you to see the artists that are deemed ‘nearby’ to the selected artist in with the closer they are on the visual ‘map’ the more likely you are to like them.

There is also an account feature which allows the user to save their preferences and find more artists in future.

Methods:

This webpage makes use of similar techniques as the first point mention in the Spotify section. By asking users to enter 3 artists they enjoy the site adds links between those artists entered allowing the algorithm to ‘learn’ about music tastes:

*‘Gnod is a self-adapting system that learns about the outer world by asking its visitors what they like and what they don't like’*

Sources: <http://www.gnoosic.com/about>

*https://www.music-map.com*

Summary:

The two different programs I looked at take very different approaches to recommending users new music mainly due to the differences in the environment that the recommendation is applied.

As mentioned, Spotify being primarily a streaming platform means it has a different range of features that a simple recommendation program will not require. However, the ability to create playlists of the songs recommended to you is a very useful feature and the recommendation algorithm is a lot stronger and more like what I hope to replicate as it does not require a large amount of users to build an accurate recommendation.

The benefit of the Gnoosic system is that although the recommendations themselves maybe not as strong as the Spotify system, the simple interface allows you to find new songs based on an exact song rather than just your overall listening preferences. This alongside the account system saving your liked artists makes for simple and quick use, that is just as functional for a new user as for a long time user.

Features Expected:

Based on my research, to be able to compete with current solutions I should design my project to meet the following criteria:

* A simple and understandable user interface that allows the user to enter 2-4 songs they enjoy and would like recommendations based on.
* Songs recommended based on a neural network based algorithm using Spotify developer API to retrieve song data to use as input values.
* Train the neural network to find songs for each user saving the trained configuration to a database so the user can use it again.
* Save the songs found to return positive results when passed through the algorithm to a database so the finalised results can be recalled as a ‘playlist’.

Limitations:

One of the main benefits of Spotify’s solution is that due to the platforms pre-existing music streaming services the recommendations can be easily and quickly converted into playable playlists allowing users to get an audible result immediately rather than just a song name which is all my system would be able to achieve.

Proposed Solution