Visual Analytics for Music Recommendation System

Progress Report for CSE6242: Data and Visual Analytics

Adithi Jagannathan

Objective/Motivation

The goal of the project is to build an interactive and visual data analytics platform that will provide song recommendations to the users for creating playlists. This data driven system will perform the following:

- Provide music recommendations to the user based on the songs selected by the user in the recommendation context.
- This recommendation system will provide visual insights to users about music trends such as danceability, acoustics etc. and create a visual path to show each user interaction and results which leads to a final playlist.
- Each song recommendation in the playlist for the user in context will be presented using a graph. User selected song will be shown as the central node and recommended songs will be shown as leaf nodes. Then the user can pick a song from recommended leaf nodes and the process will repeat until the final playlist is created.

Problem Definition

From literature survey it was seen that the current research and analyses do not give importance to visualization as much as the recommendation algorithms. Also, a very narrow focus is on the interrelation between the recommendation application domain and visualization techniques. The current recommendation systems do not show a lot of information on the connection and grouping between songs selected by the user as the recommendations are provided by algorithms behind the scenes. An interactive and visual analytics platform for music recommendation systems will provide deeper insights into input datasets and output recommendations to provide personalized recommendations. Per literature survey this representational view does not exist, this approach is not focused or researched in any of the papers specifically for music recommendation system and effort is to solve this problem.

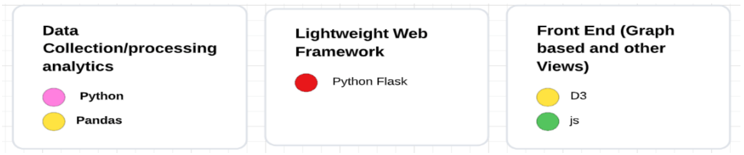
Literature Survey

Recommendation systems have been improved by complex algorithms that can process large datasets and provide recommendations that are useful and personalized to the users [5,6,13]. Many studies have been done to understand the advantages, disadvantages, and potential capabilities of recommender algorithms like context aware [2], Content based [1, 11], Collaborative [1,5,9], demographic [4], knowledge based [4], constraint based [7] and hybrid recommendation system [1,3, 4]. Research has also been conducted to study the effect of user-selected recommender algorithms [6]. Recommendation frameworks like Lenskit were developed to improve modularity, extensibility, reproducibly, etc. [17]. Some examples of the current systems' limitations include overspecialization [1, 11], cold start problems from new users and items [5], lack of information for effective user clusterizations [2], and scalability problems for large number of datasets and users [16, 18]. Very few research and surveys focus on users' representation and interaction with the system [3, 13], as well as the effect between an application's domain and the visualization techniques used [15]. Some focus on studying users' feedback from systems that visualizes inputs and outputs [3,13], others focus on specific visual representation techniques like tag cloud [14], node link and matrix-based graphs [8], tree maps [12], and time series and statistical data visualizations [10]. The application uses a less dense graph which is more representative than other methods [8].

Proposed Method

The proposed method is to create an interactive and visual analyst tool that enables a music listener to create a playlist based on interest. This system will allow the user to quickly visualize the data and get deeper insights and more personalized recommendations through the case-based recommender system. These tailored recommendations are the building blocks to creating the playlist.

The architectural tier of the project is shown below:


Data Collection/processing analytics
Python
Pandas
Lightweight Web Framework
Python Flask
Front End (Graph based and other Views)
D3
js

Intuition — why should it be better than the state of the art?

The following features of the analytics platform are innovative and make the approach distinguishable.

1. Existing recommendation systems do not provide enough explanations on why such recommendations are selected. In this approach the user can interactively view information about the key attributes which created this recommendation for each recommended song. They can then make a more informed decision on which song they would like to pick from the recommended list. The next set of song recommendations is driven by this interaction.
2. The recommendation engines can always be easily switched or combined with multiple recommendation engines and connected to this visual layer. User will also be provided access to different recommendation algorithms and given the ability to visualize the difference in the recommended items between the algorithms, the user can make choices based on the insights.
3. Multiple datasets are used to create a high dimensional recommendation system that provides a lot of data for each song in the system.
4. The platform will use different combinations of algorithms and perform A/B testing and evaluate user satisfaction.

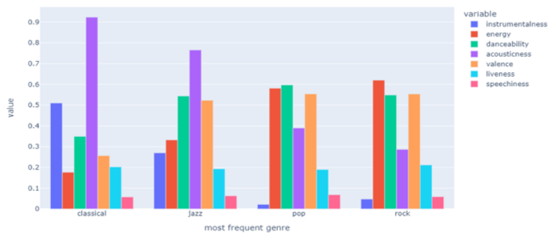
Data Extraction/Transformation

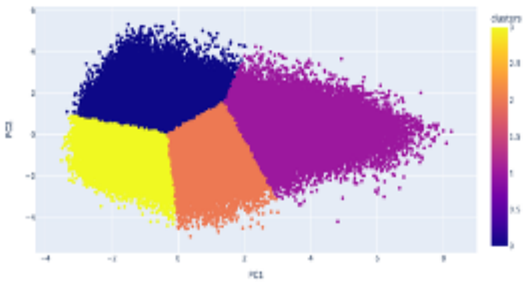
The data layer of the application is described below

- Collected the Spotify track features data and playlist data from Kaggle and aircrowd.
- Fetched the genres, listening count and playlist count using Spotify and LastFm REST APIs.
- Grouped and merged the data from the sources to create project dataset.
- The dataset has over 500,000 tracks, spanning 456 artists, 148 genres in different languages.

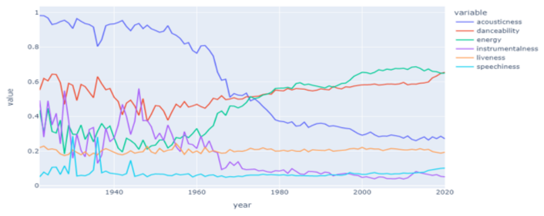
Data Preprocessing

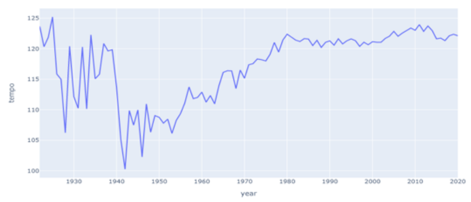
- Data preprocessing included extracting the discriminative feature that represent the songs and could be used by the algorithms that are predictive of user's interest.
- Visualizations used to understand the dataset are
  - For frequent genres visualize the average of audio features by year.
  - Clustered View of the important features of the dataset using PCA





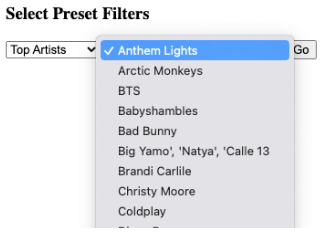
  - The music trend over the years by danceability, energy etc.

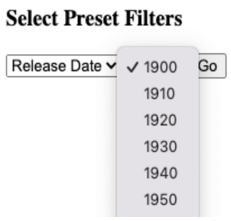


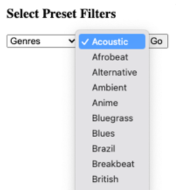


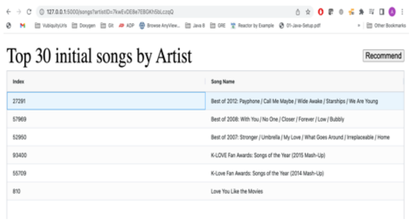
Application View/ User Interfaces

1. The user is first presented with a list of 30 songs based on user preferences like release date, genres, and top artists.





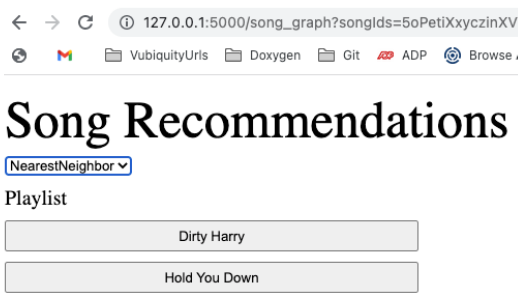



127.0.0.1:5000/songs?artistID=7kwEvDE8e7EBGKh5bLczqQ

Top 30 initial songs by Artist

Recommend

Index	Song Name
27291	Best of 2012: Payphone / Call Me Maybe / Wide Awake / Starships / We Are Young
57969	Best of 2008: With You / No One / Closer / Forever / Low / Bubbly
52950	Best of 2007: Stronger / Umbrella / My Love / What Goes Around / Irreplaceable / Home
93400	K-LOVE Fan Awards: Songs of the Year (2015 Mash-Up)
55709	K-Love Fan Awards: Songs of the Year (2014 Mash-Up)
810	Love You Like the Movies


127.0.0.1:5000/song_graph?songIds=5oPetiXxyczinXV

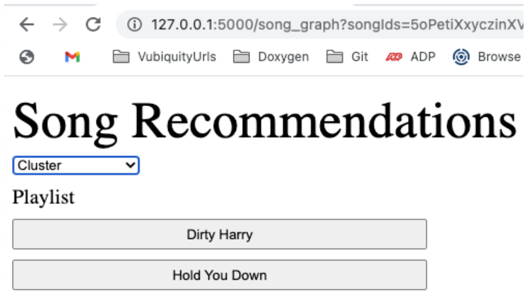
Song Recommendations

NearestNeighbor

Playlist

Dirty Harry

Hold You Down


127.0.0.1:5000/song_graph?songIds=5oPetiXxyczinXV

Song Recommendations

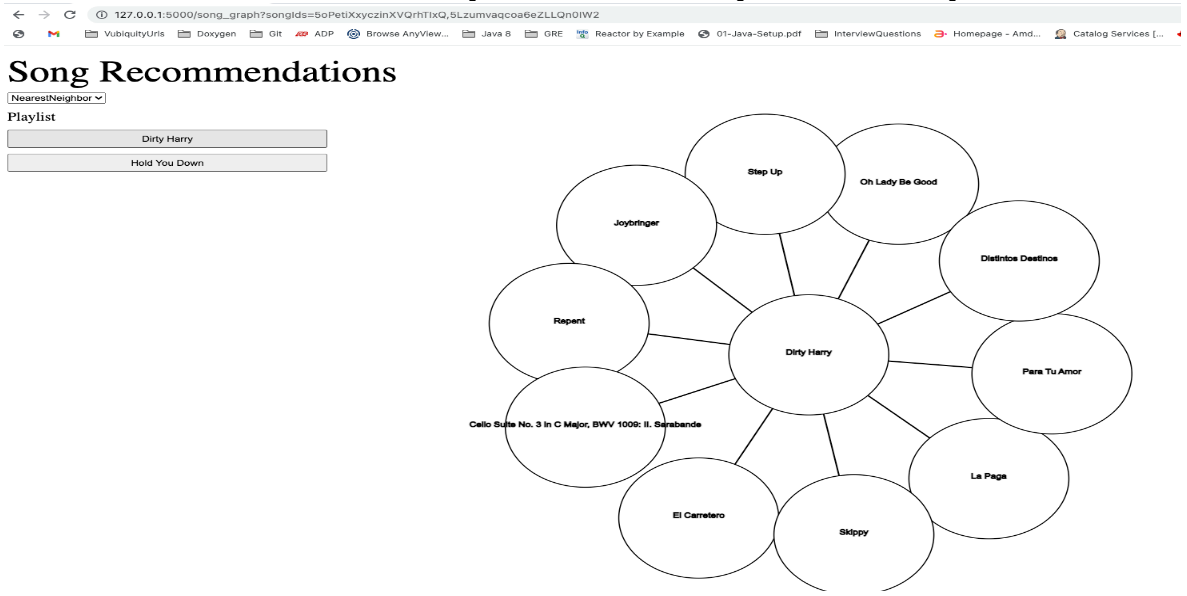
Cluster

Playlist

Dirty Harry

Hold You Down

2. On hovering, details of the song are shown to the user.
3. The user then clicks onto one of the songs based on interest, and the recommendation system will present another 10 songs based on similarity scores calculated in the backend. User can select between two recommendation algorithms – nearest neighbor or clustering.


127.0.0.1:5000/song_graph?songIds=5oPetiXxyczinXVQrhTIxQ,5Lzumvaqcoa6eZLLQn0IW2

Song Recommendations

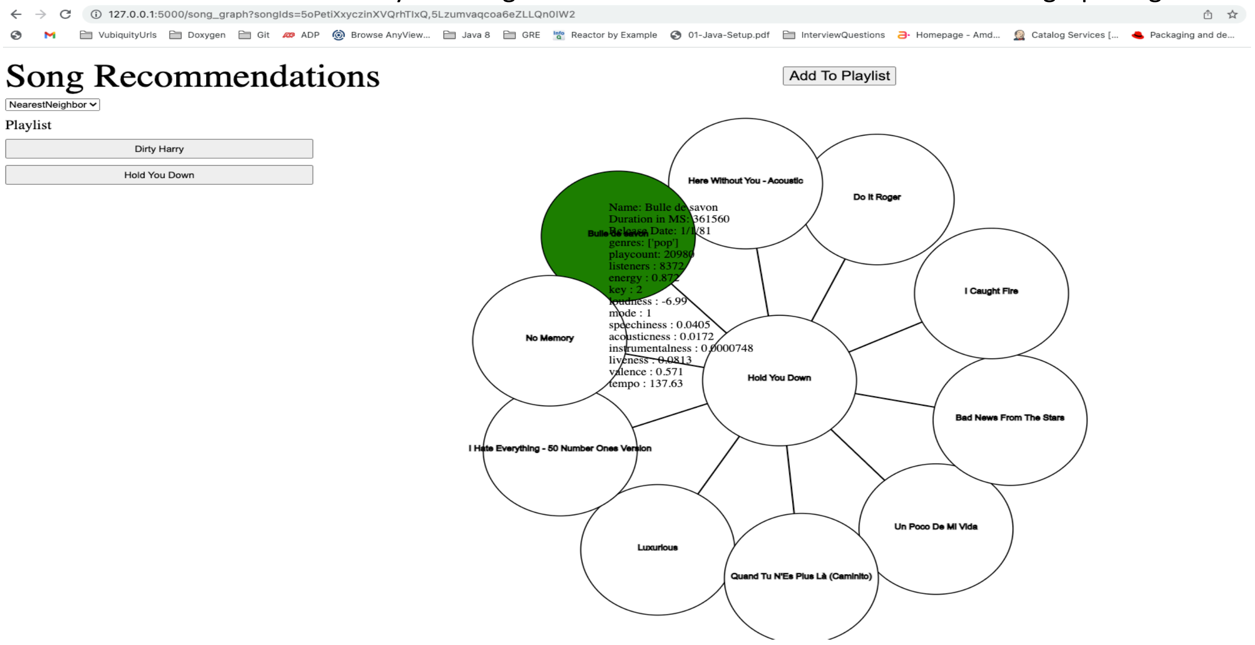
NearestNeighbor

Playlist

Dirty Harry

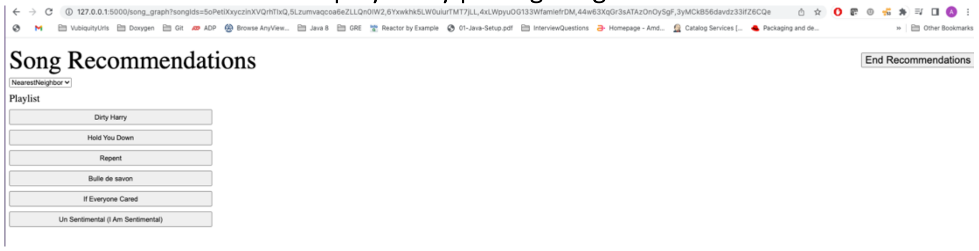
Hold You Down

4. The recommended songs will be shown to users using a graph-based method. The system will show attributes as to why the songs are chosen when the user hovers over the graph edge.

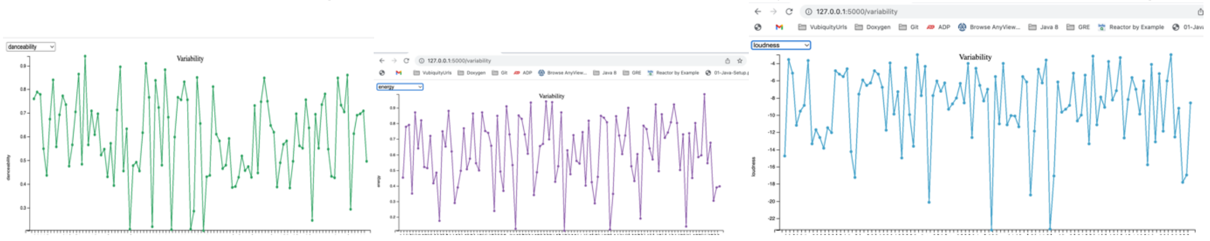


Name: Bulle de savon
Duration in MS: 361560
Release Date: 1/1/81
genres: ['pop']
playcount: 20980
listeners : 8372
energy : 0.872
key : 2
loudness : -6.99
mode : 1
speechiness : 0.0405
acousticness : 0.0172
instrumentalness : 0.0000748
liveness : 0.0813
valence : 0.571
tempo : 137.63

5. The user continues to build the playlist by picking songs in each round of the recommendation.



6. After the playlist is created, the user is provided with graphs that show features of the recommended songs. Users can then see the trends and patterns between selected songs.



Algorithms

The approach to provide song recommendation based on a user selection using the following algorithms that uses proximity between the documents for the recommendation.

1. Nearest neighborhood classification -> similarity between the tracks is measured using the cosine similarity function.
2. K- Means – Transform the data with PCA and cluster with K-Means and then apply cosine similarity on the tracks which are in the same cluster as the song chosen by user. This is an implementation decision taken.
3. K-prototype clustering - As the dataset is of mixed type use this type clustering and compared the accuracy with K-Means and found it is almost the same.

To evaluate the accuracy the following algorithms are used.

1. K means classification – To evaluate the cluster accuracy using F1- score
2. Recommendation Baselining - Use the Spotify playlist data and find the number of tracks that are in the same playlist from the recommendation engine.

Experiments/Evaluation

Data Preprocessing -Encoding

One hot encoding of genre resulted in increase in the dimension of the dataset as the number of different genres in the dataset was 4033.To decrease the dimensionality the following steps were followed: Understand the genre specific information and use the "genre seed" information from Spotify and categorize each genre in the dataset into one of the "genre seeds". The number of genres reduced to 128.

Recommendation Algorithm 1 - Nearest neighborhood classification

Distinctive audio Features of the dataset chosen to calculate similarity between items are danceability, energy, key, mode, speechiness, acousticness, instrumentalness, liveness, valence, tempo, and genres. Each feature is scaled and normalized based on its importance from studying the dataset and survey. From PCA and FAMD it was seen that the most important features in the dataset are energy and danceability. For the recommendation the danceability and energy is weighted twice.

Recommendation Algorithm 2 - Nearest neighborhood classification after clustering

The clusters were created using K means with only numerical features and K prototype clustering using the numerical and categorical variables. The clusters were cross validated using LGBM classifier and the cross validation F1 score for K-prototype clusters is 0.9897143733007867 and K-Means cluster is 0.978143733007867

This means that the tracks are grouped into meaningful and distinguishable features. Informativeness of clusters by SHAP feature importance. It was observed that the categorical variables are least important in clustering. So, the second recommendation algorithm is created with K-Means clustering

Evaluation

Analytic experiments on data

We conducted experiments on the data of 50000 drawn from a wide range of features.

The prominent artist in the playlist dataset is with id "3TVXtAsR1Inumwj472S9r4" who has songs in genres pop, metal, rock jazz, classical etc. From this it could be understood that these will be the favorite genres of the users.

Test Number of recommended Songs similar in both Recommendation algorithm

We ran the recommendation with different genres, release date, artist and it could be seen that overlap in the recommendations generated by two algorithms less than .01%.

Test with the Spotify Playlist Data

To measure the differentiation of the algorithm, compared it with the Spotify playlist data, as the test corpus, and verified how many of the recommended documents from each algorithm fall into the same playlist of the Spotify playlist dataset.

Feature weights	**Test Data**	**Number of tracks fall into same playlist**	**Overlap %**
Energy and danceability features are provided with twice the weight	Tracks with highest listening count in different genres	The playlist tracks don't belong to the same playlist in Spotify.	Per playlist the overlap =1/15
All features have the same weight	Track with highest play count for genre pop	3 Spotify playlists are present with 5 songs recommended by the algorithm within the same playlist.	overlap = 5/15.This shows that user add the songs of the same genres to the Spotify playlists.

The same above experiment was conducted for all the features and it was seen that the recommendation provided by this analytics platform is different from the Spotify recommendations. Our songs have high frequency of occurrence among multiple Spotify playlists.

<u>Measuring User Happiness</u>

To measure happiness of the user the measures are to calculate the relevance of search results using precision, recall and accuracy. The system should be live in production and save the user data to calculate these values to improve the recommendation.

Survey Results

For usability testing, I created a 4-question survey (see appendix) and interviewed 5 people as they went through the application. I found that:

1) Users like that can quickly spin up a playlist based on their current moods using attributes such as danceability, acoustic etc.
2) Users considered the genre as a prominent feature.
3) 25% of users agreed that at least 5 out of 10 songs presented are similar to their initial song.
4) 30% users agreed that the visualization is information and helpful in building the playlist.

Description of your testbed; list of questions your experiments are designed to answer.

- Correlation charts showed that there is a high positive correlation between energy and loudness and there is a high negative correlation between energy and acousticness. Also, there is considerable positive correlation between "valence" and "danceability", and negative correlation between "acousticness" and "loudness".
- Clustering experiments uncovered interesting groups of music based on energy and danceability.
- In the analysis of the behavior of two algorithms it was observed that cluster-based similarity algorithm recommends songs with high energy and danceability and the Nearest neighbor classification algorithms provide more generic recommendation. User can select the recommendation algorithm based on their interest.
- Testing with Spotify playlist data we concluded 1) The algorithm is fundamentally different from Spotify. 2) Recommends very popular songs to the users.

Conclusions and Discussions

Key conclusions derived from effort is

- Effective visualization is as important as the recommendation algorithm for users to have interest in a recommendation system
- Success of the recommender depends on the context and the user in context,
- As an overview this data analytics platform that uses data science techniques and interactive visualizations can help the users to get deeper insights to the data and improve users' trust.
- Future scope
    - provide different recommendations algorithms with specific tags or specific metadata attributes for calculating recommendation.
    - Save the user ratings of the playlist generated and use this data for parameter tuning.

References (Literature Survey)

1. Adomavicius, G., & Tuzhilin, A. (2005). Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. IEEE Transactions on Knowledge and Data Engineering, 17(6), 734–749. Retrieved from https://doi.org/10.1109/tkde.2005.99.
2. Adomavicius, G., & Tuzhilin, A. (2010). Context-aware recommender systems. Recommender Systems Handbook, 217–253. Retrieved from https://link.springer.com/chapter/10.1007/978-0-387-85820-3_7
3. Bostandjiev, S., O'Donovan, J., & Höllerer, T. (2012). Tasteweights. Proceedings of the sixth ACM conference on Recommender systems - RecSys '12 [Preprint]. Retrieved from https://doi.org/10.1145/2365952.2365964.
4. Burke, R., (2002). Hybrid recommender systems: Survey and experiments. User Modeling and User-Adapted Interaction, 12 (4), 331–370. Retrieved from https://link.springer.com/article/10.1023/A:1021240730564.
5. Burke, R., Felfernig, A., & Goker, M. H. (2011). Recommender systems: An overview. AI Magazine, 32(3), 13–18. Retrieved from https://ojs.aaai.org/index.php/aimagazine/article/view/2361.
6. Ekstrand, M. D, Kluver, D., Maxwell Harper, F., & Konstan, J. A. (2015). Letting users choose recommender algorithms. In Proc. of the 9thACM Conference on Recommender Systems (RecSys), 11–18. Retrieved from https://dl.acm.org/doi/abs/10.1145/2792838.2800195.
7. Felfernig, A., & Burke, R. (2008). Constraint-based recommender systems: Technologies and research issues. In Proc. of the 10th International Conference on Electronic Commerce (ICEC), 3, 1-10. Retrieved from https://dl.acm.org/doi/10.1145/1409540.1409544.
8. Ghoniem, M., Fekete, J. D., & Castagliola, P. (2004). A comparison of the readability of graphs using node-link and matrix-Based representations. In Proc. of the 2004 IEEE Symposium on Information Visualization (INFOVIS). 17–24. Retrieved from https://dl.acm.org/doi/10.5555/1038262.1038777.
9. Joorabloo, N., Jalili, M., & Ren, Y. (2020). Improved collaborative filtering recommendation through similarity prediction. IEEE Access, 8, 202122–202132. https://doi.org/10.1109/access.2020.3035703
10. Heer, J., Bostock, M., & Ogievetsky, V. (2010). A tour through the visualization zoo. ACM Qeue, 8 (5). Retrieved from https://dl.acm.org/doi/pdf/10.1145/1743546.1743567 .
11. Lops, P., de Gemmis, M., & Semeraro, G. (1970). Content-based recommender systems: State of the art and trends. SpringerLink. Retrieved from https://link.springer.com/chapter/10.1007/978-0-387-85820-3_3.
12. Richthammer, C., & Pernul, G. (2017). Explorative analysis of recommendations through Interactive Visualization. Lecture Notes in Business Information Processing, 46–57. https://doi.org/10.1007/978-3-319-53676-7_4
13. Richthammer, C., Sänger, J., & Pernul, G. (2017). Interactive visualization of recommender systems data. SHCIS '17: Proceedings of the 4th Workshop on Security in Highly Connected IT Systems, 19-24. Retrieved from https://dl.acm.org/doi/10.1145/3099012.3099014.
14. Seifert, C., Kump, B., Kienreich, W., Granitzer, G., & Granitzer, M. (2008). On the beauty and usability of tag clouds. 2008 12th International Conference Information Visualisation, 17-25.

Retrieved from https://www.semanticscholar.org/paper/On-the-Beauty-and-Usability-of-Tag-Clouds-Seifert-Kump/b50a6e0d5c52d1c7b184aa9357ebaedf9a314565.

15. J. Sänger and G. Pernul, "Visualizing Transaction Context in Trust and Reputation Systems," 2014 Ninth International Conference on Availability, Reliability and Security, 2014, pp. 94-103, https://dl.acm.org/doi/10.1109/ARES.2014.19.
16. Zhang, J., Wang, Y., Yuan, Z., & Jin, Q. (2020). Personalized real-time movie recommendation system: Practical Prototype and evaluation. Tsinghua Science and Technology, 25(2), 180–191. https://doi.org/10.26599/tst.2018.9010118
17. Ekstrand, M. D., Ludwig, M., Konstan, J. A., & Riedl, J. T. (2011). Rethinking the Recommender Research Ecosystem. Proceedings of the Fifth ACM Conference on Recommender Systems - RecSys '11. https://doi.org/10.1145/2043932.2043958
18. Mukund Deshpande and George Karypis. 2004. Item-based top-N recommendation algorithms. ACM Trans. Inf. Syst. 22, 1 (January 2004), 143–177. https://doi.org/10.1145/963770.963776

Other References

1. https://www.datacamp.com/tutorial/introduction-factor-analysis
2. https://towardsdatascience.com/famd-how-to-generalize-pca-to-categorical-and-numerical-data-2ddbeb2b9210
3. https://towardsdatascience.com/principal-component-analysis-pca-with-scikit-learn-1e84a0c731b0
4. https://towardsdatascience.com/k-means-clustering-using-spotify-song-features-9eb7d53d105c
5. https://predictivehacks.com/k-means-elbow-method-code-for-python/
6. https://towardsdatascience.com/clustering-algorithm-for-data-with-mixed-categorical-and-numerical-features-d4e3a48066a0
7. https://www.ics.uci.edu/~djp3/classes/2010_01_CS221/Lectures/Lecture14_01_Slides_CS221.pdf
8. https://antonsruberts.github.io/kproto-audience/
9. https://zachary-a-zazueta.medium.com/k-prototypes-clustering-for-when-youre-clustering-continuous-and-categorical-data-6ea42c2ab2b9
10. https://towardsdatascience.com/evaluating-clustering-results-f13552ee7603
11. https://library.ndsu.edu/ir/bitstream/handle/10365/32461/Evaluating%20the%20Effectiveness%20of%20Music%20Recommendations%20Using%20Cosine%20Similarities%20of%20Various%20Combined%20Feature%20Sets%20Consisting%20of%20Metadata%20and%20User%20Generated%20Tags.pdf?sequence=1&isAllowed=y
12. https://www.section.io/engineering-education/building-spotify-recommendation-engine/
13. https://towardsdatascience.com/how-to-build-an-amazing-music-recommendation-system-4cce2719a572

Appendix
Dataset Details
Dataset Files

Sl: no	File name	Size in MB
1	tracks.csv	111.4
2	SpotifyFeatures.csv	33.7
3	mpd.slice.54000-54999.json	33.3
4	mpd.slice.53000-53999.json	32.8
5	mpd.slice.52000-52999.json	33.2
6	mpd.slice.167000-167999.json	33.8
7	mpd.slice.166000-166999.json	34.2
8	mpd.slice.165000-165999.json	35.6
9	mpd.slice.164000-164999.json	33.8
10	mpd.slice.163000-163999.json	32.4
11	mpd.slice.162000-162999.json	34

Dataset REST APIS

SL NO	REST CALLS
1	https://developer.spotify.com/documentation/web-api/reference/#/operations/get-recommendation-genres
2	https://developer.spotify.com/documentation/web-api/reference/#/operations/get-several-audio-features
3	https://developer.spotify.com/documentation/web-api/reference/#/operations/get-track
4	https://developer.spotify.com/documentation/web-api/reference/#/operations/get-an-album
5	https://developer.spotify.com/documentation/web-api/reference/#/operations/get-multiple-albums
6	https://developer.spotify.com/documentation/web-api/reference/#/operations/get-multiple-artists
7	https://www.last.fm/api/show/artist.getTopTracks

Dataset Test Files

Sl: no	File name	Size in MB
1	mpd.slice.549000-549999.json	
2	mpd.slice.613000-613999.json	
3	mpd.slice.115000-115999.json	
4	mpd.slice.290000-290999.json	
5	mpd.slice.596000-596999.json	

6	mpd.slice.324000-324999.json	
7	mpd.slice.422000-422999.json	
8	mpd.slice.974000-974999.json	
9	mpd.slice.679000-679999.json	
10	mpd.slice.7000-7999.json	
11	mpd.slice.391000-391999.json	
12	mpd.slice.497000-497999.json	
13	mpd.slice.225000-225999.json	
14	mpd.slice.523000-523999.json	
15	mpd.slice.875000-875999.json	
16	mpd.slice.448000-448999.json	
17	mpd.slice.712000-712999.json	
18	mpd.slice.193000-193999.json	
19	mpd.slice.38000-38999.json	
20	mpd.slice.695000-695999.json	
21	mpd.slice.899000-899999.json	
22	mpd.slice.721000-721999.json	
23	mpd.slice.510000-510999.json	
24	mpd.slice.846000-846999.json	
25	mpd.slice.216000-216999.json	
26	mpd.slice.411000-411999.json	
27	mpd.slice.947000-947999.json	
28	mpd.slice.317000-317999.json	
29	mpd.slice.22000-22999.json	
30	mpd.slice.794000-794999.json	
31	mpd.slice.998000-998999.json	
32	mpd.slice.620000-620999.json	
33	mpd.slice.3000-3999.json	
34	mpd.slice.70000-70999.json	
35	mpd.slice.597000-597999.json	
36	mpd.slice.291000-291999.json	
37	mpd.slice.779000-779999.json	
38	mpd.slice.423000-423999.json	
39	mpd.slice.975000-975999.json	
40	mpd.slice.325000-325999.json	
41	mpd.slice.998000-998999.json	
42	mpd.slice.620000-620999.json	
43	mpd.slice.548000-548999.json	
44	mpd.slice.114000-114999.json	
45	mpd.slice.612000-612999.json	
46	mpd.slice.449000-449999.json	
47	mpd.slice.713000-713999.json	
48	mpd.slice.496000-496999.json	
49	mpd.slice.390000-390999.json	

50	mpd.slice.678000-678999.json	
51	mpd.slice.522000-522999.json	
52	mpd.slice.874000-874999.json	
53	mpd.slice.224000-224999.json	
54	mpd.slice.81000-81999.json	
55	mpd.slice.217000-217999.json	
56	mpd.slice.511000-511999.json	
57	mpd.slice.847000-847999.json	
58	mpd.slice.694000-694999.json	
59	mpd.slice.192000-192999.json	
60	mpd.slice.720000-720999.json	
61	mpd.slice.898000-898999.json	
62	mpd.slice.54000-54999.json	
63	mpd.slice.795000-795999.json	
64	mpd.slice.621000-621999.json	
65	mpd.slice.999000-999999.json	
66	mpd.slice.127000-127999.json	
67	mpd.slice.316000-316999.json	
68	mpd.slice.410000-410999.json	
69	mpd.slice.946000-946999.json	
70	mpd.slice.227000-227999.json	
71	mpd.slice.84000-84999.json	
72	mpd.slice.877000-877999.json	
73	mpd.slice.521000-521999.json	
74	mpd.slice.393000-393999.json	
75	mpd.slice.495000-495999.json	
76	mpd.slice.710000-710999.json	
77	mpd.slice.611000-611999.json	
78	mpd.slice.117000-117999.json	
79	mpd.slice.51000-51999.json	
80	mpd.slice.326000-326999.json	
81	mpd.slice.976000-976999.json	
82	mpd.slice.420000-420999.json	
83	mpd.slice.292000-292999.json	
84	mpd.slice.594000-594999.json	
85	mpd.slice.945000-945999.json	
86	mpd.slice.413000-413999.json	
87	mpd.slice.75000-75999.json	
88	mpd.slice.315000-315999.json	
89	mpd.slice.749000-749999.json	
90	mpd.slice.124000-124999.json	
91	mpd.slice.622000-622999.json	
92	mpd.slice.578000-578999.json	
93	mpd.slice.796000-796999.json	
94	mpd.slice.723000-723999.json	
95	mpd.slice.191000-191999.json	

96	mpd.slice.479000-479999.json	
97	mpd.slice.697000-697999.json	
98	mpd.slice.844000-844999.json	
99	mpd.slice.512000-512999.json	
100	mpd.slice.214000-214999.json	
101	mpd.slice.648000-648999.json	
102	mpd.slice.711000-711999.json	
103	mpd.slice.876000-876999.json	

Useability Survey Questions

1) On a scale of 1 to 5, how satisfied are you with the songs selected by the algorithms?
2) For the 10 new songs that show up based on your initial song of choice: On a scale of 1 to 5, do you think the new songs are similar to the song you have selected?
3) On a scale of 1 to 5, are the visualizations (e.g., tooltips, graphs) helpful when you're building your playlist?
4) Please explain briefly why the visualizations are helpful/not helpful.