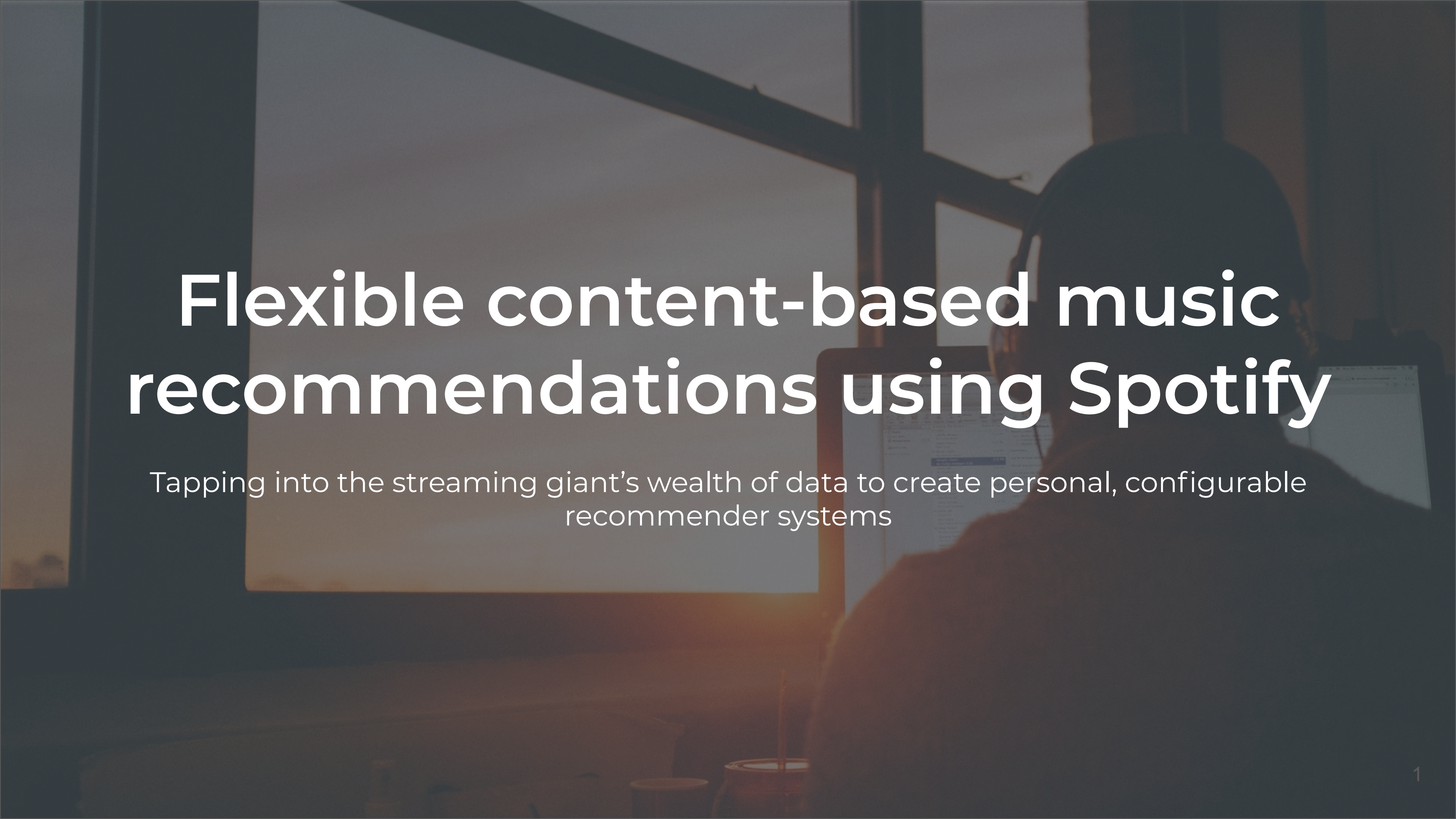
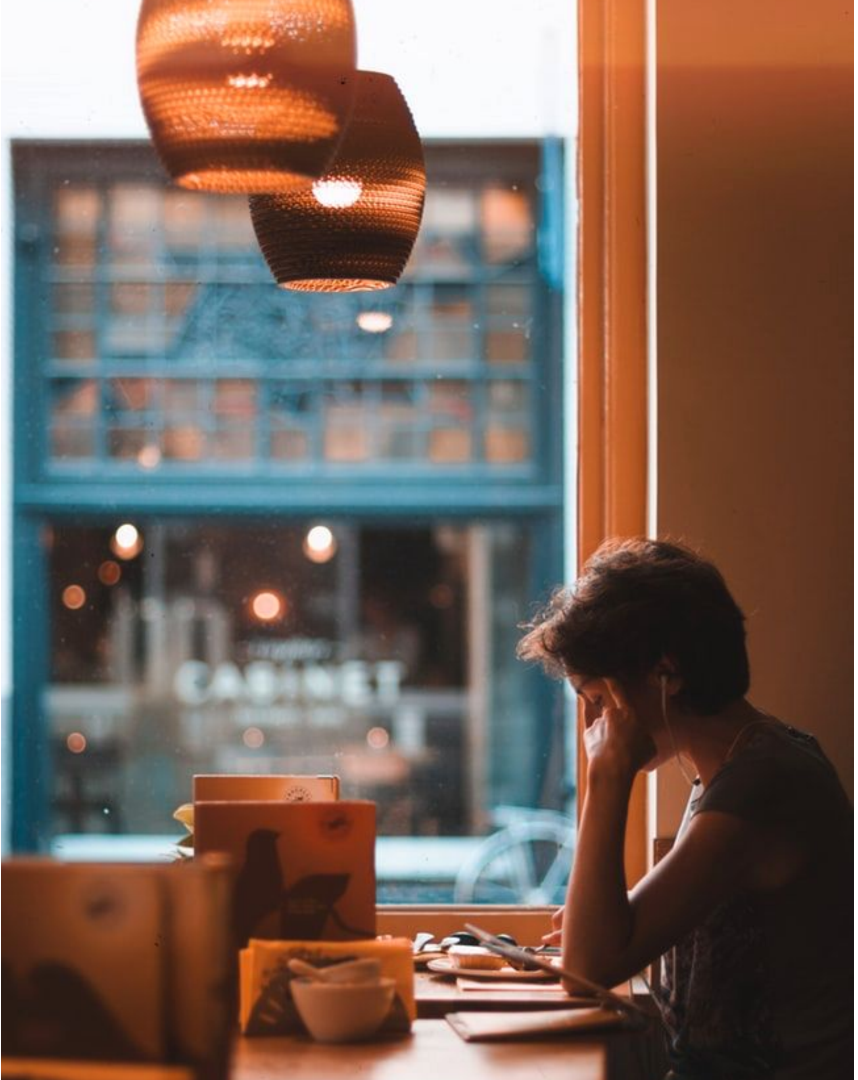


Flexible content-based music recommendations using Spotify

Tapping into the streaming giant's wealth of data to create personal, configurable recommender systems



Agenda.

> Problem definition

> Data wrangling

> Exploratory data analysis (EDA)

> Supervised learning

> Unsupervised learning

> Scalability and conclusion

When platform algorithms don’t “get” you.

> Platform recommender algorithms are opaque

> There is no interactivity

> What works well for most people, might not work for you

Reddit user u/Bleopping

Why is my discover weekly so bad?
Over the past few weeks I’ve listened to pretty much exclusively country songs and saved them to my library. Despite that, no country songs are appearing in my discover weekly.

Personal, configurable recommenders.

Key questions

1. How can we provide customizable content-based recommendations to users at scale?
2. Do supervised or unsupervised methods work better in this context?

Each of our models output:
3 themed playlists
10 songs each

Data wrangling

Acquisition, cleaning, feature engineering

We leveraged the power of the Spotify API to curate our own dataset.

Want
> library data
> user profile

Initialise Spotify API
> Request user authorisation
> Collect request tokens

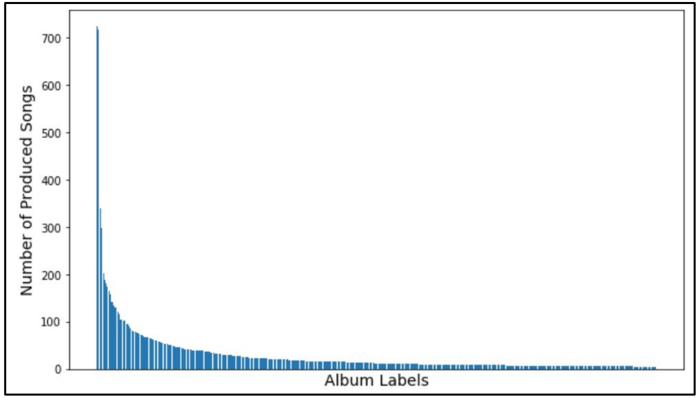
Iterative Requests
> Request user profile
> Scrape tracks and metadata

DATA!
> Preliminary cleaning

These are decisions we made for feature engineering.

Album label

- Raw data: Long tail for album labels
- Created categorical feature for big (> 80 songs from same label), medium, and small labels (< 11 songs from same label)



Long tail for album labels

Genre

- Raw data: List of sub-genres for each song
- Applied TfidfVectorizer on sub-genres, ran Truncated SVD on the Tfidf matrix, generated 13 genres with K-Means

name	artist_name	genre	genre_agg
Paradise	Bazzi	['pop', 'post-teen pop']	pop
Everything Has Changed	Taylor Swift	['dance pop', 'pop', 'post-teen pop']	pop
Boyfriend	Justin Bieber	['canadian pop', 'pop', 'post-teen pop']	pop
One Touch	Jess Glynne	['dance pop', 'pop', 'post-teen pop', 'tropica...	pop
Sexy And I Know It	LMFAO	['dance pop', 'pop', 'pop rap']	pop
Fergalicious	Fergie	['dance pop', 'pop', 'post-teen pop', 'r&b', '...	pop
Gods & Monsters	Lana Del Rey	['art pop', 'pop']	pop
Copy Cat (feat. Tierra Whack)	Melanie Martinez	['pop']	pop
human	Christina Perri	['neo mellow', 'pop', 'pop rock', 'post-teen p...	pop
Miss You 2	Gabrielle Aplin	['acoustic pop', 'dance pop', 'neo mellow', 'n...	pop
I Don't Belong In This Club	Why Don't We	['boy band', 'pop', 'post-teen pop', 'teen pop']	pop
It's All Coming Back to Me Now	Céline Dion	['canadian pop', 'dance pop']	pop

List of sub-genres clustered into genre_agg

Our dataset comprises 72 features on ~22k tracks.

Track information

- **Track**: 'name', 'popularity', 'uri'
- **Album**: 'album_name', 'album_popularity', 'track_number', 'album_uri', 'album_big_label', 'album_medium_label'
- **Artist**: 'artist_name', 'artist_popularity', 'artist_followers', 'artist_uri'
- **Genre**: 'genre_alternative metal/rock', 'genre_background', 'genre_baroque classic', 'genre_classical', 'genre_country', 'genre_hip hop', 'genre_house', 'genre_indie', 'genre_mexican', 'genre_pop', 'genre_rap', 'genre_rock'

Audio features

- **Characteristics**: 'acousticness', 'explicit','danceability', 'energy', 'instrumentalness', 'liveness', 'loudness', 'speechiness', 'valence', 'tempo', 'mode', 'duration_minutes'
- **Sections**: 'num_of_sections', 'num_of_keys', 'num_of_modes', 'num_of_time_signatures', 'section_durations_variance', 'section_durations_min', 'section_durations_max', 'section_loudnesses_variance', 'section_loudnesses_min', 'section_loudnesses_max', 'section_tempos_variance', 'section_tempos_min', 'section_tempos_max'
- **Time signature**: 'time_signature_1', … , 'time_signature_5'
- **Key**: 'overall_key_1', …, 'overall_key_11'

User specific

- **Playlist**: ‘recently_played', 'saved_tracks', 'top_tracks'

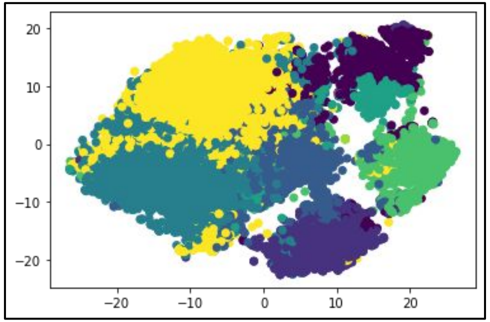
Exploratory Data Analysis

Clustering

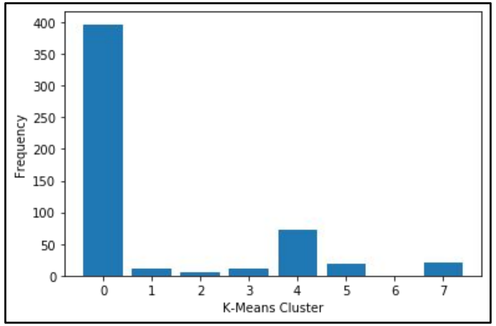
We applied different techniques to cluster songs based on their audio features to validate music genres.

	Clustering techniques	Insights
1	t-SNE	Identification of two big clusters and 5 smaller clusters after extensive tuning of learning rate and perplexity
2	DBSCAN	Generation of one cluster and outliers at maximum; limited applicability for this dataset
3	Agglomerative Clustering	Number of clusters = number of genres (13); observation of reasonable tendencies of clusters towards assigned genres
4	K-Means	Ideal number of clusters (8) < number of genres based on elbow method; observation of reasonable tendencies of clusters towards assigned genres

Boundaries of music genres are fluid, making it difficult to accurately cluster songs into genres.



t-SNE representation of K-Means clusters



K-Means clusters for music by J. S. Bach

- Found some proof that music genres can be created by clustering songs based on their audio features
- Although there are certain tendencies, there is no clear cut between music genres since boundaries are fluid
- Spotify tags their songs with multiple sub-genres, allowing them to capture the slight variations in music
- Aggregating these sub-genres to 13 genres is applicable for data exploration rather than more granular analyses

Supervised Learning

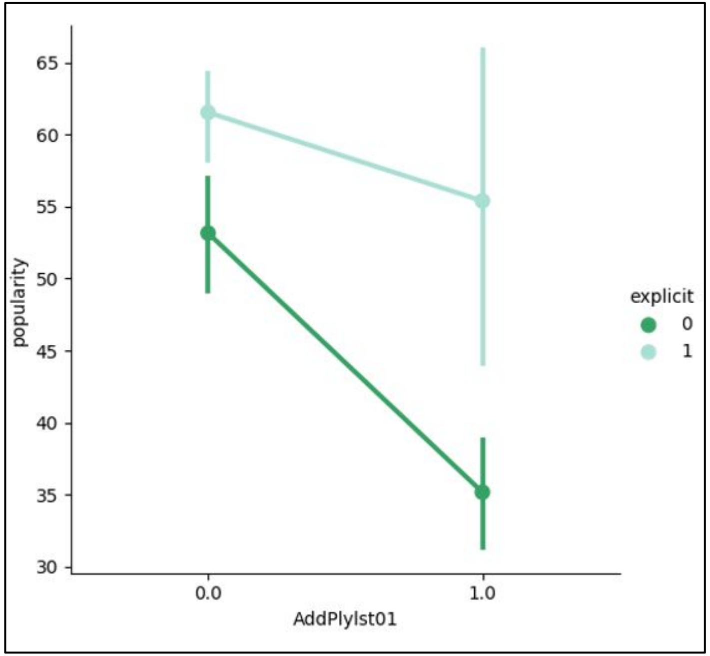
Classification, regression

We labeled data for supervised learning.

Problem: Our dataset is unlabeled

Solution:

- We create a random subset of tracks from both the the global data and the user music profile (saved tracks, top tracks, top artists)
- This new data set is randomized then ranked 0-5 and labeled 0 or 1 for like or dislike



Music preference by popularity and explicitness

Random Forest yields highest accuracy for classification-based recommendations.

Model	Accuracy
Logistic Regression	.453
Support Vector Machine	.453
K Nearest Neighbors	.640
Decision Tree	.640
Random Forest	.733
Random Forest with Grid Search CV	.747

Discussion:

- Best Accuracy: Random Forest with Grid Search CV
- This high performance suggests that labeled data may be an important aspect of Spotify's suggested songs

Regression-based recommendations.

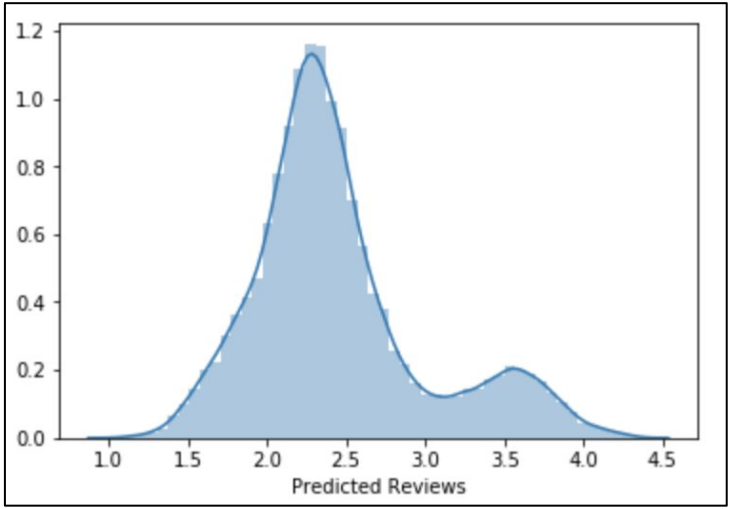
Model	RMSE
Decision Tree	1.49
Ada Boosting	1.35
Linear Regression	1.33
Decision Tree with Grid Search CV	1.32
Random Forest	1.31
Random Forest with Grid Search CV	1.30

Discussion:

- Best model is Random Forest with Grid Search CV has RMSE of 1.3
- Better results can be achieved with more labeled data from a user
- With a lower root mean squared error we can predict ratings with better accuracy which enhances the custom playlists for the user

Regression-based recommendations - Insights.

Distribution of Predicted Reviews



Insights:

- Distribution of predicted reviews are bimodal
- Prediction of reviews range from 1.5 - 4.0 and most fall between 2 - 2.5

Unsupervised Learning

Track-based recommendation, cluster-based recommendation

Recommend music based on tracks provided by the user.

Algorithm steps

1. User selects n number of tracks i.e. the all-time favorite track or all tracks in the music library
2. If $n > 1$, calculate mean of each audio feature across all tracks provided
3. Calculate cosine similarity between audio features of all tracks in our database and the input track(s)
4. Sort for tracks with highest cosine similarity
5. Filter tracks for the theme provided by the user
6. Recommend 10 songs for each playlist

Unique features:

- Interactivity: Engine is a unique way to explore new music
- User-friendliness: User defines which track(s) recommendations are based on
- Flexibility: Engine takes n number of tracks as input

Recommend based on clusters in user's listening history.

Algorithm steps

1. Fit many K-means clustering models to user listening history and PCA-dimension reduced version of user listening history with different number of clusters
2. Select K-means model and version of user data (either regular or dimension-reduced) with best silhouette coefficient score
3. Calculate cosine similarity between centroids of each cluster and every song user hasn't listened to
4. For each cluster centroid, select top *n* songs with highest cosine similarity such that output playlist is comprised of relatively equal parts of songs corresponding to each cluster

Unique features:

- Effortless: No input required from the user
- Discerning: Recommendations tailored to nuances of user preferences

Model Evaluation

Preliminary user testing, advantages and disadvantages

Preliminary user evaluation of output playlists.

	Dance	Chill	Discover	**Average**
Classification-based	2.2	2.8	3.8	**2.9**
Regression-based	2.0	2.9	2.7	**2.5**
Track-based (favorite song)	2.2	2.8	2.1	**2.4**
Cluster-based	1.7	2.0	1.8	**1.8**
Track-based (user profile)	1.1	2.6	1.0	**1.6**
Average	**1.8**	**2.6**	**2.3**	**2.3**

Trade-offs galore.

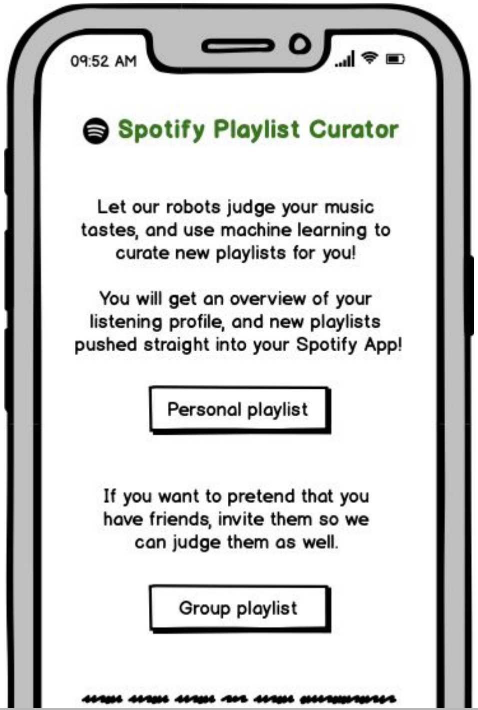
Model	Advantages	Disadvantages
Classification-based	+ Performed well in preliminary testing + Simple; gives users what they like + Can interpret contribution of individual predictors	- Requires labels - Lack of flexibility
Regression-based	+ Performed well in preliminary testing + Better predictions than unsupervised methods + Can interpret contribution of individual predictors	- Requires labels - High RMSE with a small dataset - Lack of flexibility
Track-based	+ No labeled data needed + Personalizable input song(s)	- Inaccuracies for unbalanced user profiles - No performance improvement over time
Cluster-based	+ No user input required + Balanced recommendations for users with diverse taste	- Inflexible - Clusters have no meaning outside model

Product Scalability

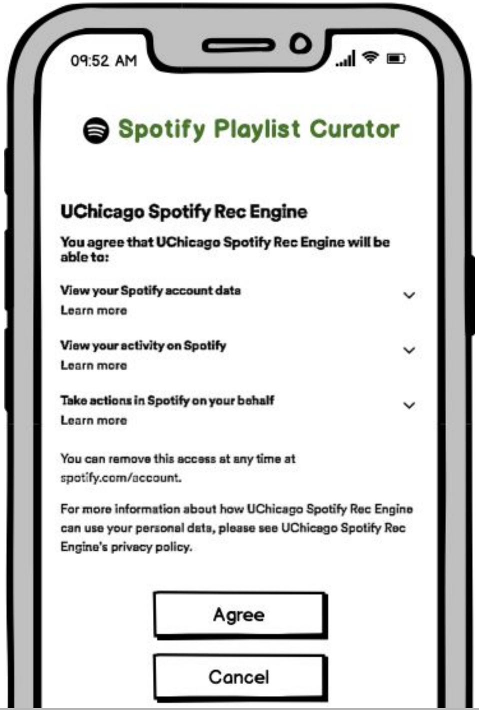
Productization of our solution

We have a scalable solution that can be hosted and coupled with an app or web interface.

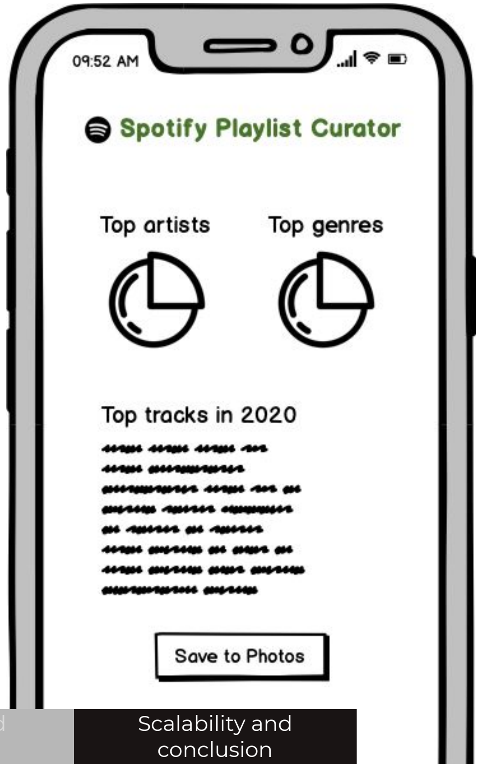
Provide individual and collaborative playlists



Handles Spotify API's OAuth 2.0 protocol

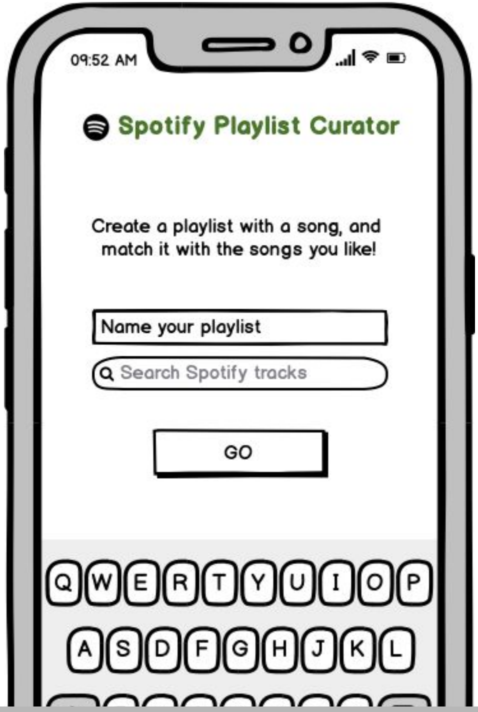


Provide insights into user' music profile

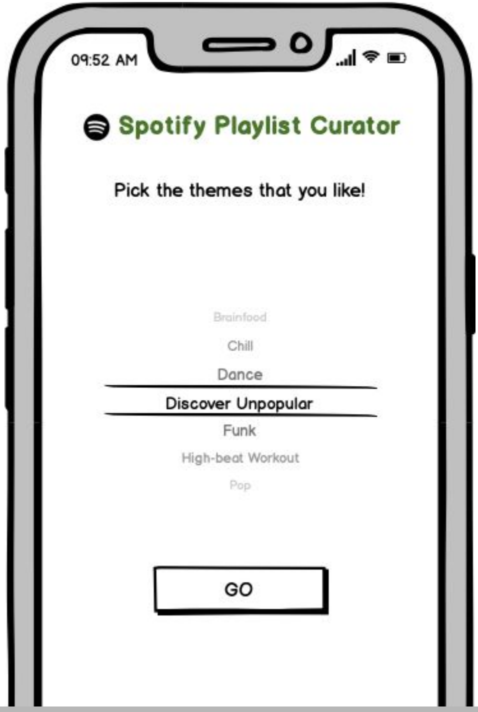


We have a scalable solution that can be hosted and coupled with an app or web interface.

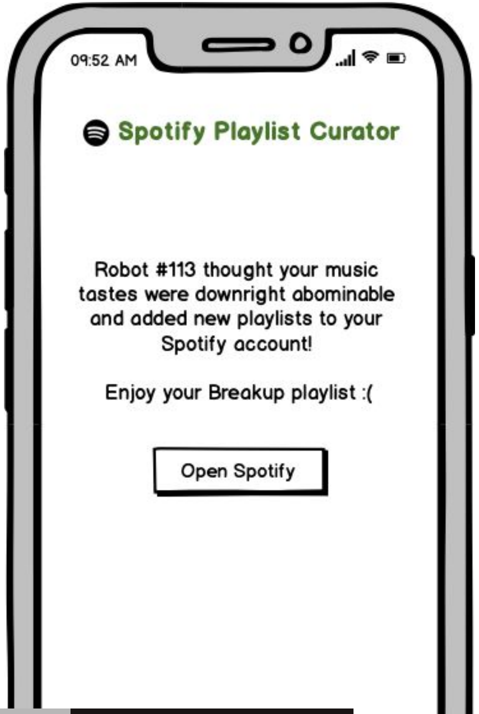
Provide track-based playlist curation service



Provide theme-based playlist curation service



Pushes new playlists into users' Spotify app



Conclusion

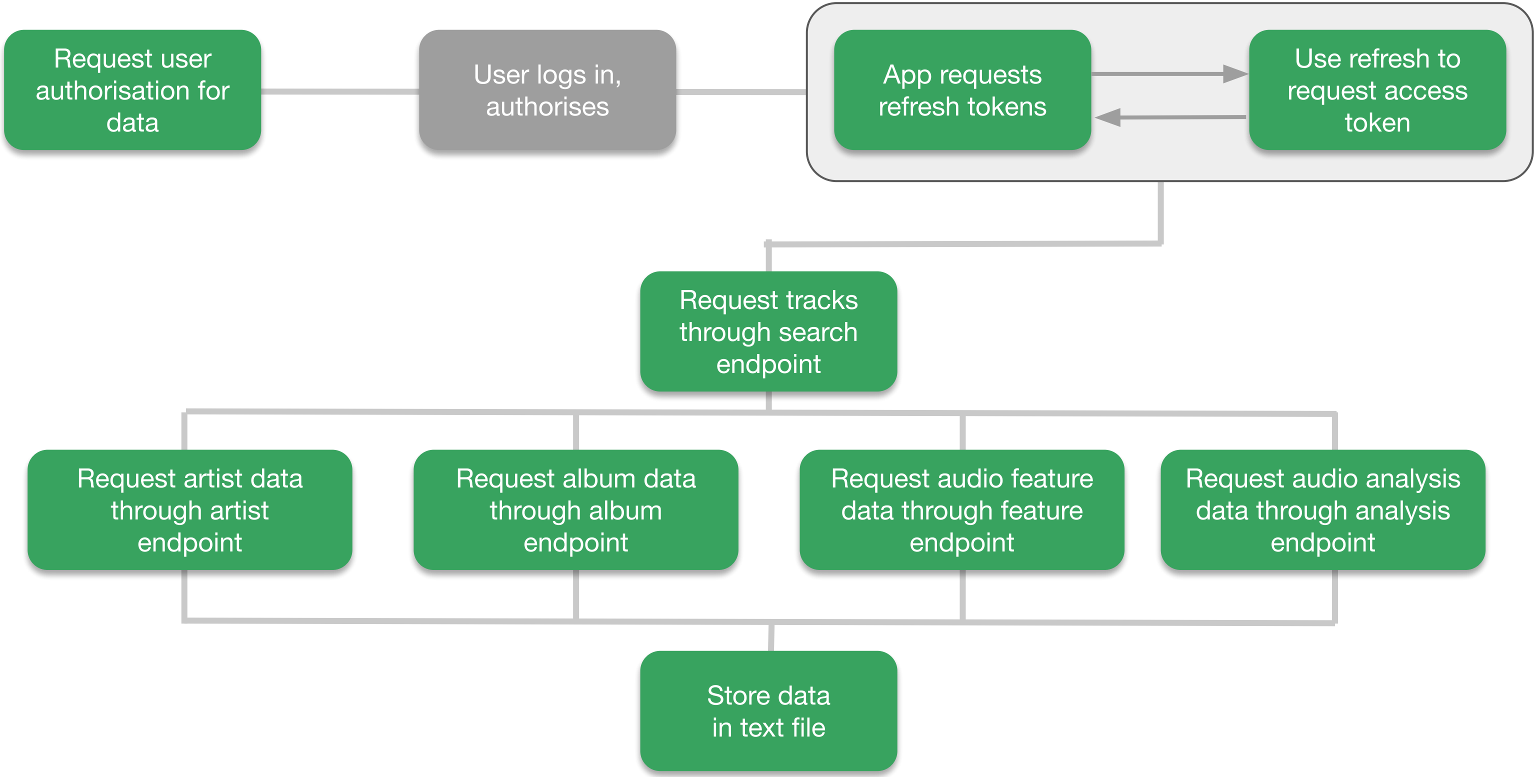
Recommendations, future work

Appendix

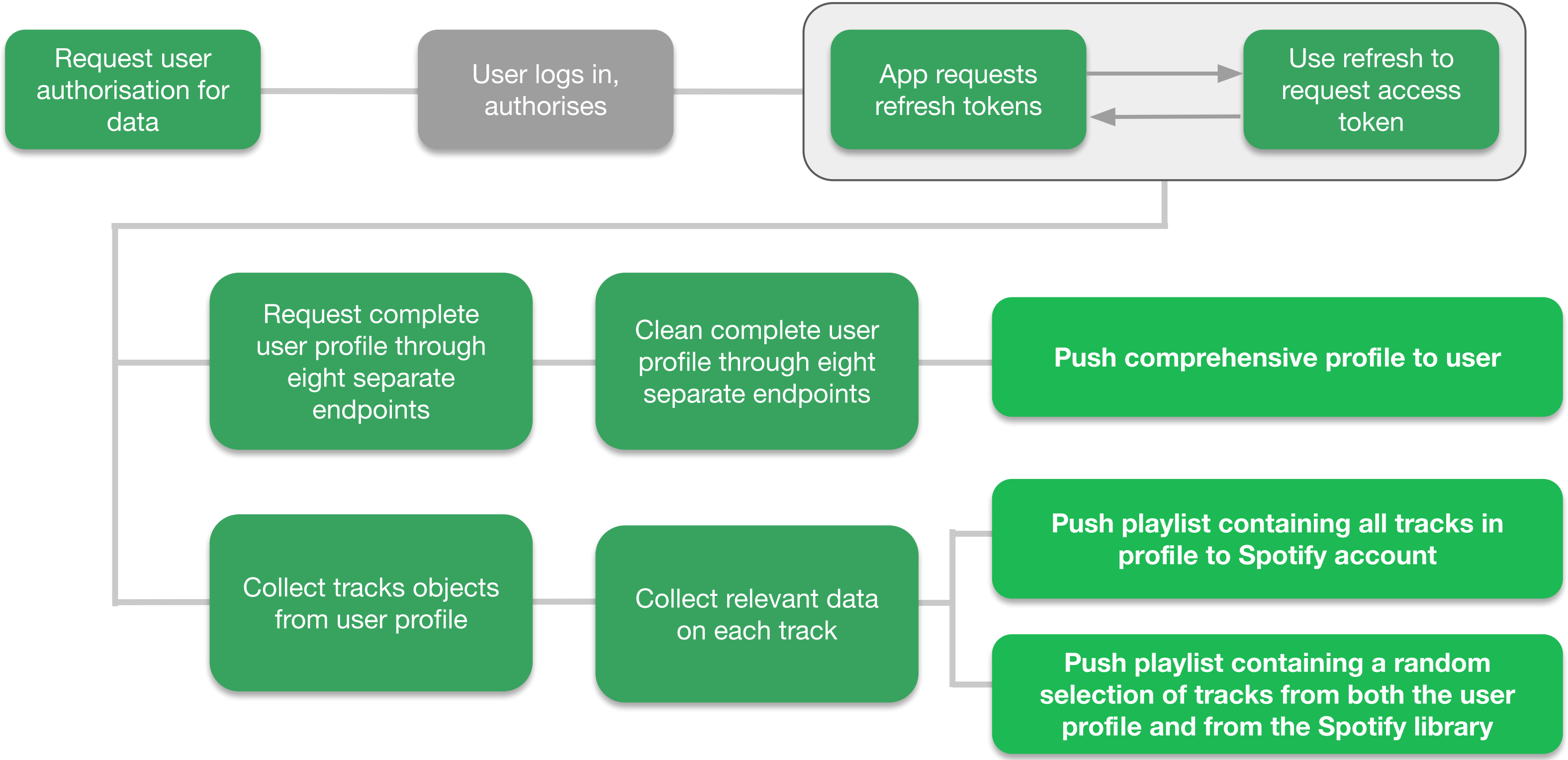
Conclusion/Recommendations.

1. Our approach:
   a. Clustering
   b. Supervised Models
   c. Unsupervised Models
2. Expansion:
   a. Youtube
   b. Netflix
   c. Pandora
3. Recommendations for Spotify:
   a. Expand the use of dislike button to all songs for labeling
   b. Use common genre to build themed playlists

We leveraged the power of the Spotify API to curate our own global dataset.



We leveraged the power of the Spotify API to collect user listening history and profile.



We focussed on feature engineering since our crawled data is rather clean.

1. Concatenate user and global data
2. Assign data types
3. Drop NAs
4. Drop irrelevant features
5. Categorize album labels
6. Convert release date in 2020 - x
7. Convert track duration to minutes
8. Extract relevant section values
9. Create string from list of sub-genres
10. Apply TfidfVectorizer on genre string
11. Run Truncated SVD on Tfidf matrix
12. Generate aggregated genres using K-Means
13. Drop duplicate tracks based on target user
14. Get dummies for categorical data

Silhouette coefficient

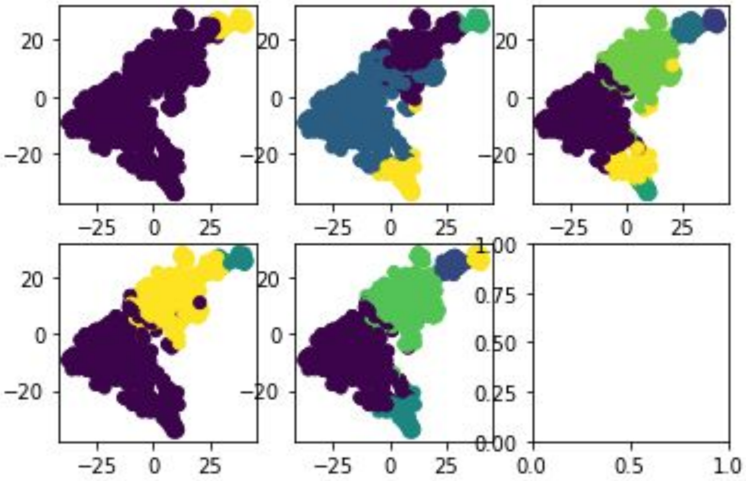
If the ground truth labels are not known, evaluation must be performed using the model itself. The Silhouette Coefficient (`sklearn.metrics.silhouette_score`) is an example of such an evaluation, where a higher Silhouette Coefficient score relates to a model with better defined clusters. The Silhouette Coefficient is defined for each sample and is composed of two scores:

- **a**: The mean distance between a sample and all other points in the same class.
- **b**: The mean distance between a sample and all other points in the *next nearest cluster*.

The Silhouette Coefficient *s* for a single sample is then given as:

$$s = \frac{b - a}{max(a, b)}$$

Iterating number of clusters



	data	clustering_method	silhouette_coeff	n_clusters_identified
0	user4_tracks	kmeans	0.648765	2
2	user4_tracks	kmeans	0.302631	4
3	user4_tracks	kmeans	0.265108	5
4	user4_tracks	kmeans	0.260709	6
1	user4_tracks	kmeans	0.249492	3