Music Recommendation with Audio-Based Features

Jefrey Judes

Contents

1. Introduction

Most recommendation algorithms, especially in the context of music, depend on *Collaborative Filtering* - inferring a users preferences from users with similar behaviour. This method is proven to consistently put the right things in front of the right people. While highly effective for the purposes of a recommender system – *recommendation* – collaborative filtering gives little insight into the musical features that define a users taste.

In this project we investigate recommendations on the basis of data related strictly to the music itself. By curating a set of features and observing how they describe a song, we would ultimately like to gain numerical insight into the highly subjective enjoyment of music.

The three primary goals for the project are as follows:

i. Obtain and carefully select a set of music features relating strictly to the audio composition of a song.

ii. Perform a thorough exploratory analysis of the features, exploring how they describe a given song or genre of music.

iii. Investigate the predictive power of these features across a variety of models for recommending a user music.

2. The Spotify Web API

In this project we will be pulling live data from Spotify using their developer API. The Spotify Web API gives access to a rich collection of data regarding audio, artists, playlists and users. While many of these channels will be of interest, we will be primarily using two categories of data: **audio features** and **audio analysis**. To obtain this data, we will be using the `spotifyr` package written by Charlie Thompson. Documentation for this package is available **here**.

Note: For the purposes of having the report compile in a reasonable amount of time, we pull a static version of the data acquired on April 26th, 2019. The code for acquiring live data is available in the source code (RMD) for the project.

2.1 Audio Features

Spotify's audio features are a high-level collection of metrics, like *danceability*, calculated for each song using proprietary algorithms. These features are based on the work of Tristan Jehan at Echonest, a company which has since been acquired by Spotify. We give a description of the 12 audio features we will be using below. This list will be an important reference for understanding the output of our analysis.

Feature	Type	Description
acousticness	`num`	A confidence level (0.0 to 1.0) of whether the song is acoustic – that is, whether it makes use of non-electronic instruments
danceability	`num`	A measure (0.0 to 1.0) of how suitable the song is for dancing. Factors contributing to this attribute include tempo, rhythm stability, beat strength, and overall regularity are considered.
duration	`int`	The length of the song measured in milliseconds (ms).
energy	`num`	A measure (0.0 to 1.0) of intensity and activity. Energetic tracks feel fast, loud, and noisy. Factors contributing to this attribute include dynamic range, perceived loudness, timbre, onset rate, and general entropy.
instrumentalness	`num`	A confidence level (0.0 to 1.0) of whether the song is (fully) instrumental – that is, whether the song lacks vocals.
liveness	`num`	A confidence level (0.0 to 1.0) of whether the song was performed live. This is predicted by way of listening for noise from audience.
loudness	`num`	Relative loudness (0.0 to 1.0) of the song. This feature has been scaled from a dB value provided by Spotify.
mode	`factor`	A categorical prediction (2 values) of whether the song is in a major or minor key.
speechiness	`num`	A measure (0.0 to 1.0) of what proportion of the song is spoken word. Genres like rap tend to be speechier than genres with sung vocals.
tempo	`num`	An estimate of the song's BMP (beats per minute).
time signature	`factor`	A categorical prediction (5 values) of how many beats are in each measure. Time signatures determine the rhythmic structure of the song.

Feature	Type	Description
valence	`num`	A measure (0.0 to 1.0) of the positivity conveyed by a song. High valence songs correspond to happy or euphoric music, whereas emotional ballads and angry music have low valence.

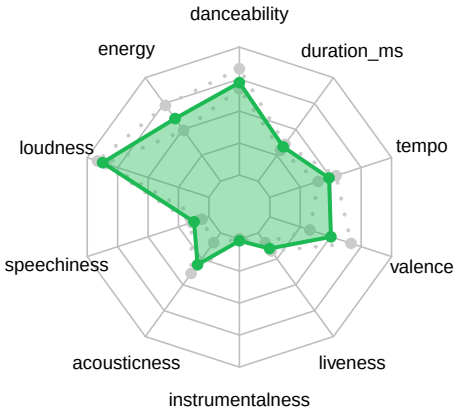
Notes:

- Some available features were omitted. The features `id`, `uri`, `track href`, `analysis url`, and `type` were metadata that have nothing to do with the songs themselves and were therefore omitted.
- The feature `key` which estimates the musical key of the song (e.g. F or C#) was omitted. The key is a rather arbitrary feature of a song which one should not expect to have any significance to its enjoyment or categorization.

A full listing of features and descriptions are available via **Spotify's website**

2.1.1 Audio Profile Graph

When comparing the audio features for several songs simultaneously, it will be advantageous to have a unified graphical summary that we can use throughout the report.



Above is a spider plot with 3 distinct lines: the green line represents the means for each numerical feature across all songs and the grey lines above and below represent the first and third quartiles. This will allow us to observe the average value and a variance of each feature at a glance. We will refer to this graph henceforth as the "Audio Profile". The above graph is the audio profile of the current Global Top 50 chart.

2.2 Audio Analysis

The audio analysis is a low-level collection of data which infers musical information about the song. The full details about the structure of the dataframes and the music theoretical concepts that underly them are described in detail on **Spotify's website**. For brevity, we will simply list the objects and what they describe below:

Object	Type	Description
bars	`dataframe`	A bar (or measure) is a segment of time defined as a given number of beats. The full count *one, two, three, four* constitutes one bar.
beats	`dataframe`	A beat is the basic regular count in music; each of *one*, *two*, *three* and *four* constitute four beats.

Object	Type	Description
tatums	`dataframe`	The smallest discernible subdivisions of a single beat.
sections	`dataframe`	Sections are defined by large variations in rhythm or timbre, e.g. chorus, verse, bridge, guitar solo, etc.
segments	`dataframe`	Segments attempts to subdivide a song into many segments, with each segment containing a roughly consitent sound throughout its duration.

This data is in its current form too primitive to constitute features that are useful for analysis – each observation (song) consists of 5 large data frames of varying dimension. Going forward, we will need some degree of feature engineering to construct meaningful features from the audio analysis.

3. Data Sampling

Our first task is to retrieve data from Spotify's Web API. Spotify (at the time of writing) has a musical library of over 40 million songs – far beyond feasible and necessity for our purposes. Moreover, the API places limits on the volume and velocity of requests, so we will choose to be modest in the amount of data we are retrieving. To that end, we are tasked then with choosing which subset of data we will collect from Spotify's library.

Fortunately, Spotify is best known for it's meticulously curated and well-loved playlists. When it concerns genre, these playlists are updated several times a week and can therefore dependably be thought to represent the current state of the genre.

We will therefore sample music from selected popular playlists on Spotify. Let us list those playlists below:

Genre Playlists: Classical Essentials, All About Country, Main Stage (EDM), Hip Hop Central, State of Jazz, Kickass Metal, R&B Right Now, Rock Your Block.

4. Exploratory Analysis of Audio Features

In this section, we thoroughly explore the distribution and characteristics of Spotify's audio features. The way Spotify calculates these features is unknown to us, so it is especially important that we have some numerical understanding of how they are describing the data.

Note: The given descriptions of each of the 12 features are listed above in $\S 2.1$.

Many of the metrics provided numerically describe subjective descriptions of songs. "Danceability", for example, would obviously vary based on who you ask. Moreover, the exact criteria for "danceability" would reasonably change based on the genre of music – a danceable country song might be quantified differently than a danceable hip hop song. It should therefore be granted that these measures don't exactly describe the quality they are named after, but rather some musical feature that correlates closely with the description.

With that in mind, let's see if the feature values roughly match our intuitions by examining the values for a well known song. The following are values for **"Don't Stop Believin"' by Journey**.

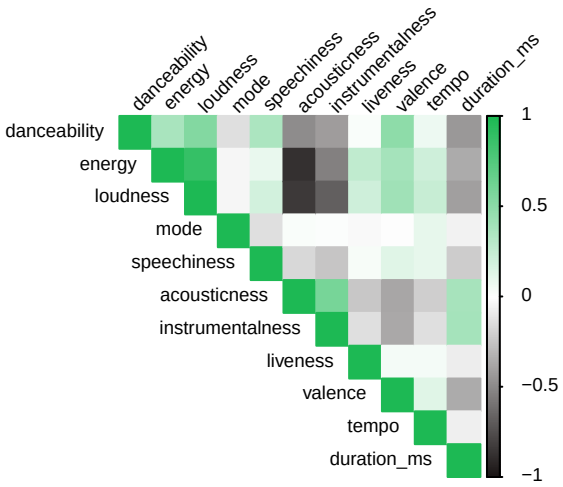
danceability	energy	loudness	mode	speechiness	acousticness
0.5	0.748	0.81856	1	0.0363	0.127

instrumentalness	liveness	valence	tempo	duration_ms	time_signature
0	0.447	0.514	118.852	250987	4

We see that these values do roughly match our intuition – *Don't Stop Believin'* is a loud, high-energy, feel-good (valence) song. As the features suggest, the song is indeed not instrumental, acoustic, or live.

4.1 Correlation Between Features

As due diligence requires of us, we begin by examing relations between each defined numerical feature. Using all the songs fetched in §3 let us compute and plot a correlation matrix.



Note: For the purposes of computing correlation, it is unproblematic to treat `mode` as a binary numerical variable. However `time_signature` must be omitted since it is not binary and no numerical translation would be fitting.

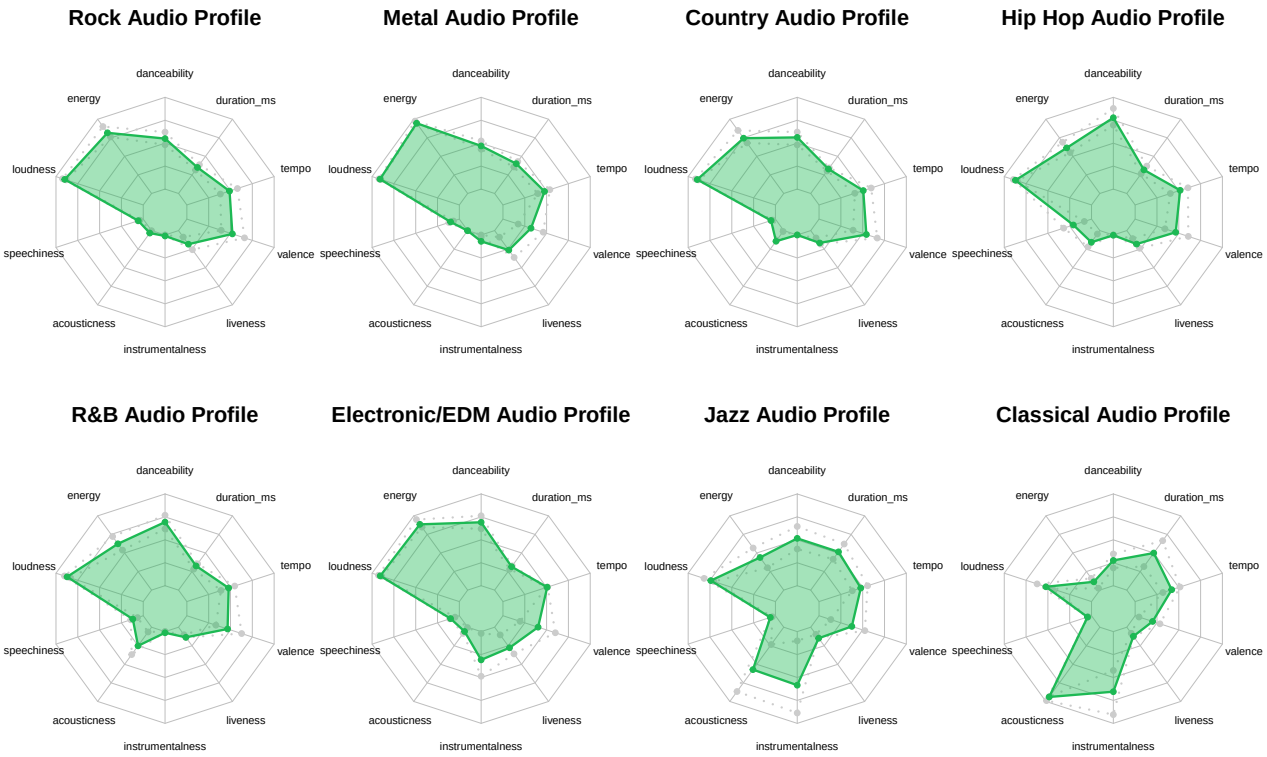
In the above plot we see that our variables are quite comfortably uncorrelated. Most correlations fall within a $\pm$ 0.25 range. The most correlated features are:

- A strong positive correlation between `loudness` and `energy`
- A strong negative correlation between `loudness`/`energy` and `acousticness`/`instrumentalness`

We keep these correlations in mind moving forward.

4.2 Distribution by Genre

Let us examine how popular genres differ with respect to Spotify's audio features. The chosen playlists of each genre are listed above in §3. To compare the songs in each genre we produce an audio profile graph as described in §2.1.1.



In the above audio profiles, we notice some interesting similarities and differences in "shape".

- In the first three plots, we see that Country nearly identical to Rock except in the measure of `acousticness`. Moreover, Rock is quite similar to Metal except in `energy` and `valence`.
- Among the fourth and fifth plot, we see Hip Hop and R&B are nearly identical except in that Hip Hop has a little more `danceability` and R&B has a little more `acousticness`.
- Among the last two plots, we see that Jazz and Classical are quite similar except in that Jazz has much more `energy` whereas Classical has drastically more `acousticness`.

Notice that these differences reflect our intuition - this gives us confidence that our features match their description.

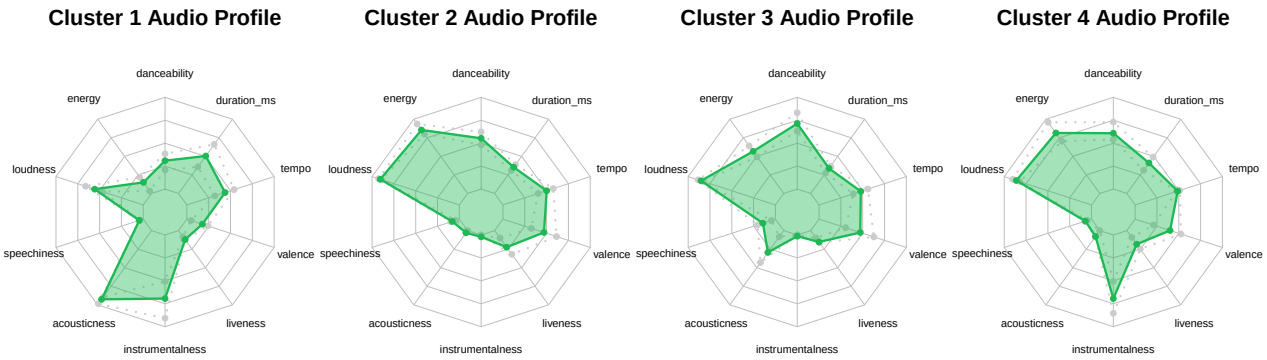
4.3 Clustering with Audio Features

The plots above depict that genres form distinct "shapes" with respect to the audio profiles, where similar genres have similar shapes. Of course, the way we describe genres are not with respect to tempo, energy, valence, etc. – but an interesting question is whether our intuitive notion of genre categorizes music in the same way a machine could.

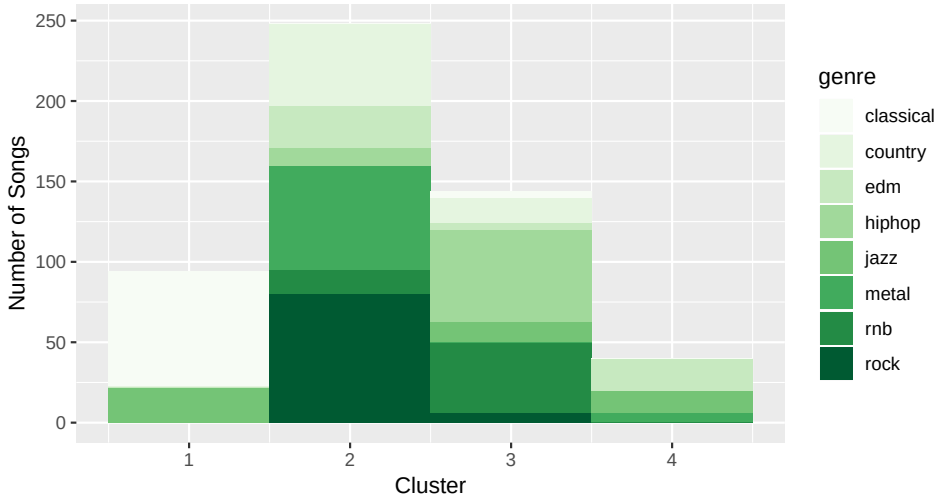
To investigate, let us perform a k-means clustering of the dataset and observe the following:

1. What does a categorization with maximal separation of audio features look like?
2. Does this categorization pick up on our intuitive notion of genre?

In our genre audio profiles, we observe four distinct shapes - Rock/Country/Metal, Hip Hop/R&B, EDM, and Jazz/Classical. For this reason, we will form four clusters and see if the shapes roughly match the distinct shapes defined by these genres. Let us plot the audio profiles of the four clusters below:



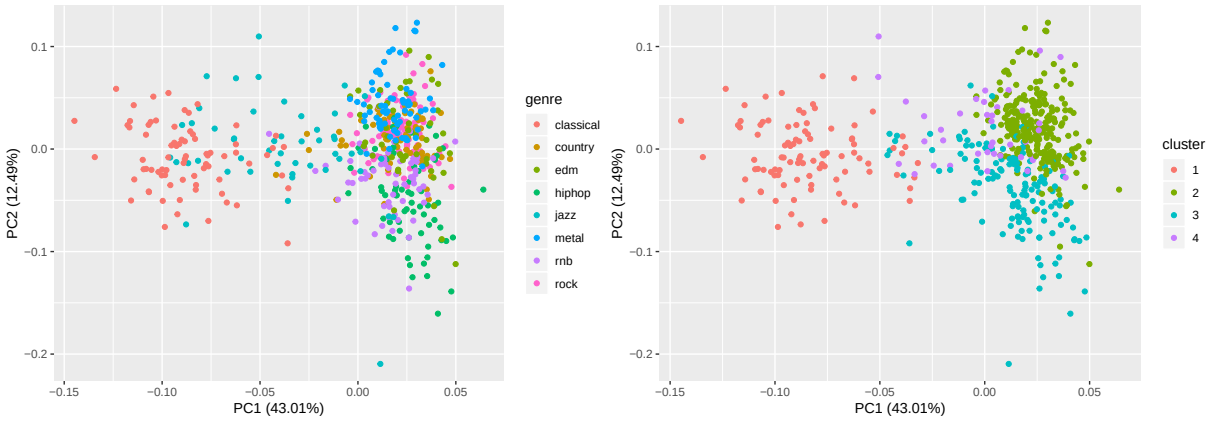
We immediately see some similarities to the genre audio profiles: one of the clusters *closely* resembles the jazz/classical profile and another one *closely* resembles the hip hop/R&B profile. Among the remaining two clusters, one seems to resemble a few genres whereas the other seems to have an entirely new shape. Let us plot the distribution of genres in each cluster:



We see that one of the clusters does indeed pick up on almost all classical music, however jazz seems to be dispersed. One cluster seems to amalgamate rock, country, metal and EDM – this suggest that the audio features are least adept at discerning these genres from one another. One cluster absorbs almost all hip hop and R&B songs as expected. The remaining cluster is certainly the most anomalous cluster, with songs in unrelated genres.

4.4 Principle Component Analysis

Let us try to visualize how genres and clusters appear spatially. To do so, we perform a principle component analysis and plot the data with respect to the first two principle components:



In the first plot, we see that classical and jazz music have a drastically different spread than other genres. However, aside from those two it is difficult to discern a pattern.

The second plot give a little more clarity - on one side we see the region occupied by classical music cluster. On ther other however, we see a split between the upper and lower halves of the vertical spread. In the upper half we have the cluster which captures Hip-Hop/R&B, and in the lower half we have the cluster which captures Rock/Metal/Country as well as EDM.

5. Feature Engineering with Audio Analysis

Before we continue towards our classification task, let us try to obtain more usable features from the audio analysis (described in $\S 2.2$.

5.1 Creating Features

The audio analysis is a series of dataframes with variable numbers of rows for any given song. As such we can't use columns of the data frame as features in and of themselves, however we can compute statistical measures using the columns that are viable as features.

In doing this we need to be careful of two things:

1. The computed features describe a meaningful aspect of the song
2. The computed features do not coincide with an already existing feature in the audio analysis

The available data frames are: `bars`, `beats`, `tatums`, `sections`, `segments`.

5.1.1 Bars

In the bars dataframe we obtain no new information. The number of bars and the mean bar duration is captured already as a function of tempo, duration and time signature – we omit these features to avoid multicollinearity. Moreover, the duration of the bar tends to remain constant throughout a song so in almost all cases the variance in bar duration would only capture noise.

5.1.2 Beats

Again to avoid multicollinearity we do not include any information about beats since the number of beats as well as their mean duration is also a function of tempo, duration and time signature.

5.1.3 Tatums

The tatum is the smallest meaningful division of the beat - songs with smaller divisions of the beat will have a different feel than songs with a larger subdivision. Since this ratio cannot be calculated from existing information the **mean tatum duration** may be meaningful. (We scale this by the mean bar duration to give a ratio that has a better musical interpretation).

5.1.4 Sections

The partitioning of the sections does not have obvious relations to the features we've encountered thus far – therefore we take the **number of sections**, **mean section length**, and **variation of section length**. The mean section loudness would almost exactly match the loudness audio feature so we choose to omit it, but the **variance of section loudness** is a usable feature. The remaining features tend not to change and align with the audio analysis so we will omit them.

5.1.5 Segments

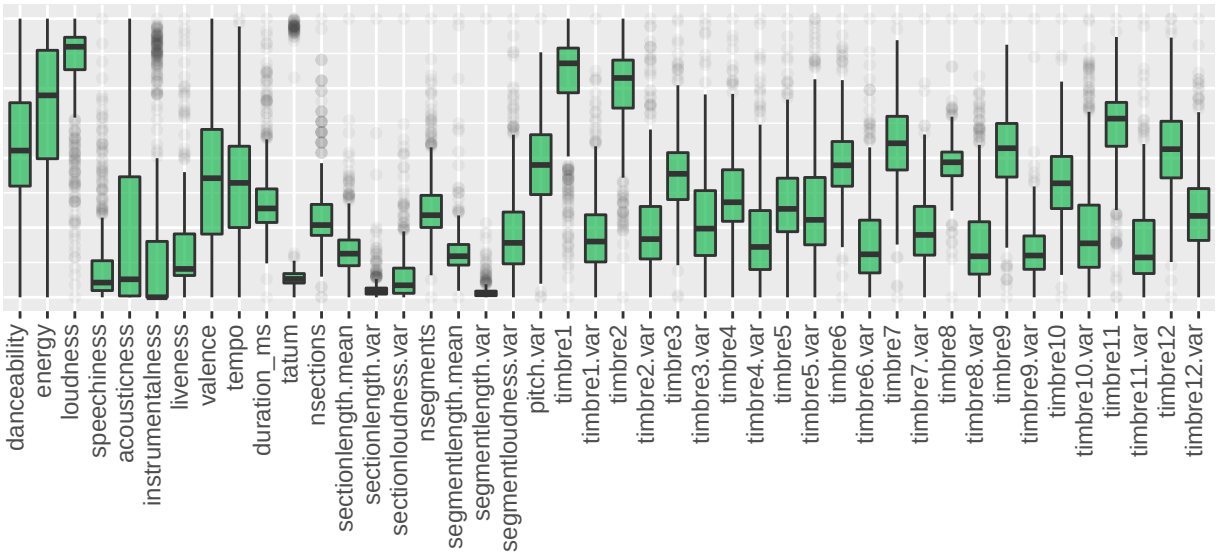
Similar to the above, the partitioning of the segments does not have obvious relations to the features we've encountered thus far – therefore we take the **number of segments**, **mean segment length**, and **variation of segment length**. Again, the mean segment loudness would almost exactly match the loudness audio feature so we choose to omit it, but the **variance of segment loudness** is a usable feature.

The most interesting features we extract from the audio analysis are the timbres and pitches of segments. The timbre is split into 12 components, each discerning a type of sound (limited information is given on the website referenced in §2.2). For each of these 12 components, we take the **mean timbre** and **timbre variance**.

The pitches must be handled a little more delicately - the 12 pitches (corresponding to 12 notes C, C#, D, etc.) and their relative frequency relate to the key of the song. Early on we reasoned that this does not provide meaningful information. We therefore omit the means, but the variance of the pitches is still of interest. It is reasonable to think that a higher variance in pitch would describe a more dynamic melody. However the particular variances in each of the 12 pitches would again relate to the key of the song - therefore, the correct feature to extract would be the **maximum variance in pitch**.

5.2 Initial Feature Selection

We've added several features that seem promising, however we're still unsure of how they are distributed and how they relate to pre-existing features. Ultimately we are trying to decide which features to keep. To investigate the spread of our new features we produce a boxplot – here we scale our data to the same range to visualize their spread relative to one another.



From the boxplot, we see the tatum variable seems to have a discretized distribution. The majority of observations are concentrated at the median 0.125 and the remaining observations are contentrated around 0.16. These values are suggestive since they correspond to eigth notes in 4/4 and 3/4 time signatures respectively.

	Mean Tatum
3/4	0.1664936
4/4	0.1257148
Irregular	0.1247578

Indeed, the 0.16 values correpond to songs with 3/4 time signatures. Since time signature is already included as a feature, we remove **tatums**.

We are moreover interested in minimizing the correlations between features - especially correlations between our new and old features. As a result of correlation with `timbre1` and `timbre1.var`, we decide to remove **loudness**, **sectionloudness.var** and **segmentloudness.var**. A full correlation plot and further details are attached in the appendices below.

6. Song Recommendation

We can now proceed to perform a simple classification task:

Given a Spotify user U, a set of songs that U likes and a set of songs that U dislikes, predict whether the user will like a given new song.

6.1 Sample Dataset

Ideally we would have a large sample of users, each with large sets of liked and disliked songs. However, we are limited in how much data we can acquire and it would also be extremely computationally expensive to compute the features for such a large number of observations.

As such we are forced to work with a relatively small sample. The playlists we will consider have been constructed manually to represent a possible user taste – we will split these songs into a training and test set that represents a new sample of songs.

The two playlists can be viewed in there entirity here: **Liked** / **Disliked**. Given a small dataset and a small scale of analysis, we don't introduce very much nuance in terms of what is liked versus disliked. In short, the playlist compositions are:

Liked (~100): Chance the Rapper (10), Drake (10), Amy Winehouse (10), Frank Sinatra (10), Assorted Jazz (10), Assorted Hip Hop (10), James Bay (5), Marvin Gaye (5), Jazz Lo-Fi (10), Calvin Harris (10), Stevie Wonder (10)

Disliked (~100): Assorted Death Metal (10), Assorted Country (10), Assorted Worship Music (10), Assorted Pop (10), Assorted Kids Music (10), Avant-Garde Classical (~10), Emo Punk (10), J-Pop (10), Sleep Music (10), Reggae (10)

6.2 Model Fitting

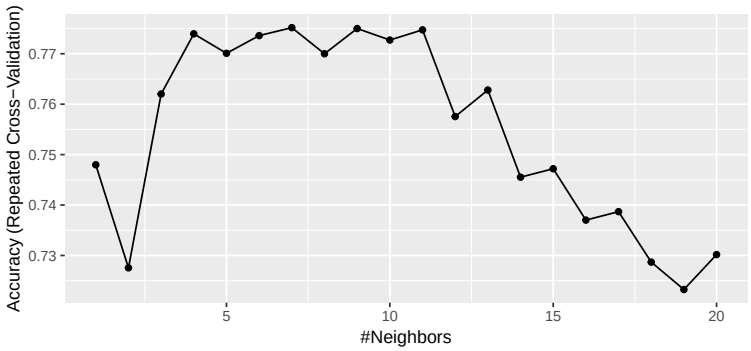
For the purposes of classification, we fit the following selected models: **Multinomial Logistic Regression**, **k-Nearest Neighbours**, **C5.0 Classification Tree**, and **Random Forest**. In general, these models have been selected by virtue of having weak assumptions about the distribution of the data and being effective at handling highly non-linear data. As for the validation scheme, we use `caret` to perform a **10-fold repeated cross-validation** with 3 repetitions for each of these models.

6.2.1 Multi-Logistic Regression

The logistic model was chosen for its simplicity, but primarily to serve as a baseline for the other models. In fact, we *expect* a somewhat poor performance from the logistic classifier – despite carefully selecting features, there may still be multicollinearity between the pre-defined audio features and timbre features from the audio analysis (both of these sets of features are produced with proprietary methods).

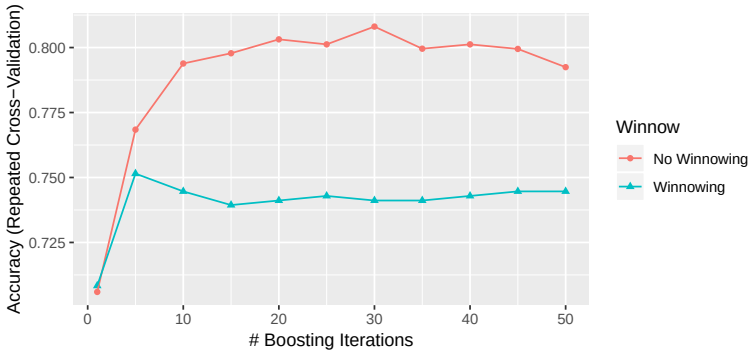
We ultimately obtain a cross-validated accuracy of 0.7520468. We let this serve as a baseline for the remaining models which are better at handling interactions between variables.

6.2.1 K-Nearest Neighbours



The kNN algorithm performs only marginally better on the dataset, with a cross-validated accuracy of 0.7751754 and an optimal 7 number of neighbours (depicted above). As we observed in the principle components analysis, there isn't a large degree of spatial separation even between songs that are quite different. Therefore the nearest neighbour approach also falls somewhat short.

6.2.3 Classification Tree



We again see improvements using Classification Trees – after fitting, we obtain a cross-validated accuracy of 0.8080702. In the above plot we see that, despite having a relative abundance of features, winnowing does not improve the predictive power of the decision tree. This give some affirmation that the majority of selected variables are relevant.

6.2.4 Random Forest

The Random Forest model gives a cross-validated accuracy of 0.8113158 – similar to that of the C5.0 model. In the appendices, we attach the variable importance plot obtained from fitting the Random Forest with the optimal `mtry` value of 4. We see in this plot that many of the variables extracted from the audio analysis turn out to be the most important, indicating their relevance to this classification problem.

6.3 Results

Below is a summary of the cross-validated accuracies obtained from each model as well as their standard deviations:

	Logistic	kNN	C50	RForest
Accuracy	0.7520468	0.7751754	0.8080702	0.8113158
St. Dev	0.0962910	0.1101144	0.1007232	0.1052748

We find overall that the best models (among those we've fitted) are C5.0 Classification Tree and Random Forest tree-based models. Both models have similar predictive power both in terms of accuracy and consistency.

7. Conclusion

In this report we've compiled a set of audio-based features extracted from Spotify's Web API, conducted a thorough exploratory analysis of the features, and used them to perform a simple classification task that mirrors music suggestion.

In our exploratory analysis, we were able to visualize the distribution of songs across many genres with respect to the features we'd compiled. Using the "audio profile", we compared the numerical descriptions obtained in several popular genres of music. We noticed that Rock, Metal and Country were very similar to each other but more surprisingly also similar to EDM. Even given an unsupervised clustering of the data, these genres despite being quite different are difficult to distinguish.

In the principle component analysis, we saw that while classical was easy to distinguish from other genres, in general the numerical distinction between genres is quite fuzzy. This may be indicative of the fact the genres frequently take influence from one another and share musical features.

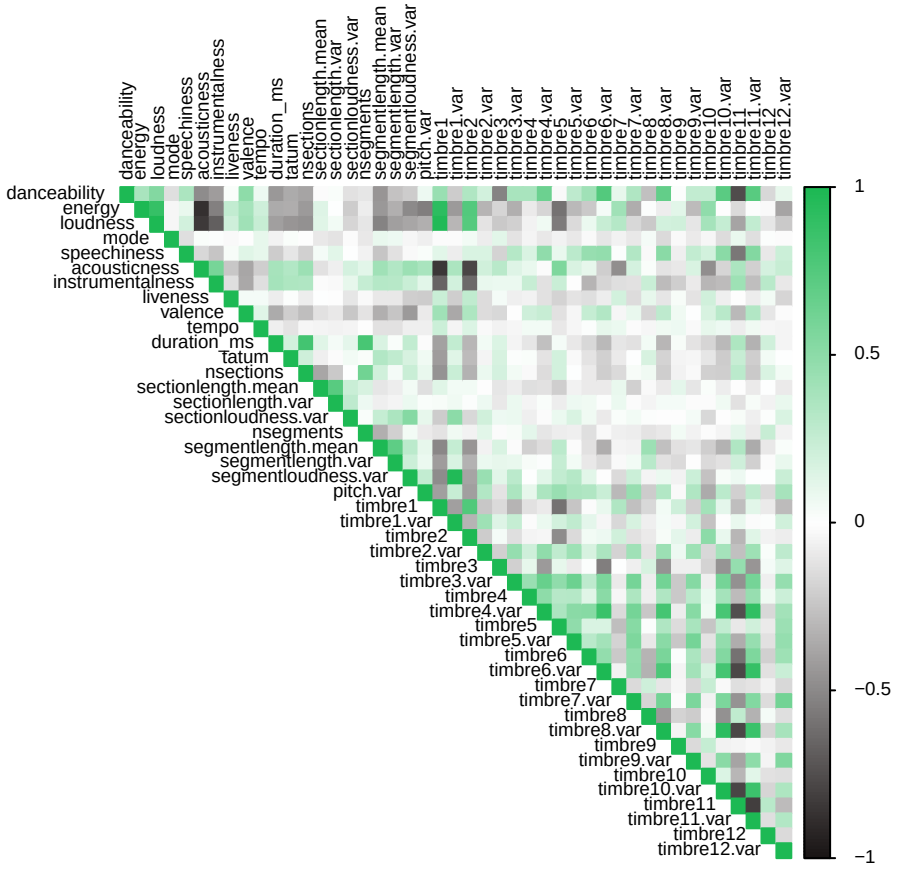
Despite this, the predictive power of the audio-based features is relatively good. We found that tree-based models achieved a rough 80% accuracy, validated with 10-fold repeated cross validation. It should be noted however that we worked in a very simplified setting. We were given a balanced and labelled dataset of songs without much nuance in the distinction between liked and disliked songs. In practice, a set of "disliked" songs is difficult to acquire but this simple setting is suitable for our purposes.

Therefore while we have not demonstrated that music suggestions on the basis of audio data alone is feasible, we can safely conclude that they would very likely be a valuable addition to a collaborative filtering model.

To expand on this analysis we may use more nuanced data and begin to work without a set of "Disliked" music. An interesting addition might be to acquire song lyrics for the given songs and incorporate features on the basis of sentiment analysis.

Appendices

A1. Complete Correlation Plot for Extracted Features



- We see that our original loudness and timbre 1 correlate near perfectly. Moreover, the variance of section and segment loudness correlate very strongly with timbre 1 variance. This suggests that timbre 1 detects loudness and other loudness measures are redundant – indeed, a similar note is made in Spotify's description. We therefore remove **loudness**, **sectionloudness.var** and **segmentloudness.var**.
- As expected the duration, the number of sections and the number of segments have a rather strong positive correlation. It is known however that they are not expressed as a function of each other or other variables so for the time being we do not omit them.
- There seem to be some haphazard correlations between timbres and their variances. But again there is no obvious reason why this might be the case so we refrain from omitting timbre variables for the time being.

A2. Random Forest Variable Importance

The following is the variable importance obtained in the final cross-validated Random Forest model in order of importance: