Recommendation based on Spotify songs database

Sam King

15th Jan 2021

Chapter 1

Introduction and overview

We have a dataset that is comprised of over $170,000$ songs gained from Spotify. Every song has metadata features as well as metrics calculated around the song. The goal of this report is to be able to recommend songs that have similar metrics to the songs that a user may already have in their playlist. To do this I will try various clustering techniques, using both clustering algorithms as well as dimensionality reduction methods. The aim is to be able to group the data into several classes in order to be able to effectively group the users songs. Some side questions that may be asked include investigating how indicative a song's features are of its popularity or the year/decade it was released. The process that I will be using to carry out this task is Microsoft's Team Data Science Process [3].

Chapter 2

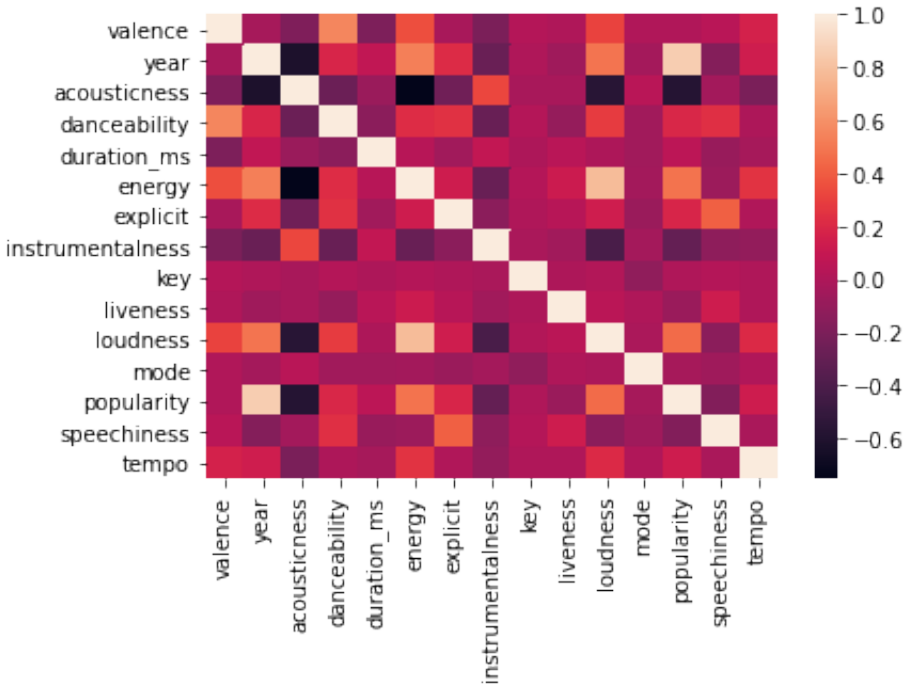
Business Understanding

A benefit from a business point of view of recommending songs to the user is that they continue using the service provided. It is hoped that most songs recommended to them will be enjoyed as they have similar features to the songs that they may already like. Of course the reason someone may like a song might have nothing to do with the features of a song and more to an emotional connection, but for the most part a person's enjoyment of music stems from their enjoyment of certain features around it.

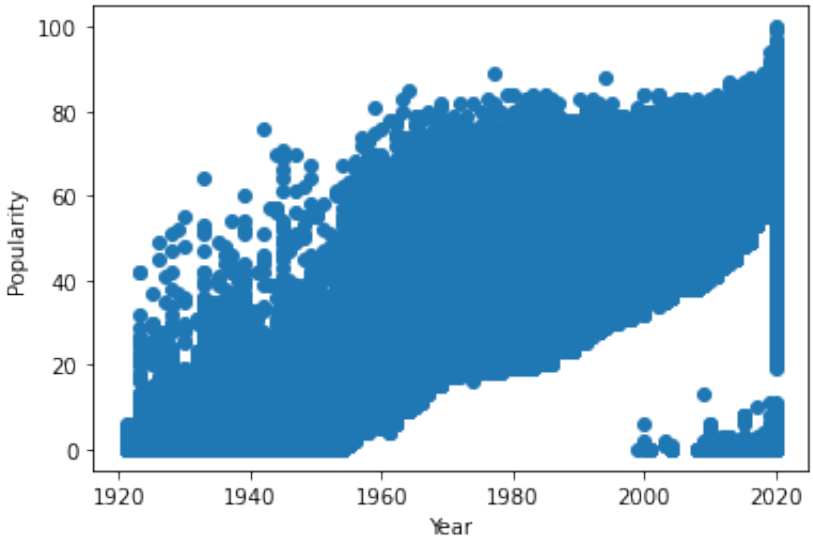
Chapter 3

Data Acquisition & Understanding

This data was acquired from Kaggle [1] and had mostly already been cleaned. All the data points were the datatype I wanted them to be in, and there are no missing values. There are 19 different features for each song but we will not use all of them. Due to the number of artists in this dataset I will not use the artist names as a feature, although I would imagine that two songs from the same artist could be very similar. The same goes for the song title, I will not be investigating how similar two titles are with each other as this has no bearing on whether the content of the two songs is actually similar or not. The ID of the song is irrelevant to us in this phase as it is an arbitrary value assigned to each song so has no bearing on the features of the song whatsoever however could prove useful later on in the deployment stage.The year and the release date are mostly more or less the same so we can drop release data as a feature. Because of this we now drop artists, id, name, and release date from our dataframe. The year and the popularity could be useful, but like the other dropped features they are also metadata so aren't indicative of the content of the song. We will still keep them for now though in order to investigate whether our metrics actually have any bearings on popularity or if certain metrics increase as the year increases. To begin we will do some exploratory data analysis in order to get a sense of the data that we are working with.



Above is the heatmap of the data points. There are some key things to take away from this as it shows how correlated the features are with each other. Key and liveness have little to no correlation with anything else. This is to be expected, since this has no reflection on the other features of the song. Two contrasting songs could have a similar key, and vice versa. We see minor levels of correlation between various features, such as acousticness with several different features. As acousticness goes down one could expect the song to have certain features and this is especially true with energy and loudness which are more associated with electronic types of music. We can see that popularity is largely unaffected by most variables which answers our question about how much the metrics of a song can impact the popularity of it. The year the song is released is strongly correlated with its popularity, although as this is on Spotify it is reasonable to expect that the more recent a song is the more popular it will be with the typical audience base of Spotfiy. This is shown in the graph below:

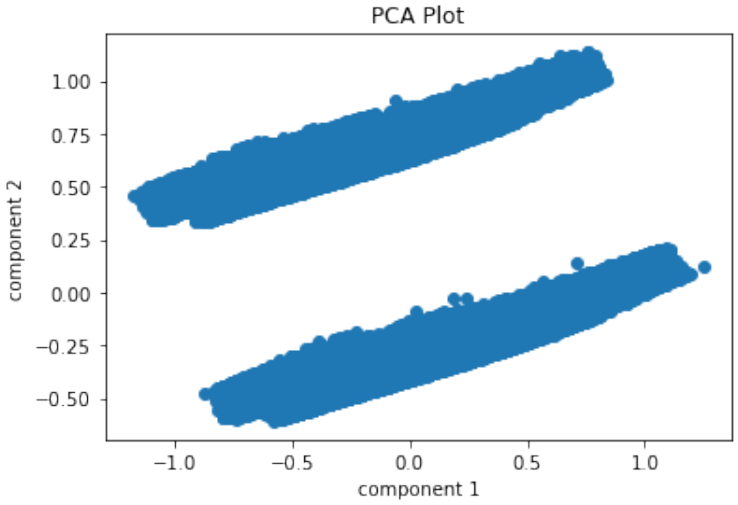


We can see from this the correlation between popularity. Of course there are outliers as every modern song will not be that popular, but generally more people will listen to the more modern songs.

Chapter 4

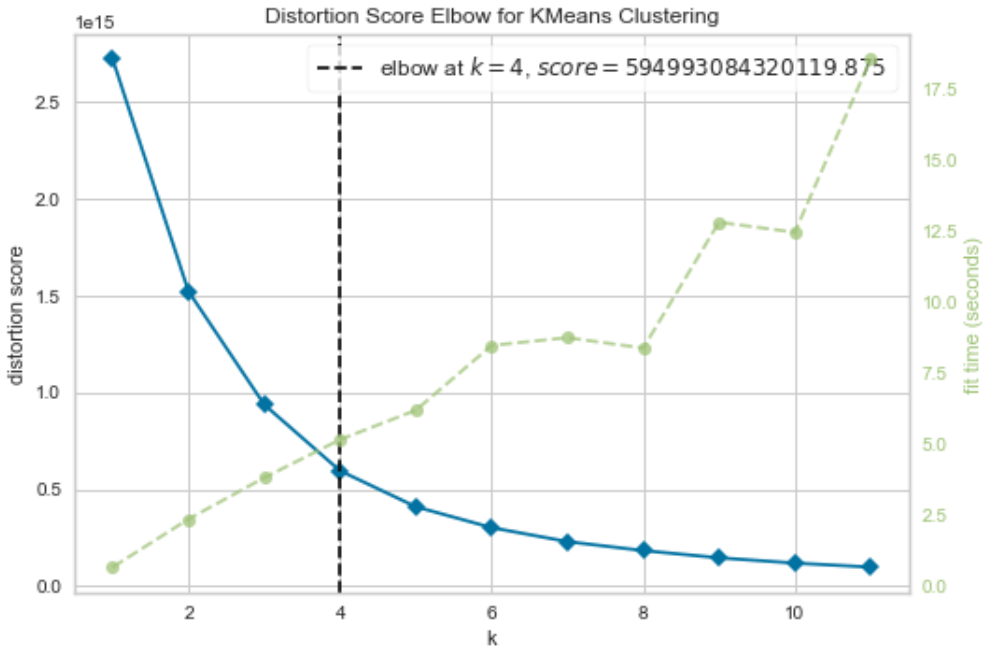
Dimensionality Reduction & Modelling

Now that we have cleaned and scaled our data we can now start looking for any clusters within the data. I want to find as many clusters as I can to ensure that the recommended songs are as similar to the ones they already have as possible whilst still maintaining that the clusters have to be clear enough. We will try different methods of dimensionality reduction before applying different clustering algorithms to determine how many have been found. We start by performing a principal component analysis on our initial data. Below is what the data is projected to look like in 2 dimensions.

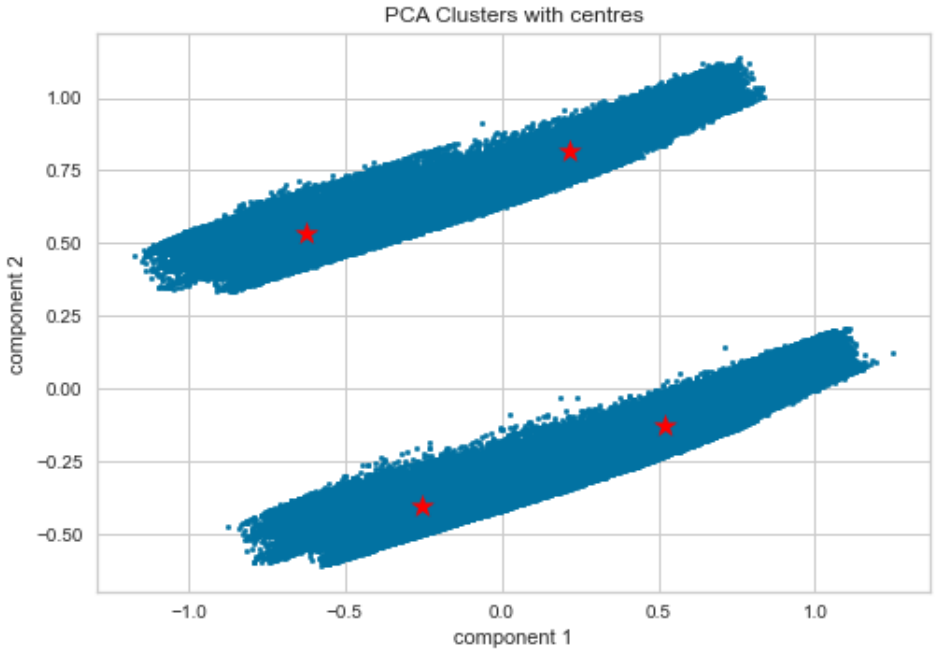


From this graph we can see that there are 2 clear clusters and each

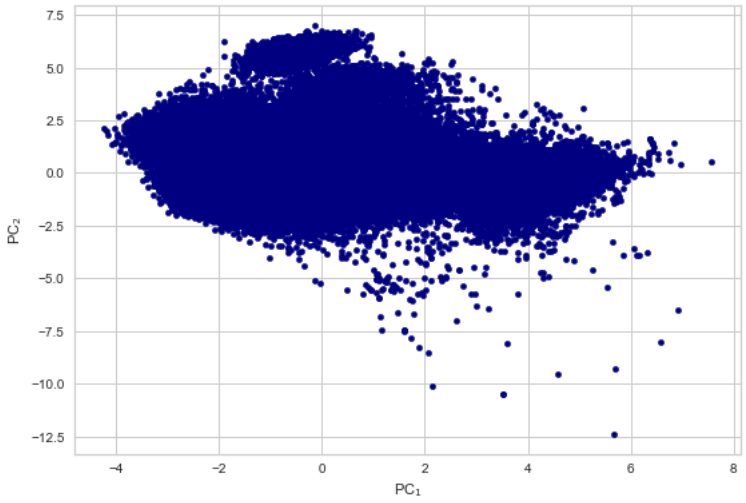
principal component seems to have a linear relationship in each cluster. So already we have 2 distinct classes of songs that we can use. We will use the elbow method for k-means clustering on our principal components to find the optimal number of clusters in our graph.



The elbow method has given us a k value of 4 meaning that each cluster we saw in our projection will probably have 2 in there. Using the projection graph of the data and using k-means clustering we can plot the centres of the clusters on the graph as shown below.

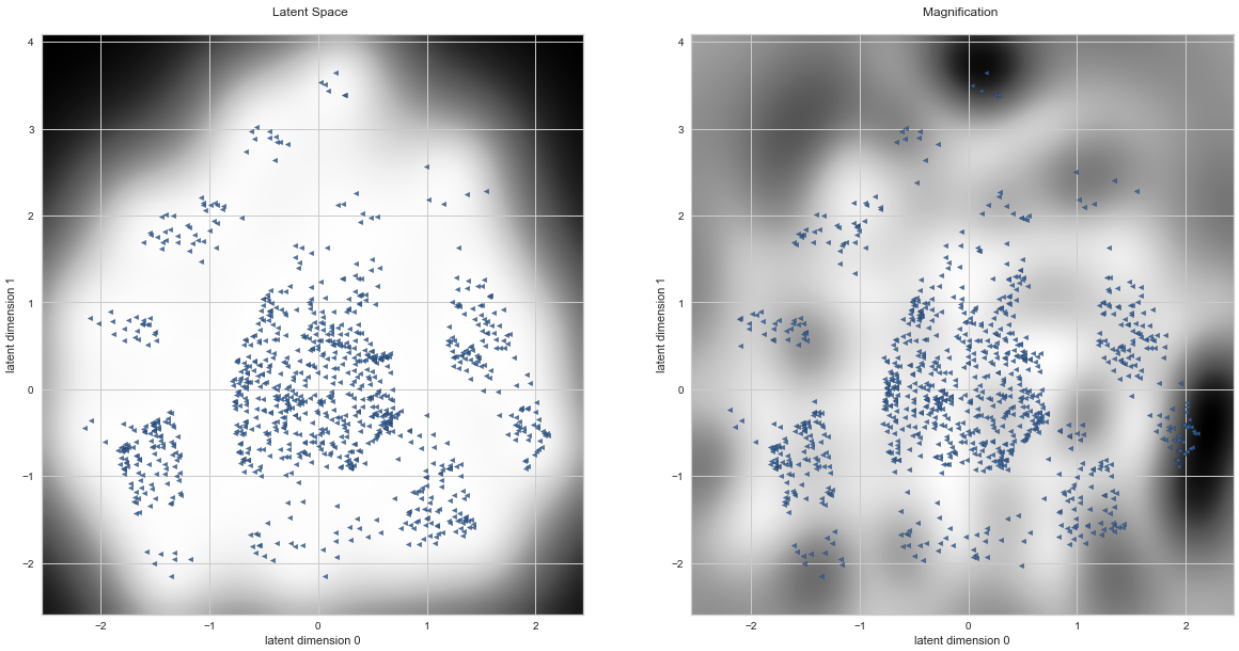


Before we proceed with investigating if these 4 clusters can be separated any further, we will look at other dimensionality reduction techniques to see if we can find more clusters. We start to see if the Gaussian representation of PCA is any different to the one we calculated.



As we can see from the graph above there aren't as many clear clusters

as in the PCA we calculated before. There are however a lot more outliers in this projection than the other PCA in which the points were fairly uniform in two linear clusters. I then decided to try a GPLVM to see if I could get a better projection of the data with more clear clusters that I could use to group the songs in. In order for the algorithm to run more smoothly and quickly I elected to sample 1% of the data. I had attempted to use the IVM method stated in [2] in order to find the optimal number to use however my attempts to use the method effectively were not successful.



This method has several more clear clusters than my previous attempts of dimensionality reduction so I decided to use the number of clusters I could see in this graph. I counted 20 fairly clear clusters from this graph and decided that it was an appropriate amount of classes to be able to recommend songs. Because of this I used to k-means clustering algorithm with $k = 20$ in order to find the centres of the clusters and their associated labels. Using this I now had 20 clusters ready to be used for my deployment stage.

Chapter 5

Deployment

Having found the number of clusters from the dimensionality reduction, we can try to deploy our method. I have taken 2 songs out of the set randomly to use as a test for deployment. I will find which cluster centre they are closest to and select 5 random songs from the cluster to compare. Below is the first song that I used.

```
valence                                                    0.207
year                                                        1958
acousticness                                               0.798
artists            ['Wolfgang Amadeus Mozart', 'George Szell', 'C...
danceability                                               0.266
duration_ms                                               317027
energy                                                    0.0692
explicit                                                       0
id                                        2wXu2k0TvfqEFNRUZikStI
instrumentalness                                           0.709
key                                                            0
liveness                                                  0.0949
loudness                                                 -22.05
mode                                                           1
name               Serenade No. 13 in G Major, K. 525, "Eine Klei...
popularity                                                    21
release_date                                                1958
speechiness                                               0.0452
tempo                                                     81.662
Name: 44954, dtype: object
```

And now we have the songs that were recommended for it:

	valence	year	acousticness	artists	danceability	duration_ms	energy	explicit	id	instrumentalness	key	liveness	loudness	mode	name	popularity	release_date	speechiness	tempo
9	0.771	1921	0.982	['Fortugé']	0.684	196560	0.257	0	08zfJvRLp7pjAb94MA9JmF	0.0000	8	0.5040	-16.415	1	Il Etait Syndiqué	0	1921	0.3990	109.378
35	0.791	1921	0.991	['Ermanno Wolf-Ferrari', 'Arturo Toscanini']	0.465	151187	0.284	0	18BvzCRC1jeAsj1U70YuRy	0.0225	9	0.1460	-17.776	1	Overture	0	1921	0.0591	98.336
55	0.959	1921	0.951	['Louis Boucot']	0.823	169267	0.329	0	2EV71z9RFz15SDR8WezEXG	0.0000	5	0.0646	-12.853	1	Une Canne Et Des Gants	0	1921	0.1430	119.061
90	0.716	1921	0.978	['Georgius']	0.502	158173	0.403	0	3ZlvfjWLxFqfOGK2FTR2ph	0.0000	7	0.2550	-16.814	1	Un Agent Courait	0	1921	0.4660	67.883
108	0.918	1921	0.834	['Dennis Day']	0.661	167027	0.489	0	4v8CkyrDdgtgqkssZwaDe8	0.0000	10	0.1860	-11.043	1	St. Patrick's Day Parade	1	1921	0.1030	125.023

There were a few formatting issues the dataframe so the years are not in the right place. The key success from this selection is the selection of another classical music song, namely the Overture. In our database the Mozart song is listed as being from 1958, and whilst we didn't use the year in our clustering it would potentially carry the same song metrics as other songs from that era. We see that as the other chosen songs are all songs from pre-1950. So whilst the song selection is rather inaccurate in some places, it has managed to find songs that have a lot of similar metrics.

Here is the second random song chosen:

```
valence                          0.625
year                              1996
acousticness                  0.000391
artists                     ['Weezer']
danceability                      0.338
duration_ms                      172600
energy                            0.895
explicit                              0
id                 1rvbY7HXYsFXSlLqxphocK
instrumentalness                      0
key                                   1
liveness                          0.304
loudness                         -4.901
mode                                  0
name                            Getchoo
popularity                           42
release_date                 1996-09-24
speechiness                      0.0466
tempo                            104.492
Name: 87367, dtype: object
```

And the selections that were made:

	valence	year	acousticness	artists	danceability	duration_ms	energy	explicit	id	instrumentalness	key	liveness	loudness	mode	name	popularity	release_date	speechiness	tempo
2539	0.731	1934	0.139	['Rock Projection']	0.711	234429	0.680	0	1H99uz4Nja1yfvvQzyK8zl	0.50500	2	0.3230	-8.165	1	Stay Calm	0	1934	0.0290	112.506
3038	0.627	1937	0.231	['Gibson/Miller Band']	0.685	206907	0.537	0	4RGclfERT7Fy1a9WgKyocT	0.00000	4	0.0854	-9.723	1	Mammas Don't Let Your Babies Grow Up to Be Cow...	21	1937	0.0390	113.017
3102	0.666	1937	0.393	['Johnny Cash', 'Waylon Jennings']	0.693	182227	0.538	0	10MTrUZCuQbJw189sLz2nC	0.00001	4	0.0779	-14.922	1	Even Cowgirls Get the Blues	7	1937	0.0513	177.546
5049	0.974	1947	0.194	['Johnny Horton']	0.694	129960	0.728	0	2o0PmPnrtJVSOlNvgMa4Z6	0.00000	4	0.1660	-8.718	1	They Shined Up Rudolph's Nose	13	1947	0.0249	98.185
5089	0.960	1947	0.326	["Flo Sandon's"]	0.768	197107	0.600	0	2CtBrlLPV3azP1Ktv0MV4S	0.00000	3	0.3330	-8.351	1	El Negro Zumbon	15	1947-01-01	0.0963	82.450

This is a worse selection than the one before. A few of the songs are roughly in the same genre but there are a couple that have vaguely similar features to the song that we had selected.

Chapter 6

Conclusion

To conclude my clusters weren't quite as accurate as they could've been. They worked in some places in terms of the random samples I had picked out but with the nature of the dataset they weren't going to be the exact same type of music. There were many songs with many different features so successfully categorising them would have been tough. With standard PCA we found that there were 4 clear clusters, but I felt that these were not enough to successfully suggest songs that were similar to the song they already had in their playlist

.

Code can be found [here] where the jupyter notebook is given.

Bibliography

[1] Yamac Eren Ay. Spotify dataset 1921-2020, 160k+ tracks. https://www.kaggle.com/yamaerenay/spotify-dataset-19212020-160k-tracks?select=data.csv, 2020.

[2] Neil D Lawrence and John C Platt. Learning to learn with the informative vector machine. In *Proceedings of the twenty-first international conference on Machine learning*, page 65, 2004.

[3] Microsoft. What is the team data science process? https://docs.microsoft.com/en-us/azure/machine-learning/team-data-science-process/overview, 2020.