Imperial College Business School

Using Principal Component Analysis and K-Means Clustering for classical music recommendations

Maria Amanta Zachopoulou [redacted]

MSc Business Analytics

September 2021

Word Count: 5,486

Abstract

The incorporation of machine learning techniques within the music industry has significantly transformed listeners' experience by allowing for personalised recommendations. At the forefront of this evolution is Music Information Retrieval, a field that develops music analysis algorithms by extracting characteristics from audio files. The aim of this project was to exploit MIR by exploring an unsupervised learning approach that captures the attributes of western classical music and creates recommendations based on a selected piece. While there has been extensive research on genre detection, there is not enough literature exploring classical music clustering.

Motivated by some inaccuracies in Spotify's auto-generated playlists, an algorithm was created that reproduced this functionality. The model was built using different feature categories, and the dimensionality was reduced using PCA. K-Means clustering was then used to divide musical pieces into playlists. The algorithm yielded satisfactory results and it was concluded that a combination of data driven approaches and music expertise is essential in evaluating such a model. Classical music listeners form a significant part of Spotify's membership base. It is therefore essential that such approaches are considered to improve listeners' experience and avoid churning to streaming platforms that are more heavily focused on classical music.

Table of Contents

Chapter 1: Introduction

1.1 Background

The way people listen to music has changed drastically over the centuries; from the gramophone, vinyl records and audio cassettes to the walkman, CDs and mp3 players. All these inventions share something in common: Whether it is through sound wave etching, audio sampling or mp3 file downloads, music has to be stored within the device. Thanks to music streaming, this is no longer a requirement.

Streaming platforms like Spotify and Apple Music allow users to search and listen to millions of songs instantaneously. Furthermore, the use of machine learning algorithms like collaborative filtering and natural language processing (NLP) has enhanced the listeners' experience by offering personalised song recommendations [1]. This report shall focus on one specific recommendation feature: Spotify's 'radio' functionality.

Spotify's support team described the feature as follows: "Imagine you're not really sure what to listen to specifically right now, but you know what mood you're in. Simply select lets say a song which represents this mood best and start the Radio from there!" [2]. Depending on what they feel, listeners often seek playlists to set a particular atmosphere or match their mood. The ability to select a song and automatically receive a tailored playlist with similar tracks is a revolutionary way of listening to music that sets Spotify apart from other streaming platforms.

Spotify's radio works via content-based filtering [3], where the features of each song are analysed and used to make filtered recommendations [4]. Collaborative filtering is also used to locate music that users who listen to your selected song also like [3]. Despite the high potential of the radio feature, many users have expressed their dissatisfaction due to repetitive or inaccurate recommendations [5]. This is especially the case for classical music [6][7][8]

Western classical music is divided into several sub-genres that are defined by different time periods; the four predominant ones being the Baroque (1600-1750), Classical (1750-1830), Romantic (1830-1900) and 20th Century (1900-2000) eras, each of which is defined by its own musical characteristics and composition styles[1].

[1]A table describing the differences between these musical eras can be found in Appendix C

A key limitation of Spotify's radio is that it sometimes fails to separate eras from each other. An example of this can be found in Appendix A, which presents a screenshot of the 'radio' playlist based on the Prelude from Bach's first cello suite[2]. One of the pieces included in the playlist is Aaron Copland's Rodeo[3]. Firstly, Arron Copland's piece was written more than two centuries after Bach's first Suite, therefore making it substantially different in music characteristics and compositional style. In addition, although the way we perceive music is entirely subjective, it could be argued that the former piece is much calmer than the latter. Hence, the two pieces should appear in different playlists as they represent contrasting moods.

Motivated by this observation, the aim of this research project was to explore an alternative way of recommending music based on a selected classical piece. As opposed to Spotify's algorithm, this approach placed more focus on the audio characteristics of the pieces while also taking into account the era in which they were written. Classical music listeners form an integral part of Spotify's members, with 7.2, 6.1 and 5.7 million people listening to Bach, Beethoven and Mozart respectively per month [37]. It is therefore essential that appropriate methods are acquired to ensure a smooth listening experience for such individuals, as the probability of them churning to more classical focused streaming platforms can be high.

The first and most crucial objective of this investigation was to source an appropriate dataset with mp3 extracts of western classical pieces. A machine learning algorithm was then created, which took as input a classical piece and categorised it to its respective classical era. It then identified pieces with similar characteristics and overall mood using K-Means clustering. The desired outcome of the algorithm was an automatically generated Spotify playlist that contained pieces similar to the one selected by the user.

1.2 Literature Review

Music classification falls under the category of Music Information Retrieval (MIR), a field that explores ways of extracting information from audio files. This notion was first introduced by Krumhansl and Kessler [9], and is now widely used for tasks such as instrument recognition, and genre classification [10]. Since machine learning models are not trained with audio files directly, feature engineering is a crucial step of such processes. Depending on the task, there are several feature categories that can be extracted from an audio sample. The literature that will be examined focuses on three popular feature types: Symbolic features, 'Spotify' features and Spectral features.

[2] Audio: https://www.youtube.com/watch?v=1prweT95Mo0
[3] Audio: https://www.youtube.com/watch?v=VXCfEuU2eu0

Symbolic feature extraction has been widely used for music analysis by utilising packages such as *music21*; an open source toolkit developed by M.I.T which analyses symbolic music data [11]. In their paper, "*Feature Extraction and Machine Learning on Symbolic Music using the music21 Toolkit",* Cuthbert, Ariza and Friedland [11] examine several machine learning models to distinguish between compositions of Monteverdi and Bach. They achieved this by using a dataset of Musical Instrument Digital Interface (MIDI) files and extracting features such as the percentage of notes that contain accidentals and the most common note length. Symbolic features are also used by [12] and [13] for genre classification. Some of the features used are key, dynamics, rhythm and instrumentation.

Although all three papers reported good results, it is important to note that obtaining MIDI files is a very challenging task. According to [12], websites that offer MIDI files do not have a public API for use. Creating a sufficiently large dataset thus requires a substantial amount of time to download enough MIDI files through a scraper. Moreover, there is not enough available symbolic data of good quality [14]. Finally, MIDI files are not appropriate for the purpose of this report as the algorithm is trying to reproduce Spotify's radio where the input format is always audio files.

The aforementioned 'Spotify' features refer to the information that is included within Spotify's metadata for each piece. Examples of such features are the loudness, tempo and energy of a song. [15] as well as [16] are using such features to provide song recommendations based on a pre-existing playlist. Both publications do this by using k-Means clustering, and have recorded satisfactory results. It is important to note however, that [16] observed no strong correlations among the features and no distinct patterns based on a song's genre. In addition, neither publication focused on classical music.

Despite Spotify's features capturing some important high level attributes, they are not enough to distinguish between subcategories of classical music. For example, an orchestral piece and a solo piano piece might have the same tempo and key. In addition, since both pieces are instrumental and contain no human voice, their instrumentalness metric should also be the same. However, the change in texture and timbre from a full orchestra to a solo piano convey a different atmosphere, and can arguably be classified as two different moods. This is a musical aspect that is not captured by Spotify's features and should thus be collected from a different source.

A spectrogram is a way of visualising the spectrum of frequencies of sound as it varies with time [17]. Spectral features are defined as the "distributions of energy over a set of frequencies" [18] and form the foundation for many techniques in MIR. During the research phase of this project, it was observed that, as opposed to general genre classification studies, all classical music

publications make use of spectral features. [19] proved that instrument distinction using neural networks is very effective when using spectral features like zero crossing rate and spectral centroid. More specifically, their model achieved an average accuracy of 85% when distinguishing between flute, violin and piano pieces.

Similar features are also used in Weiss' exploration of classical era and composer classification [14]. As his publication focuses on the analysis of audio recordings, he emphasises that some systems that convert audio recordings to symbolic scores do not yield satisfactory results. This further confirms that Symbolic features would not be suitable for the purpose of this project. Spectral features can be obtained using libROSA[4] [20], a python library that converts audio files to spectral images and then extracts the relevant features.

Overall, there has been extensive research in the field of genre classification. Such analyses are usually supported by supervised learning techniques such as K-nearest neighbours [12], Support Vector Machines [31] and Decision Trees [11]. Support Vector Machines in particular were a lot more effective than traditional Euclidean distance methods [31]. However, their high computational complexity does not make them suitable in the context of Spotify, as the platform stores millions of songs. Moreover, the goal behind Spotify's radio is to cluster pieces of the same mood. As music does not have 'mood' or 'character' labels, unsupervised learning techniques as used by [16] and [20] are more suitable.

The aim of this review was to explore the different types of features used in MIR projects. It is clear from the publications reviewed that the types of features used depend on the context of the analysis and the aim of the research. As the dataset of this project consisted solely of audio files, only Spotify and Spectral features were considered. In general, the majority of music recommendation literature focuses primarily on genre classification. A small subset of those publications concerns western classical music and an even smaller percentage looks into unsupervised learning techniques. As the appreciation for classical music is decreasing through generations, services like Spotify's radio are important to encourage the exploration of new pieces from the western classical repertoire. Some listeners might not be familiar with classical composers or eras, but can definitely distinguish between the mood of two pieces, which is what motivated the clustering approach of this algorithm.

[4] A list of all libraries used can be found in Appendix B

Chapter 2: Methodology

2.1 Data Collection and Feature Extraction

Prior to the data collection process, 49 classical composers were selected and the links of their Spotify playlists were saved. Their birth years were also added [21] as indicators of their respective classical era. Through the use of Python's *Spotipy* and *GSA* (General Spotify Analyser) libraries, an algorithm was created which looped through the different playlists and extracted the track names and Spotify features. Some of the Spotify features - like liveness, which measures how much a track sounds like it was recorded live - were removed, since they do not capture any classical music characteristics.

To obtain the Spectral features, mp3 samples were extracted for each track and converted to Waveform (WAV) audio files using the R's *TuneR* package. This was done because *libROSA*, the library that extracts spectral features, requires the input files to be in WAV format. Since looping through 2760 audio files and extracting features is a computationally expensive process, a loop was written using the *pydub* library that trimmed the files to 10 seconds.

Once the dataset was finalised, *Pandas* data frame commands were used to create a third feature category: Composition Features. Classical compositions are given different names, such as Symphony or Concerto, depending on the ensemble that the piece is for. A piece for a full symphony orchestra has a different overall effect compared to a sonata for flute and piano. Five binary columns were added that contained 1 if the equivalent composition type was included in the track's title, and 0 otherwise.

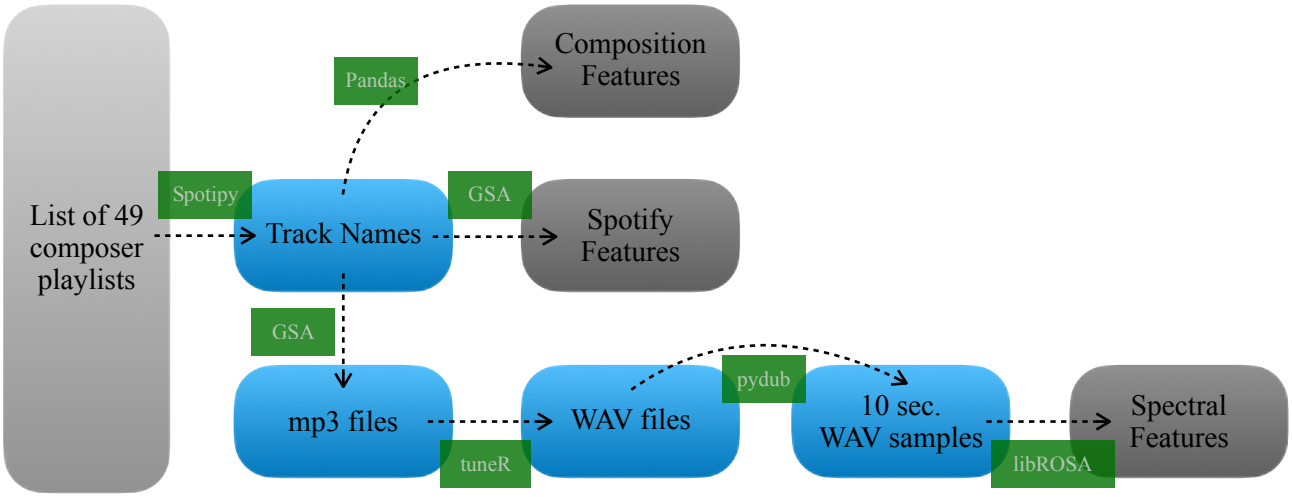


Figure 1: Feature Engineering Process

Spotify Features [33]	
Danceability	The dancing suitability of a track based on tempo and rhythm (scale = 0-1)
Energy	A measure of intensity and activity (scale = 0-1)
Loudness	A measure of how loud a piece is (scale = decibels)
Instrumentalness	A measure of whether a piece contains no vocals (scale = 0-1)
Valence	A measure of musical 'positiveness' or happiness (scale = 0-1)
Tempo	A measure of the overall speed (scale = beats per minute)
Key	The key that the piece is based on (scale = 0-11 where 0=C, 1=C$^{\#}$ and so on)
Mode	The modality that the piece is based on (scale = binary)
Spectral Features [34]	
Chroma STFT	The word chroma refers to the 12 different notes (pitches). The chroma short term Fourier transform therefore represents the chroma (semitone) distribution of a sound.
Spectral Centroid	A weighted mean of a sound's frequencies. The SC describes the 'brightness' of a sound by locating its centre of mass. The louder a sound the higher the SC. The SC for the i-th sound frame and sub-band b can be represented by: $$SC_{i,b} = \frac{\sum_{f=l_b}^{u_b} f \mid S_i[f] \mid^2}{\sum_{f=l_b}^{u_b} \mid S_i[f] \mid^2}$$ where *Si[f]* is the scope of the i-th sound frame and can be divided in two mutually-exclusive sub-bands defined by a lower frequency and upper frequency boundary l_b and u_b.
Spectral Bandwidth	It is the mean distance between increasing frequency to the spectral centroid of the sub-band and thus measures the state of the audio at a particular time.
Spectral Rolloff	A measure of the signal shape by identifying high frequencies.
Zero Crossing Rate	A measure of how often a signal changes polarity along the time axis. Higher values usually indicate highly percussive sounds.
Mel Frequency Cepstral Coefficients	Twenty coefficients (MFCCs) which capture the overall shape of a spectral envelope. MFCCs are predominantly used for speech recognition, but are recently also utilised for MIR thanks to their ability to describe the timbre of a sound. The MFC coefficient of each frequency is given by: $$Mel(f) = 2595 * log_{10}(1 + \frac{f}{700})$$
Composition Features	
Symphony	A piece for a full orchestra (scale = binary)
Concerto	A piece for a solo instrument and an orchestra as accompaniment (scale = binary)
Sonata	A piece for a solo instrument and a piano as accompaniment (scale = binary)
Quartet	A piece for four instruments (scale = binary)
Trio	A piece for three instruments (scale = binary)

Table 1: Feature Descriptions

2.2 Algorithm Overview

The aim of this algorithm was to reproduce Spotify's radio feature so that upon the input of a classical piece it returns a playlist with music that is similar in characteristics and overall mood. Once a piece is selected, the algorithm filters the dataset so that it only includes music by composers who were born 50 years before or after the composer of the piece in question. This step is based on the fact that composers within the same century had similar compositional styles. The dataset dimensionality was then reduced using Principal Component Analysis (PCA), and K-Means clustering was performed to divide the pieces into clusters. Finally, the cluster that contains the selected piece is returned and converted into a Spotify playlist using *spotipy* commands.

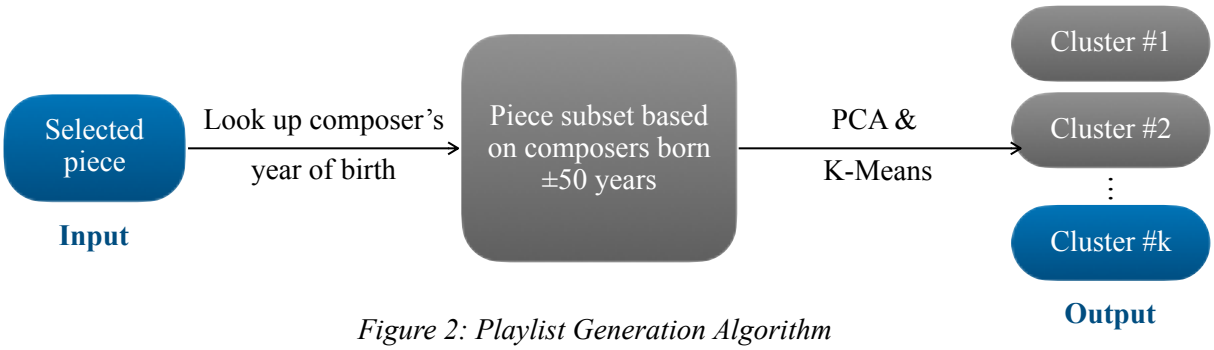


Figure 2: Playlist Generation Algorithm

2.3 Principal Components Analysis

PCA is a dimensionality reduction technique that is often used for information retrieval due to the large number of variables present. It works by replacing the original variables with a smaller number of 'principal components'. These are linear combinations of the existing variables and are often able to retain a lot of the variability explained by the original variables [22]. For this analysis, the first principal component can be written as:

$$Z_1 = \phi_{11}[\text{Danceability}] + \phi_{21}[\text{Energy}] + \phi_{31}[\text{Loudness}] + \ldots + \phi_{39,1}[\text{Sonata}]$$

where the elements φ_{i1} represent the loadings of the first principal component. In our case, Z_1 is a vector containing 2760 linear combinations, since the dataset used has 2760 observations. This is an optimisation problem that maximises the variance of the values in Z_1 while keeping the sum of squared loadings equal to 1. This constraint is placed to prevent the variance from becoming arbitrarily large. Once the first principal component is calculated, the second one can be obtained by finding the maximum variance of all linear combinations uncorrelated with Z_1. The same process is followed for the rest of the principal components [25].

To calculate the principal components, the *pca* tool from the *sklearn.decomposition* library was imported in Python. In addition, all variables were scaled so that they have mean 0 and variance 1. This was done because different features are based on different scales and, if they were not normalised, PCA would be biased towards the variables with the highest variance. Furthermore, as proposed by [32], 'a sample size of 300 is good and over 1000 is excellent', Therefore, the dataset of 2760 observations used was appropriate for the purpose of this analysis.

For the purpose of this project, reducing the number of dimensions was essential. This is because K-Means clustering requires a dataset of 2^m observations, where m is the number of variables [24]. Since there are 38 variables considered, this would require a dataset of over 274 billion observations. By reducing the dataset to 10 principal components, which were found to explain over 60% of the variance, the minimum requirement for a sufficient dataset would be 1024, thus making any subset of the dataset suitable.

2.4 K-Means Clustering

K-Means Clustering is an unsupervised learning algorithm which finds distinct subgroups among the observations of a dataset. The algorithm takes as input unlabelled data and separates the observations in a way that minimises the within-cluster variation. This is a measure of how much the observations in a cluster differ from each other [25]. In order to apply K-Means, the *KMeans* tool was imported from the *sklearn.cluster* library.

It is important to note that there are two popular methods of clustering: K-Means and Hierarchical clustering. While K-Means tries to separate observations among a stated number of clusters, Hierarchical clustering does not need any pre-specified information. This is advantageous, since it is usually not possible to know the exact number of clusters when dealing with an unlabelled dataset. However, Hierarchical clustering computes and stores an n×n distance matrix, where n is the number of observations. This can be particularly expensive and slow for large datasets [26]. Therefore, since this project aims to provide an alternative to Spotify's radio, thus needing to consider millions of observations in practice, the K-Means algorithm was preferred.

The algorithm works by first selecting the number of clusters K. This was done using the elbow method, which plots the within-cluster variance against the number of clusters. The ideal number of clusters is identified by finding the part of the plot that creates an angular shape like an elbow. This is the point where the information gain becomes lower. However, this cannot always be determined without a sense of ambiguity [27]. The *KElbowVisualizer* tool was thus imported from

the *yellowbrick.cluster* library, as it is able to automatically detect the elbow point by using the knee point detection algorithm [28].

Once the optimal K is selected, K-Means randomly assigns observations to one of the K clusters. The centroid of each cluster is then calculated, which is the vector of the feature means of the points in that cluster. The observations are then re-assigned to the cluster whose centroid is the closest. This process is repeated until the clusters are no longer changing [25]. The distance between each point and the centroids is calculated using the Euclidean distance, which is defined as [29]:

$$d(a,b) = \sum_i \sqrt{(a_i - b_i)^2}$$

where *a* and *b* are coordinates. Since it is a distance based algorithm, it is essential that all features are scaled so that there are no different units of measurement. Since this was also the case for PCA, the dataset was already scaled in the previous step.

Finally, the clusters obtained were evaluated using a silhouette analysis, which measures the distance between each point in one cluster to the points in nearby clusters. A silhouette coefficient near +1 indicates that the selected points are far away from the nearby clusters. On the other hand, 0 indicates that the points are on or very close to the cluster boundary and any negative values up to -1 suggest that those points might have been wrongly assigned [30]. The formula used to obtain the silhouette coefficients is

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

where *a* is the intra-cluster distance and *b* is the mean nearest-cluster distance for each sample [27]. Those distances can be calculated using any distance based measure, such as the Euclidean distance.

Chapter 3: Results

Once the data collection and modelling process were completed the results were analysed. The correlations between Spotify and Spectral features are shown in the heat matrix below, while the distribution of each MFCC is presented in Figure 4.

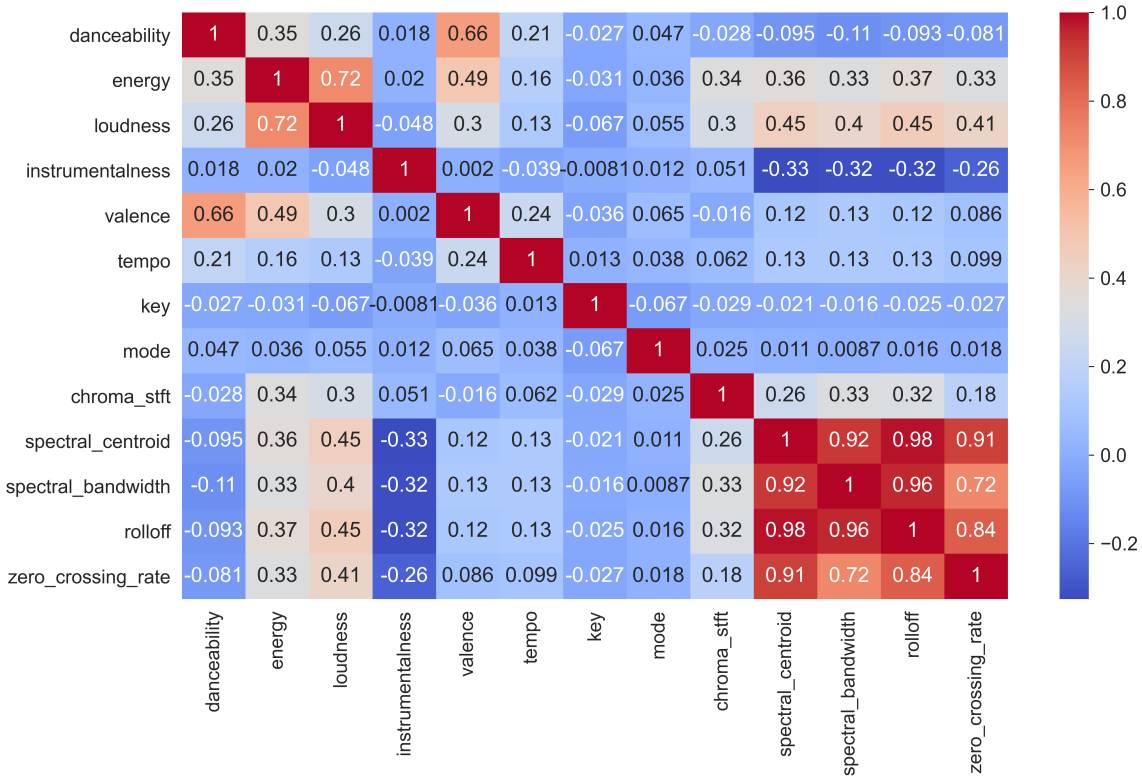


Figure 3: Correlation Heat map of Spotify and Spectral Features

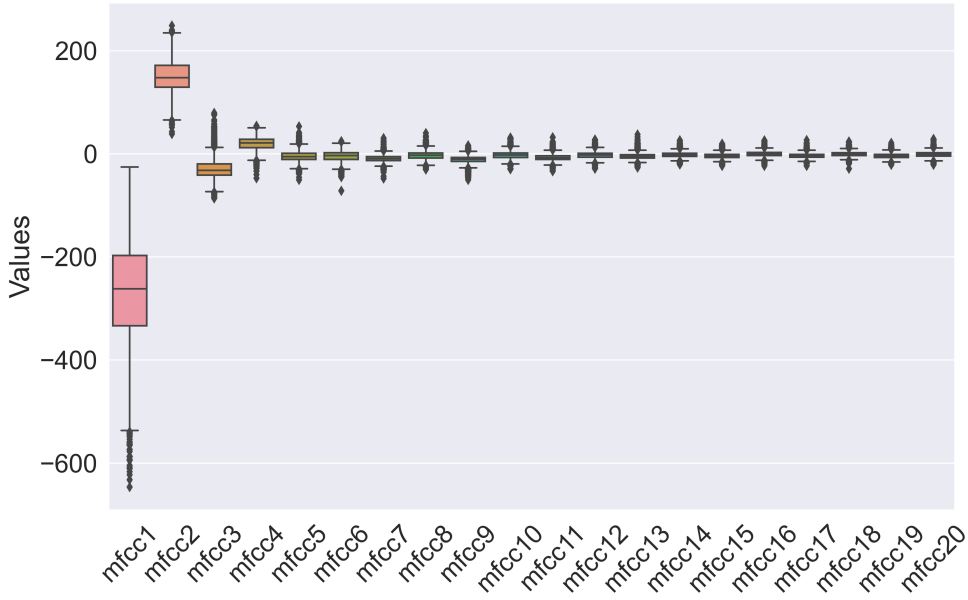


Figure 4: Data Distribution of Mel Frequency Cepstral Coefficients

To understand how different composition types behave across the different spectral measures, the following violin plots were considered. These represent the distribution of values for each composition type. For simplicity, only the first Mel Frequency Cepstral Coefficient is shown.

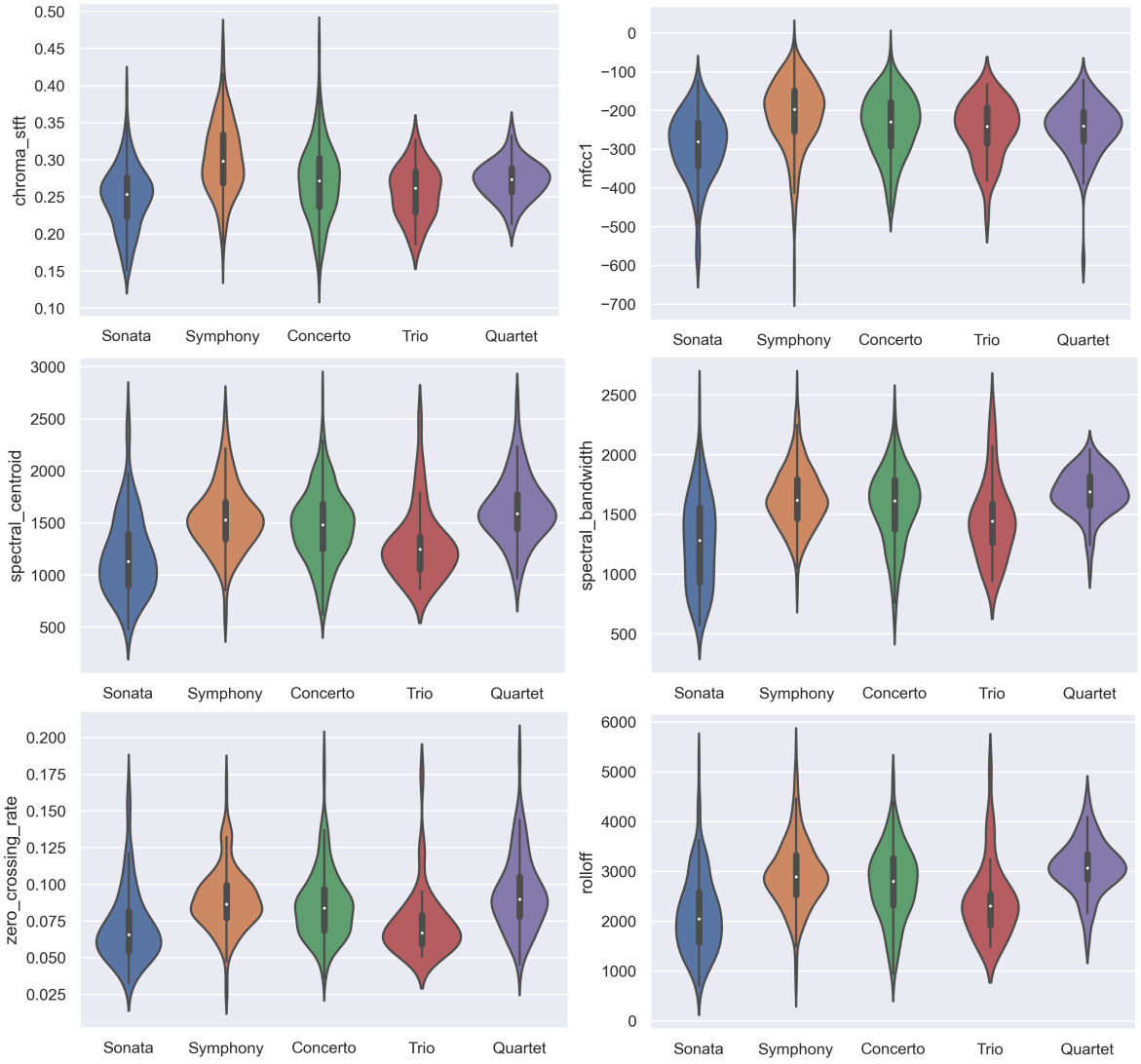


Figure 5: Distribution plots of Spectral Features across Compositional Features

To demonstrate the performance of the algorithm, three pieces were selected at random and the results were recorded. The three pieces are Mozart's '*Symphony No. 40 in G Minor, K. 550: I. Allegro molto*', Elgar's '*Cello Concerto in E Minor, Op. 85: I. Adagio - Moderato*', and Chopin's '*Nocturne No. 2 in E-Flat Major, Op. 9 No. 2*'. The Spotify playlists that were automatically generated by the algorithm can be accessed here[5]. For simplicity, the pieces will be referred to by the name of the equivalent composer.

[5] Mozart: https://open.spotify.com/playlist/1jdpK8LW40SJlga9PhuxSB?si=d7f1d27c289e4ef5
Chopin: https://open.spotify.com/playlist/3Gih2AA013Xh6xXtMOKsUk?si=daf90dbd7f7d4bd5
Elgar: https://open.spotify.com/playlist/4shsj0AOzvDIvmhQjm8PgN?si=27671668f5de4ee4

After selecting the pieces, their respective spectrograms were produced by importing the WAV samples in Python. These are shown in figure 6. A spectrogram is a way of visualising sound that allows us to study its properties. Brighter sounds appear higher up, louder tones have brighter colours and longer tones are shown as longer marks. In addition, the x-axis represents the time while the y-axis is the frequency of a sound.

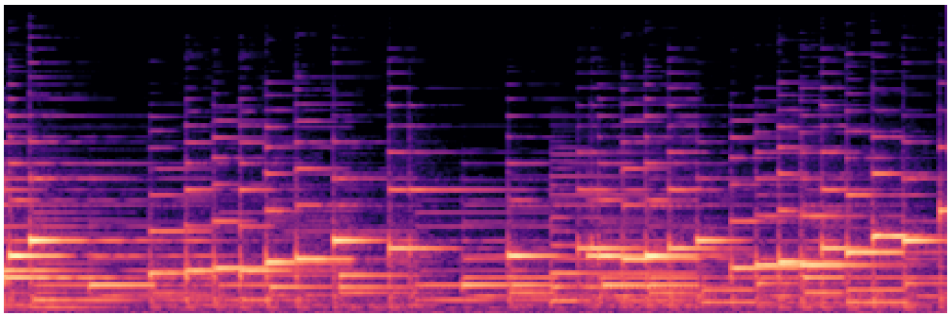


Figure 6a: Spectrogram of Chopin's Nocturne No. 2

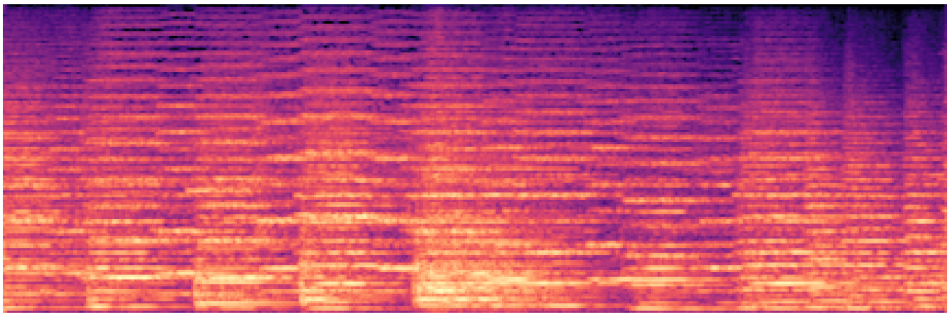


Figure 6b: Spectrogram of Elgar's Cello Concerto

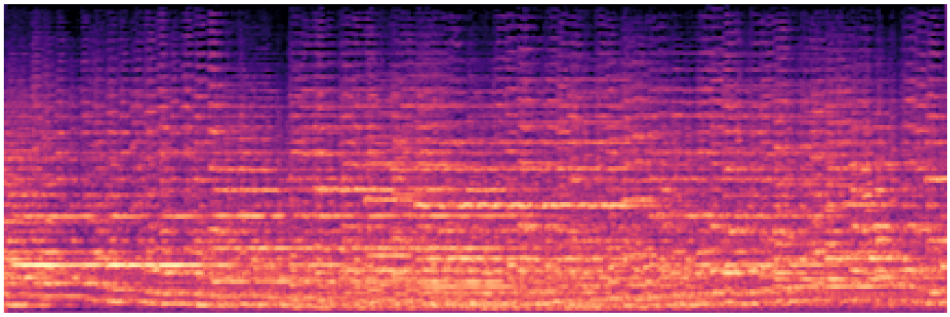
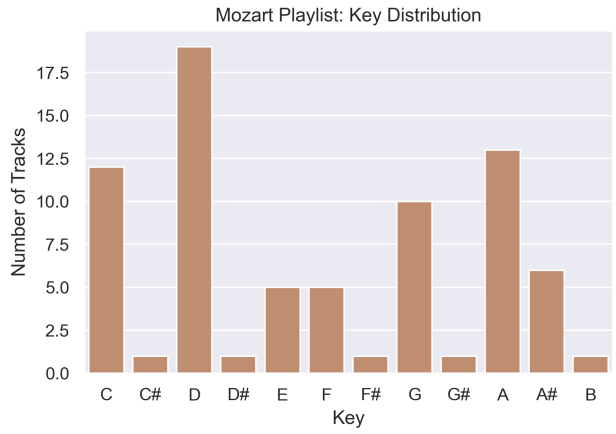
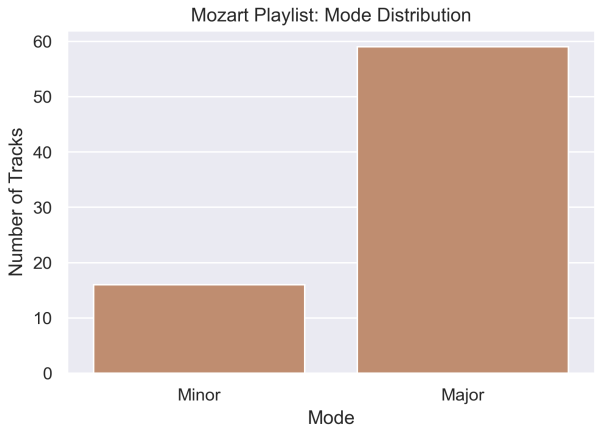


Figure 6c: Spectrogram of Mozart's Symphony No. 40

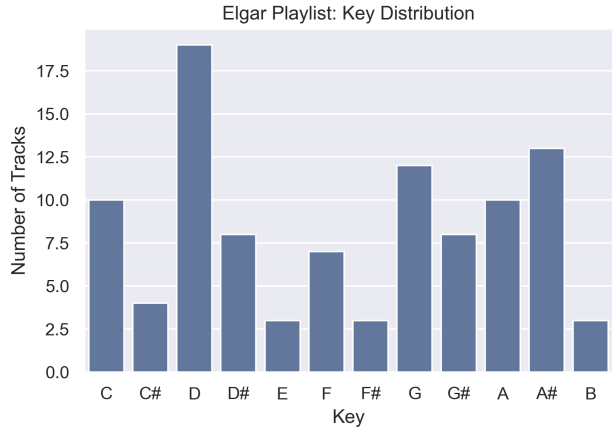
Once the pieces were added as input to the algorithm and their playlists were generated, a series of analyses were performed to assess the clustering performance. Firstly, the key and mode distributions were obtained for each playlist and compared to the original key and mode of the pieces. The average value for each Spotify and Spectral feature was then calculated and compared to that of the original pieces by looking at the percent deviation. For simplicity, only the first 3 MFC coefficients were included. Finally, the silhouette scores were obtained and reported in table 2.

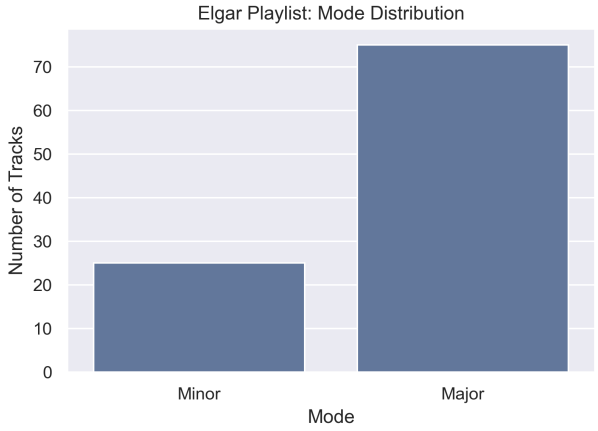




Selected Piece Key: G

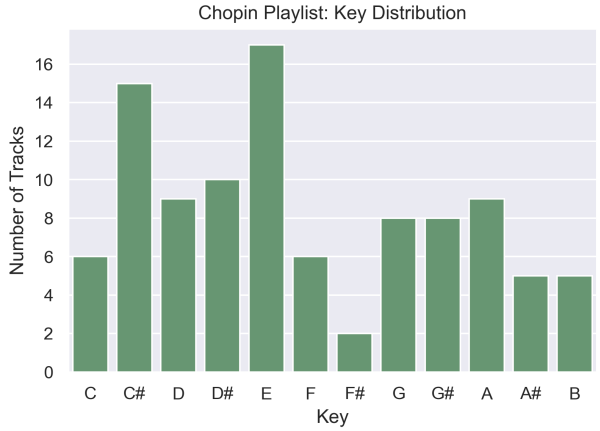
Selected Piece Mode: Minor

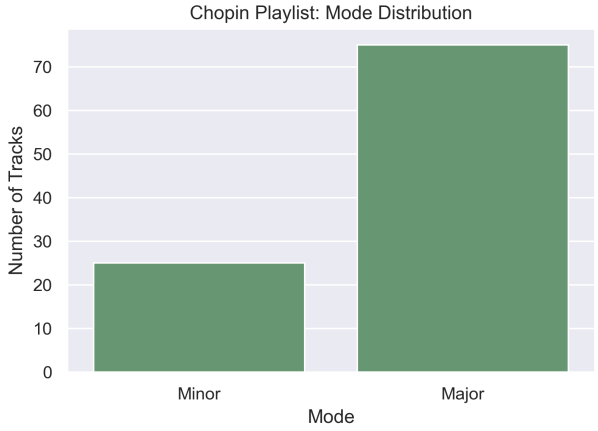




Selected Piece Key: E

Selected Piece Mode: Minor





Selected Piece Key: D# (=E*b*)

Selected Piece Mode: Major

Figure 7: Key and Mode Distributions within auto-generated playlists. The Key and Mode of the selected pieces are included below each visualisation for comparison.

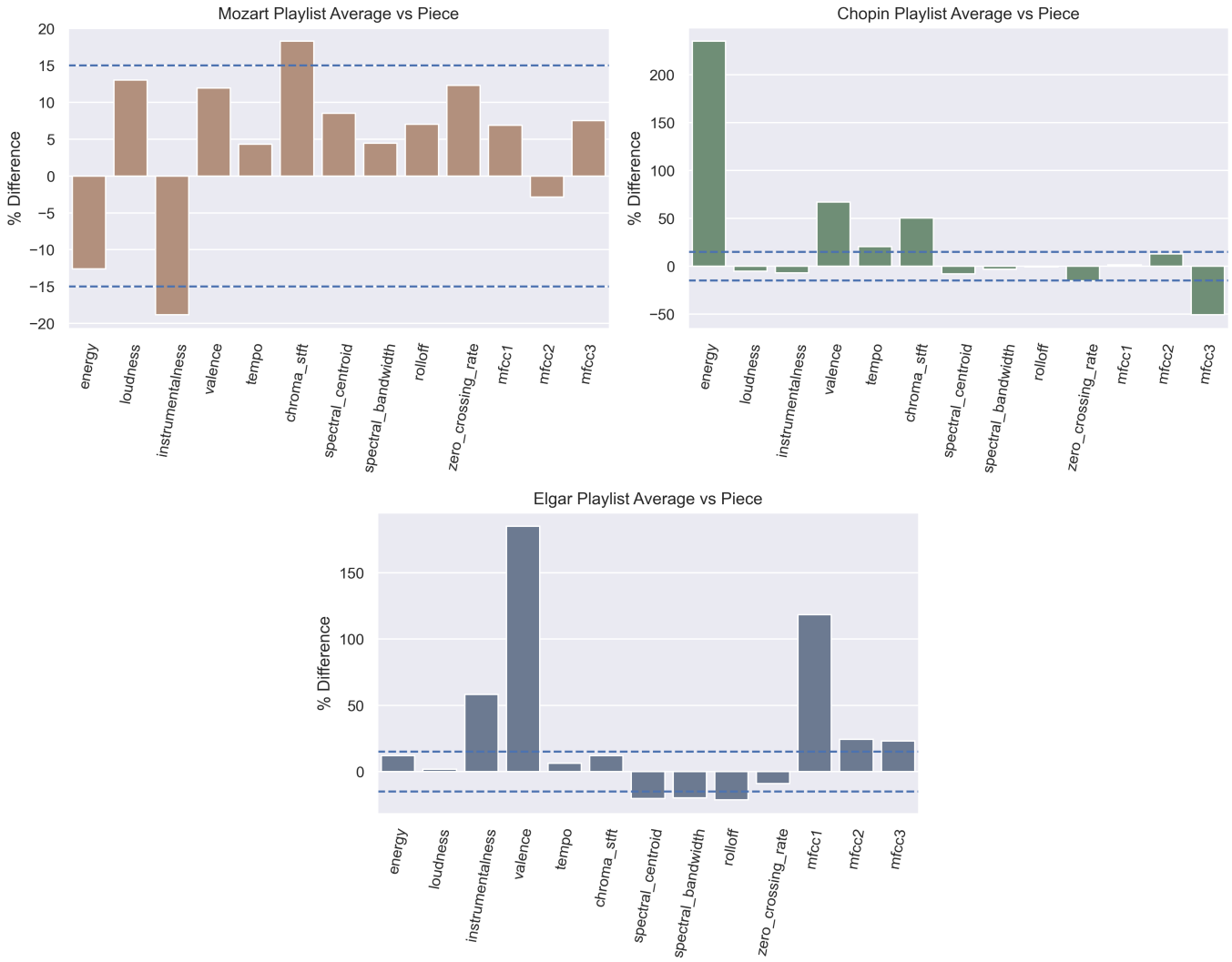


Figure 8: %Deviations of Playlist average feature values from selected pieces. Dotted lines represent ±15% deviation

Piece Playlist	Silhouette Score
Mozart	0.11
Chopin	0.18
Elgar	0.11

Table 2: Silhouette Scores of auto-generated playlists

Chapter 4: Discussion

4.1 Conclusions and Business Implications

Collaborative filtering is at the core of Spotify's recommendation algorithms and has made it one of the most successful streaming platforms in the world. The incorporation of content based filtering makes the algorithms even more powerful by allowing for further personalisation capabilities. However, the way listeners perceive music is mostly subjective, thus making song-based recommendations significantly challenging. This is particularly the case for classical music due to its complex structure and vast repertoire.

This project aimed to reproduce Spotify's radio algorithm specifically for western classical music, by combining three different feature categories and using unsupervised learning techniques. The main observation that was made during the research phase of this project was that many playlists generated by Spotify's radio contained pieces that were written in completely different eras than the chosen song. Although one could argue that two pieces written 200 years apart can still be similar, narrowing down the search can firstly speed up the algorithm, and secondly ensure that the recommendations are more accurate. This is because pieces written within the same classical era share similar characteristics.

Figure 3 presents a heat map of correlations between Spotify and Spectral features. The majority of variables are not highly correlated with each other. However, certain pairs such as loudness with energy and valence with danceability are strongly positively correlated. This makes intuitive sense, as the more energetic a piece of music is the louder it gets, while a high danceability measure can lead to a more uplifting mood and thus a higher valence score. Furthermore, as described on table 1, a loud piece results in a higher spectral centroid value. This is evident from the slight positive correlation of 0.45 between loudness and the spectral centroid.

As the spectral centroid, bandwidth, rolloff and zero crossing rate are all dependent on the spectrogram of a piece, and more specifically on the sub-bands formed, we can see that they are all highly correlated with each other. This is not the case for the chroma STFT variable, as it is defined by the distribution of notes. Finally, the MFCC box plot shown in figure 4 proves that the first few coefficients retain the most information about a piece.

The spectrograms generated in figure 5 can give a lot of insight about a piece's characteristics and therefore prove that spectral features are vital in the context of MIR. Chopin's

nocturne is a piece for solo piano that mainly uses mid-range notes. This is clearly shown in its respective spectrogram, where the dark upper part indicates that the piece does not consist of significantly high frequencies or bright sounds. Instead, since there is only one instrument involved, the predominant frequencies are observed at the bottom of the spectrum, while the fainter colours above represent the overtones of the sound. In addition, the lack of too many bright colours suggest that the overall loudness of the piece is not as high. The spectrograms for Mozart and Elgar on the other hand, are filled with bright colours across the entire plot, reflecting their orchestral nature, loud sounds and use of high frequencies that presumably come from higher pitched instruments like violins and flutes.

When evaluating the performance of the algorithm, the keys and modes of each playlist were compared to those of the selected pieces. Figure 7 shows that the playlists for Mozart's and Chopin's pieces contain a high number of tracks in the same key as the original tracks. However, despite the piece's key revealing a lot about its character, music does not need to be in the same key to be similar. It is therefore good that the algorithm returns a range of pieces from different keys with only a slight focus on the original key. A particularly interesting observation is that for Mozart's playlist, the majority of pieces are in the key of D. While this is not the same key as the selected piece (key=G), it is in fact the dominant key of the D minor scale. Many composers often shift to the dominant (fifth) key of a scale at certain points throughout the piece. It may therefore be the case that the particular sample extracted happened to be at such a point, and was therefore captured by the chroma STFT feature.

As opposed to the key of a piece, the modality is an important feature that can influence the character of a piece. Figure 7 counts the number of major and minor pieces in each playlist. It is clear from the plot that the majority of pieces across all three playlists are written in major scales, despite the mode of the original track. This suggests that the algorithm does not capture the mode of the piece very well, despite the inclusion of the equivalent feature. This is also clear in figure 8, where the playlists for Chopin and Elgar in particular have high valence deviation. Valence is a measure of the overall positiveness of a piece, and can be impacted by modality since major keys are mainly associated with positive, uplifting melodies while minor keys create more dramatic and sentimental atmospheres.

When comparing the percent deviations of the average playlist values to those of the original pieces, it is satisfactory to see that most stay within a 15% margin. Mozart's playlist is particularly accurate, with just instrumentalness and chroma STFT deviating further away from the selected

piece. The lower instrumentalness suggests that more vocal pieces have been included. However, a quick look in the playlist proves that is no vocal music involved. The lower instrumentalness could therefore mean that there are sounds closer to the human voice, such as violins and cellos. In Chopin's playlist, the significantly higher energy indicates that the playlist includes many pieces that are more intense and uplifting than the original track. Finally, the valence in both Elgar's and Chopin's playlists is also higher, showing that the overall mood of the pieces has not been accurately captured most of the time.

The silhouette scores in table 2 suggest that the observations are correctly assigned into their respective clusters. However, as all three scores are closer to 0 than 1, it seems that there are some observations that are near the decision boundaries of the clusters. We can thus conclude that there are further improvements that can be made to enhance the performance of the algorithm.

On top of the quantitative evaluations, the playlists were also inspected by ear to understand the extent to which certain musical elements have been captured by the algorithm. It is interesting to see that Mozart's playlist consists of predominantly symphonic pieces with brighter sounds and a heavy focus on strings, Chopin's playlist is mainly made up of solo piano pieces, and Elgar's playlist has grande, intense and highly virtuosic music with the inclusion of many concertos. This proves that, despite the little differentiation of spectral features across composition categories in figure 5, the composition features were very helpful in forming relevant clusters. In addition, thanks to the chronological filtering that takes place at the beginning of the algorithm, each playlist focuses on the right classical era, thus containing similar compositional styles.

Listeners with classical music background are trained to recognise the characteristics of a piece and are often capable of recognising the exact era it originates from. Streaming platforms should therefore take extra care in curating accurate classical playlists, to avoid the churning of members to other platforms like Idagio which place a higher focus on playlist creation [35]. In addition, improved classical playlists can also encourage non-classical listeners to explore and learn more about the genre. Consequently, it is important for such streaming platforms to concentrate on improving such functionalities by not just improving their algorithms, but also by consulting experts in the classical music domain. This is crucial because, as we saw above, quantitative evaluation measures can often misrepresent the acoustic reality of such playlists.

4.2 Limitations

Due to the time consuming process of data collection only 2760 observations were considered, while a platform like Spotify generates playlists out of millions of observations. It was therefore impossible to replicate the exact functionality of Spotify's radio. However, certain techniques such as K-Means as opposed to Hierarchical clustering were preferred, so that the algorithm can be less computational intensive for larger datasets.

In addition, spectral features were extracted by using 10 second samples of each track. As classical music can change drastically throughout a piece, those samples may not be representative of the entire piece. For example, a 10 second sample of a piano concerto that only captures the orchestral part and not the solo instrument can make the algorithm think that this is in fact an orchestral piece. This could also be the reason why we did not see any drastic differences in the spectral feature distributions across compositional features in figure 5.

Although Principal Component Analysis is an easy-to-implement dimensionality reduction technique that can speed up machine learning algorithms, it comes with certain limitations. Due to the linear combination of features, it is often hard to identify the most important features, thus reducing interpretability. Additionally, there is a trade-off between dimensionality reduction and information loss, as selecting a small number of principal components leads to loss of vital information.

K-Means clustering is a popular unsupervised learning model which allowed us to create satisfactory music clusters based on the characteristics of each piece. However, it is known to be quite susceptible to outliers due to the centroids being dependent on the mean value of the cluster. Since K-Means seeks to minimise the within-cluster variation, it can often place outliers in clusters without severely affecting the overall mean. In the context of classical music clustering, this can mean that a playlist of a slow-paced, relaxing piece can also contain a very small number of high tempo and intense tracks. Furthermore, the elbow method of selecting the optimal number of clusters is often subjective depending on how the data scientist or algorithm locates the elbow point and the result may change depending on the maximum number of clusters considered.

4.3 Further Work

There are numerous areas that can be explored to improve the efficiency of a classical music clustering algorithm. Firstly, longer samples can be collected to make the spectral feature extraction more robust. However, this may also significantly lengthen the running time of the algorithm. An alternative approach is to take multiple samples from the same piece, extract the spectral features of each sample and then take the average.

One of the main elements of the algorithm built is that is firstly extracts a sample of tracks that were written within the same period as the chosen piece. In order to further improve this step, Hierarchical clustering can be used to first group composers based on similarities in their compositional styles. Such an algorithm can also take into account features like the origin of the composer, since it highly influences their writing. Another interesting exploration would be to transform the composers' data into a network and carry out community detection algorithms. This approach steers away from conventional machine learning methods while it focuses more on graph theory and topology.

In order to improve the actual clustering of the music, one could firstly consider a series of unsupervised learning models like Neural Networks and compare their performance, or combine them to create an ensemble learning model. In addition, Natural Language Processing can be used to generate further features based on each track's title. For example, one can create groups out of instruments mentioned or identify operatic terms such as Aria and Recitative. Finally, despite the unlabelled nature of the dataset, there are certain feature importance techniques that can identify the features that explain the most variability. For example, the Laplacian Score can be used to evaluate the locality preserving power of a feature by calculating the feature-specific distance between observations. The most important features are the ones with the lowest Laplacian Score [36].

Finally, the model performance evaluation can be improved both quantitatively and qualitatively. To assess the clustering efficiency, a loop can be written that runs the algorithm for every piece in the dataset and calculates the equivalent silhouette score. The average of all scores can then be computed to obtain an overall accuracy measure. In terms of qualitative evaluation, the algorithm can be deployed as an application and shared with a large number of listeners, who will be requested to rate the playlists generated.

References

[1] Whitehouse, K. (2021) *How Spotify Uses Artificial Intelligence, Big Data, and Machine Learning*. Available from: https://www.datasciencecentral.com/profiles/blogs/6448529:BlogPost:1041799 [Accessed 16 August 2021].

[2] Spotify Support Team (2019) *How does Spotify Radio work?*. Available from: https://community.spotify.com/t5/FAQs/How-does-Spotify-Radio-work/ta-p/4757060 [Accessed 16 August 2021].

[3] Dieleman, S. (2020) *Recommending music on Spotify with deep learning*. Available from: https://benanne.github.io/2014/08/05/spotify-cnns.html [Accessed 17 August 2021].

[4] Liu, Y. (2014) *Exploring drawbacks in music recommender systems*. Available from: https://www.diva-portal.org/smash/get/diva2:896794/FULLTEXT01.pdf [Accessed 17 August 2021].

[5] Snickars, P. (2017) *More of the Same - On Spotify Radio.* Culture Unbound: Journal of Current Cultural Research. Available from: https://www.researchgate.net/publication/320738138_More_of_the_Same_-_On_Spotify_Radio [Accessed 18 August 2021].

[6] bradamant (2014) Comment on: *Classical radio has terrible selections and is really repetitive*. [online] Available from: https://community.spotify.com/t5/Other-Podcasts-Partners-etc/Classical-radio-has-terrible-selections-and-is-really-repetitive/td-p/888301 [Accessed 16 August 2021].

[7] Jonahman10, (2014) Comment on: *Classical radio has terrible selections and is really repetitive*. Available from: https://community.spotify.com/t5/Other-Podcasts-Partners-etc/Classical-radio-has-terrible-selections-and-is-really-repetitive/td-p/888301 [Accessed 16 August 2021].

[8] tolgold77 (2020) Comment on: *Classical radio has terrible selections and is really repetitive*. Available from: https://community.spotify.com/t5/Other-Podcasts-Partners-etc/Classical-radio-has-terrible-selections-and-is-really-repetitive/td-p/888301 [Accessed 16 August 2021].

[9] Kessler, M. (1966) *Toward musical information retrieval*. Perspectives of New Music.

[10] Downie, J. (2005) *Music information retrieval*. Annual Review of Information Science and Technology. Available from: https://doi.org/10.1002/aris.1440370108 [Accessed 19 August 2021].

[11] Cuthbert, M. S., Ariza, C. and Friedland L (2011) *Feature Extraction and Machine Learning on Symbolic Music using the music21 Toolkit*. Available from: http://web.mit.edu/music21/papers/Cuthbert_Ariza_Friedland_Feature-Extraction_ISMIR_2011.pdf [Accessed 19 August 2021].

[12] Chiu, D., Liu, D. and Wang, Y. (2013) *Bartok: Music Time Period Classification.* Available from: http://cs229.stanford.edu/proj2013/Bartok-Final-F.pdf [Accessed 19 August 2021].

[13] Basili, R., Serafini, A. and Stellato, A. (2004) *Classification of musical genre: A machine learning approach*. Available from: http://art.uniroma2.it/research/musicIR/BasSeraStel_ISMIR04.pdf [Accessed 19 August 2021].

[14] Weiss, C. (2017) *Computational methods for tonality based style analysis of classical music audio recordings*. Available from: https://www.db-thueringen.de/servlets/MCRFileNodeServlet/dbt_derivate_00039054/ilm1-2017000293.pdf [Accessed 19 August 2021].

[15] Langensiepen, C., Cripps, A. and Cant, R. (2018) *Using PCA and K-Means to Predict Likeable Songs from Playlist Information*. Available from: https://ieeexplore.ieee.org/document/8588173 [Accessed 19 August 2021].

[16] Vlerick, A. (2020) *K-Means clustering using Spotify song features*. Available from: https://towardsdatascience.com/k-means-clustering-using-spotify-song-features-9eb7d53d105c [Accessed 19 August 2021].

[17] Thibeault, M. D. (2011) *Learning From Looking at Sound: Using Multimedia Spectrograms to Explore World Music*, General Music Today. Available from: https://doi.org/10.1177/1048371311414050 [Accessed 19 August 2021].

[18] McFee, B., Raffel, C., Liang, D., Ellis, D. P. W., McVicar, M., Battenberg, E. and Nieto, O. (2015) *librosa: Audio and Music Signal Analysis in Python.* Available from: https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.701.4288&rep=rep1&type=pdf [Accessed 19 August 2021].

[19] Paiva, R. P., Mendes, A. J., Mendes T. and Cardoso, A. (2004) *Classification of recorded classical music: A methodology and a comparative study.* Available from: https://www.researchgate.net/publication/237656052_CLASSIFICATION_OF_RECORDED_CLASSICAL_MUSIC_A_METHODOLOGY_AND_A_COMPARATIVE_STUDY *[Accessed 19 August 2021].*

[20] Shingte, G. and d'Aquin, M. (2019) *Unsupervised Learning Approach for Identifying Sub-genres in Music Scores.* Available from: http://ceur-ws.org/Vol-2563/aics_7.pdf [Accessed 19 August 2021].

[21] Classical Net (2021) *Classical Net - Composer Data - Birthday List*. Available from: http://www.classical.net/music/composer/dates/comp2.php [Accessed 14 August 2021].

[22] Jolliffe, I. (2005) *Principal Component Analysis*. Encyclopedia of Statistics in Behavioral Science. Available from: https://doi.org/10.1002/0470013192.bsa501 [Accessed 19 August 2021].

[23] Jolliffe I. and Cadima J. (2016) *Principal component analysis: a review and recent developments.* Philosophical transactions of the Royal Society A. Available from: http://doi.org/10.1098/rsta.2015.0202 [Accessed 19 August 2021].

[24] Wedel, M. And Kamakura W. A. (2000) *Market segmentation: conceptual and methodological foundations*, 2nd edn. Kluwer Academic, Boston, NE

[25] James, G., Witten, D., Hastie, T. and Tibshirani, R. (2013) *An introduction to statistical learning: with applications in R*. New York: Springer, Chapter 10.

[26] master_abhig (2021) *Difference between K Means and Hierarchical Clustering*. GeeksforGeeks. Available from: https://www.geeksforgeeks.org/difference-between-k-means-and-hierarchical-clustering/ [Accessed 24 August 2021].

[27] Nanjundan, S., Sankaran, S., Arjun, C. and Anand, G. (2019). *Identifying the number of clusters for K-Means: A hypersphere density based approach.* Available from: https://arxiv.org/abs/1912.00643 [Accessed 24 August 2021].

[28] scikit-yb.org (2019) *Elbow Method — Yellowbrick v1.3.post1 documentation*. Available from: https://www.scikit-yb.org/en/latest/api/cluster/elbow.html [Accessed 24 August 2021].

[29] Medium (2021) *A Comprehensive Introduction to Clustering Methods*. Available from: https://shairozsohail.medium.com/a-comprehensive-introduction-to-clustering-methods-1e1e4f95b501 [Accessed 24 August 2021].

[30] scikit-learn.org (2021) *Selecting the number of clusters with silhouette analysis on KMeans clustering — scikit-learn 0.24.2 documentation*.Available from: https://scikit-learn.org/stable/auto_examples/cluster/plot_kmeans_silhouette_analysis.html [Accessed 24 August 2021].

[31] Changsheng Xu, Namunu C. Maddage, Xi Shao, Fang Cao, Qi Tian (2003) *Musical genre classification using support vector machines*. Available from: https://www.researchgate.net/publication/4015150_Musical_genre_classification_using_support_vector_machines [Accessed 19 August 2021].

[32] Comrey, A. L., and Lee, H. B. (1992). *A first course in factor analysis* (2nd ed.). Lawrence Erlbaum Associates, Inc.

[33] developer.spotify.com (2021) *Web API Reference | Spotify for Developers | AudioFeaturesObject*. Available from: https://developer.spotify.com/documentation/web-api/reference/#objects-index [Accessed 24 August 2021].

[34] Sandhya, P., Spoorthy, V., Koolagudi S. G. and Sobhana N. V. (2020) *Spectral Features for Emotional Speaker Recognition.* Third International Conference on Advances in Electronics, Computers and Communications (ICAECC). Available from: https://ieeexplore.ieee.org/document/9339502 [Accessed 24 August 2021].

[35] bachtrack.com (2021) *Six of the best classical music streaming services in 2021*. Available from: https://bachtrack.com/feature-best-classical-music-streaming-services-apple-idagio-primephonic-qobuz-spotify-tidal-february-2021 [Accessed 25 August 2021].

[36] Xiaofei, H., Deng, C., and Partha, N. (2005) *Laplacian score for feature selection*. In Proceedings of the 18th International Conference on Neural Information Processing Systems (NIPS'05). MIT Press, Cambridge, MA, USA, 507–514.

[37] Richter, F. (2020) *Infographic: Classical Composers in the Digital Age*. Statista Infographics. Available from: https://www.statista.com/chart/23793/classical-composers-on-spotify/ [Accessed 3 September 2021].

Appendices

Appendix A: Spotify Radio Examples

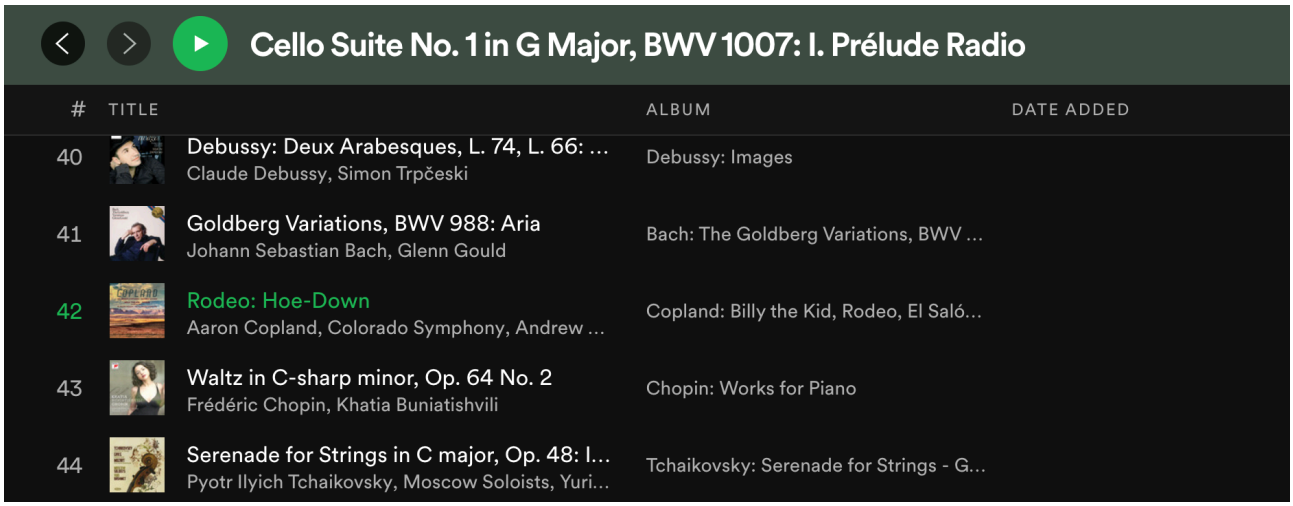


Figure 9: Screenshot of Spotify radio of Bach's cello prelude from suite No.1

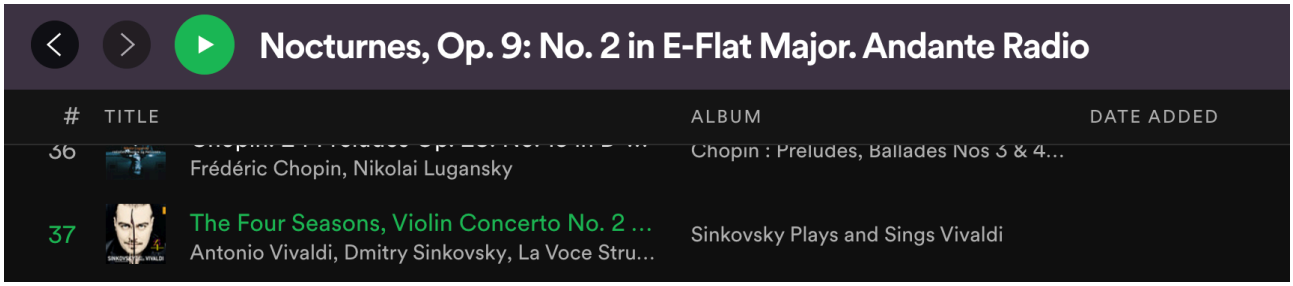


Figure 10: Screenshot of Spotify radio for Chopin's Necturne No.2. While this piece is a calm piece for solo piano, the playlist includes the 'Winter' from Vivaldi's Four Seasons; an intense and dramatic movement for orchestra and violin. The Four Seasons were written in the Baroque era while Chopin is a classical era composer.

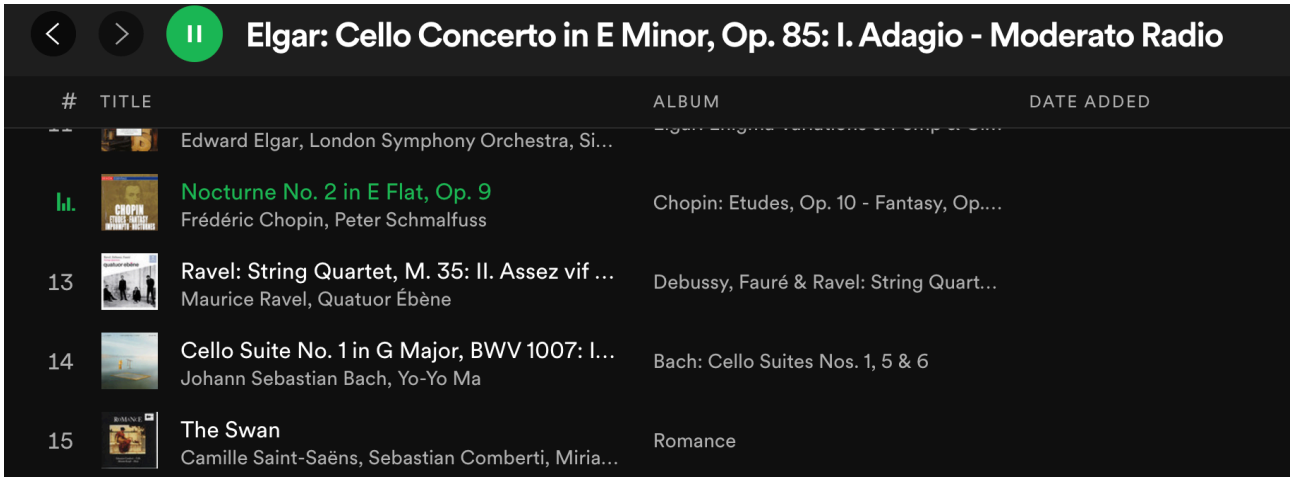


Figure 11: Screenshot of Spotify radio for the first movement of Elgar's cello concerto; a grand and dramatic piece with rich textures for orchestra and solo cello. This contrasts the recommended nocturne No.2 described above. In addition, other famous cello pieces like Bach's prelude and Saint-Saens' The Swan are included without necessarily containing similar musical features to the concerto.

Appendix B: Programming

B.1 Coding File Descriptions

File	Description
GSA.py	The source code for the Generalised Spotify Analyser library
spotifyConstants.py	A python script that includes Spotify Developer account credentials that are used within GSA's functions
data_collection.py	A python script that was used to collect music data from Spotify and extract relevant features
mp3_to_wav.R	An R script used to convert mp3 to WAV files.
algorithm.py	The main project algorithm that takes 3 select pieces as inputs, filters the dataset to include the relevant classical eras and performs PCA and K-Means clustering to produce similar playlist
EDA.py	A python script that performs exploratory data analysis of the dataset and creates visualisations for the algorithm results

B.2 Libraries Used

Libraries / Packages	Documentation
Spotipy	https://spotipy.readthedocs.io/en/2.19.0/
Pandas	https://pandas.pydata.org
GSA	https://github.com/OleAd/GeneralizedSpotifyAnalyser
Pydub	https://github.com/jiaaro/pydub
libROSA	https://librosa.org
tuneR	https://cran.r-project.org/web/packages/tuneR/tuneR.pdf
Scikit-Learn	https://scikit-learn.org/stable/

Appendix C: Music Terminology

C.1 Classical Era Characteristics

Classical Era	Characteristics	Popular Composers
Baroque	Polyphonic textures, repeated melodies, sudden changes in dynamics and frequent use of the harpsichord.	Bach, Vivaldi, Handel
Classical	Single melody with accompaniment, short and clear melodic patterns, more emphasis on instrumental music	Mozart, Haydn, Beethoven
Romantic	Longer emotional melodies, great variations in pitch, dynamics and rhythm, use of unexpected keys	Brahms, Verdi, Chopin, Schumann, Schubert
20th Century	Increased use of dissonance, huge variety in style, asymmetrical rhythms, external influences such as jazz	Schoenberg , Stravinsky, Copland, Glass

C.2 Definitions

Accidental: A note that does not belong to the scale or mode indicated by the overall key signature of a piece.

Aria: A part of a larger operatic work. It is a piece written for one voice and can either be with or without accompaniment.

Dissonance: The lack of harmony among pitches.

Dominant: The fifth note of a diatonic scale.

Harmony: A combination of multiple pitches.

Key: The group of notes on which a music composition is based on.

MIDI file: Musical Instrument Digital Interface file. This is a file that contains information about the notes, volumes, sounds, and effects of a piece of music.

Modality: The type of scale the piece is based on. Western classical pieces are either based on major or minor scales.

Recitative: A, usually operative, type of singing that is closer to speaking than singing.

Scale: A set of ordered notes.

Semitone: The smallest interval between two notes that is used in western classical music.

Tempo: The speed of a piece

Tonic: The first note of a diatonic scale.