UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
ENGENHARIA DE COMPUTAÇÃO

RAFAEL LUCAS FERREIRA GASPAROTO

ANÁLISE MUSICAL: MÉTODO DE AGRUPAMENTO K-MEANS

PROJETO DE BANCO DE DADOS

TOLEDO
2022

SUMÁRIO

1. Introdução

A música é um conjunto de sons que possuem uma harmonia, por ser uma das expressões artísticas mais primitivas da civilização humana ela contém inúmeros ritmos e timbres, apesar disto é possível notar músicas com semelhanças sonoras, para classificar se uma determinada música pertence a um certo grupo foi criado os gêneros musicais, cada gênero possui características únicas. A proposta deste projeto é encontrar características que contribuem para definir um gênero musical e através do uso do método K-Means, em Python, agrupar as músicas por gênero.

2. Coleta de Dados

2.1 API Spotify

Para conseguir os dados necessários foi utilizado a API do Spotify, com ela é possível requisitar as músicas contidas dentro de uma playlist por gênero. A API disponibiliza diversos dados de uma música, alguns mais técnicos como por exemplo a “danceability” que mensura o quão boa para dançar uma música é de acordo com os batimentos por minuto, estabilidade do ritmo, força da batida e regularidade geral, outros valores também disponíveis são como o ID do artista, nome da música e etc.

```
def call_playlist(creator, playlist_id):
    playlist_features_list = ["artist","album","artist_id","track_name", "track_id", "danceability","energy","key","loudness",

    playlist_df = pd.DataFrame(columns = playlist_features_list)

    playlist = sp.user_playlist_tracks(creator, playlist_id)["items"]
    for track in playlist:
        playlist_features = {}
        playlist_features["artist"] = track["track"]["album"]["artists"][0]["name"]
        playlist_features["album"] = track["track"]["album"]["name"]
        playlist_features["track_name"] = track["track"]["name"]
        playlist_features["track_id"] = track["track"]["id"]
        playlist_features["artist_id"] = track["track"]["artists"][0]["id"]

        audio_features = sp.audio_features(playlist_features["track_id"])[0]
        for feature in playlist_features_list[5:]:
            playlist_features[feature] = audio_features[feature]

        track_df = pd.DataFrame(playlist_features, index = [0])
        playlist_df = pd.concat([playlist_df, track_df], ignore_index = True)

    return playlist_df
```

Código 1 - Função Para Requisitar as Playlists

2.2 Extração e análise das playlists

Foram selecionadas 4 playlists de gêneros distintos para a análise, dentre elas uma de pagode, rock, ópera e lo-fi. O primeiro passo é analisar quais as melhores características técnicas que definem um gênero musical, para isto foram feitos gráficos de correlação para cada playlist.

Abaixo é possível visualizar os gráficos de correlação de todas as características de cada playlist.

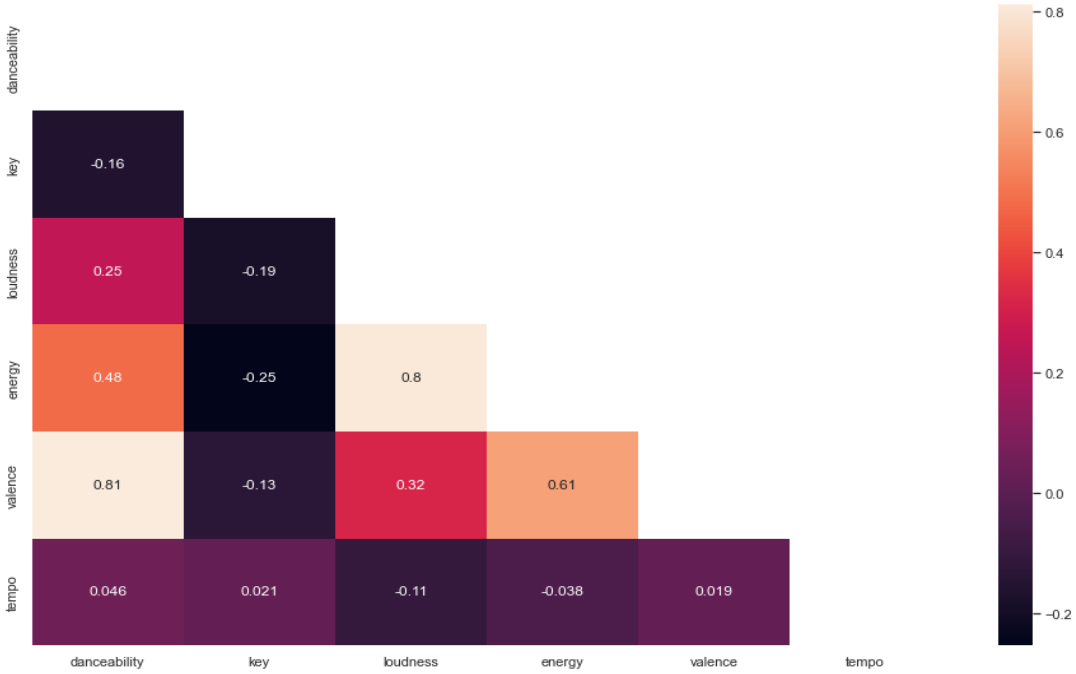

Gráfico 1 - Correlações Playlist de Ópera

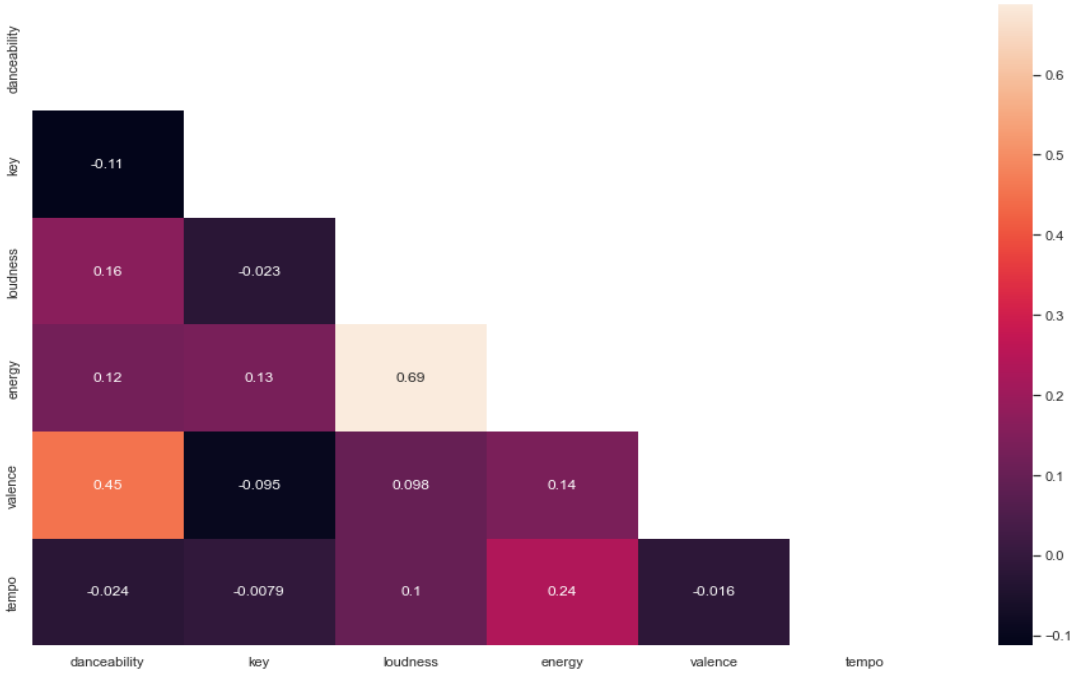

Gráfico 2 - Correlações Playlist de Lo-Fi

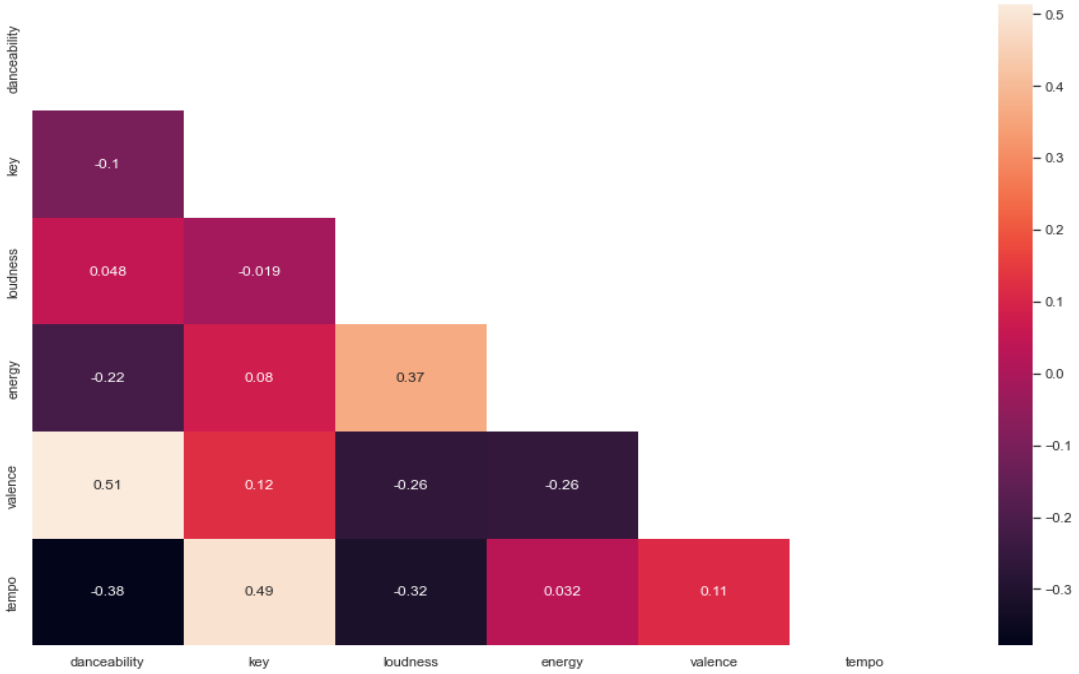

Gráfico 3 - Correlações Playlist de Rock

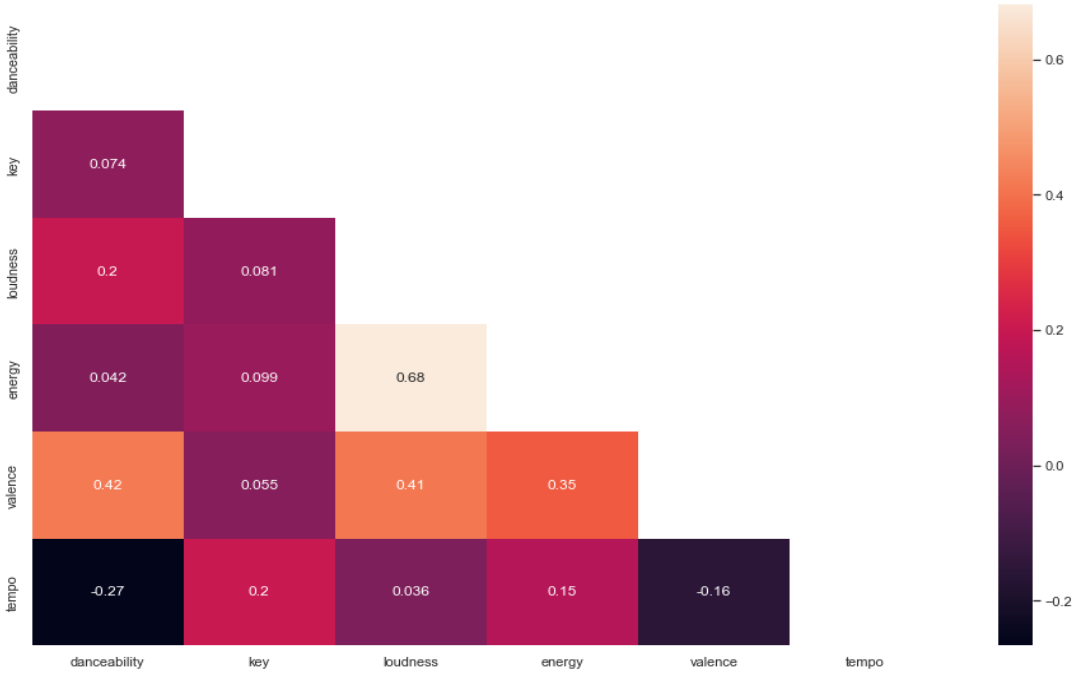

Gráfico 4 - Correlações Playlist de Pagode

Após uma análise dos gráficos de correlação foram selecionadas duas características, a “valence” que define em uma escala de 0.0 a 1.0 a positividade musical transmitida pela música, músicas com baixa “valence” tendem a ser músicas tristes ou raivosas enquanto músicas com uma alta “valence” são músicas alegres ou eufóricas, a outra característica selecionada foi a “energy”

que mensura em uma escala de 0.0 a 1.0 a atividade e intensidade da música, músicas com alta “energy” são rápidas e barulhentas caso contrário lentas e silenciosas.

Foram feitos gráficos para calcular o histograma bivariado para uma melhor visualização de concentração do relacionamento entre as duas características de “valence” e “energy”, além disto foi feito o uso uma aplicação de regressão linear para estimar o relacionamento entre as variáveis.

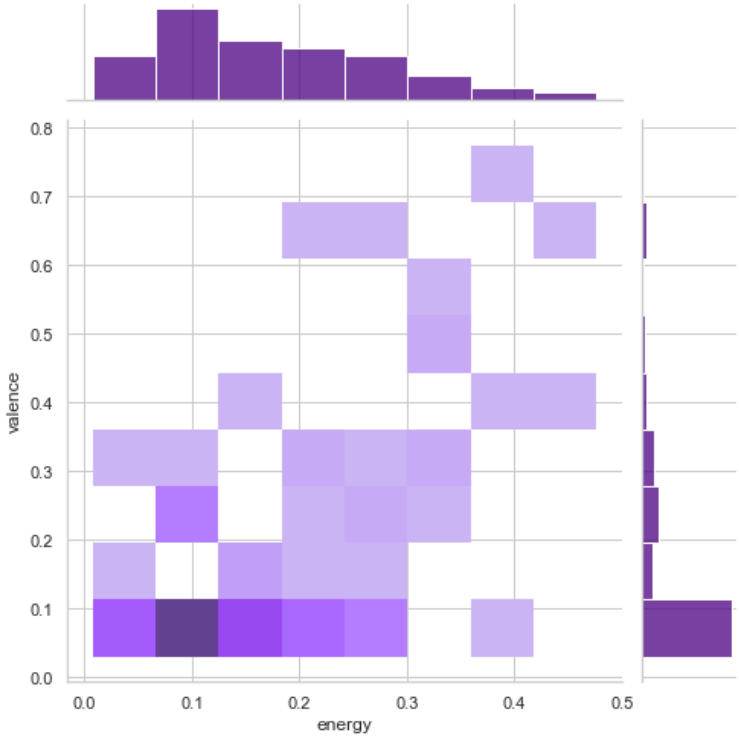

Gráfico 5 - Histograma Bivariado Playlist de Ópera

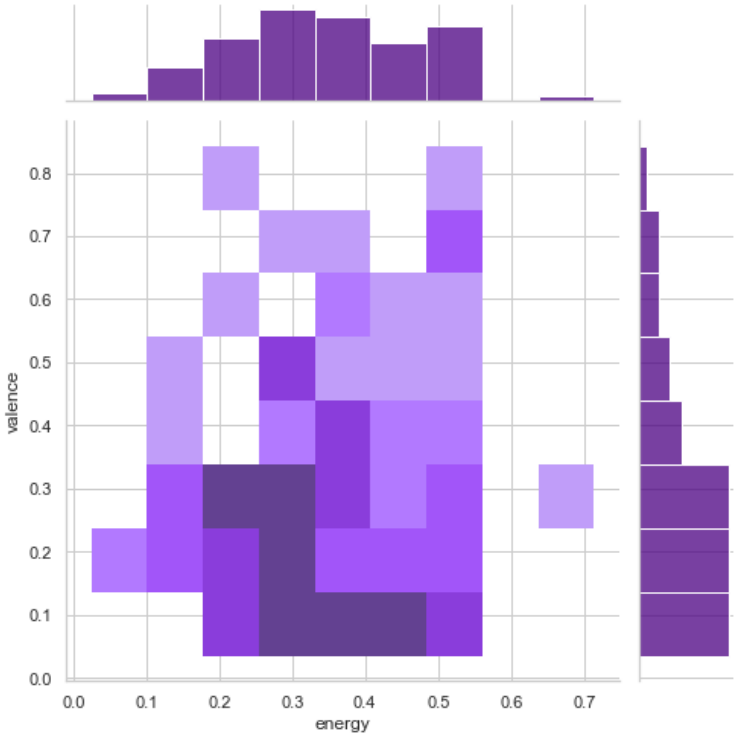

Gráfico 6 - Histograma Bivariado Playlist de Lo-fi

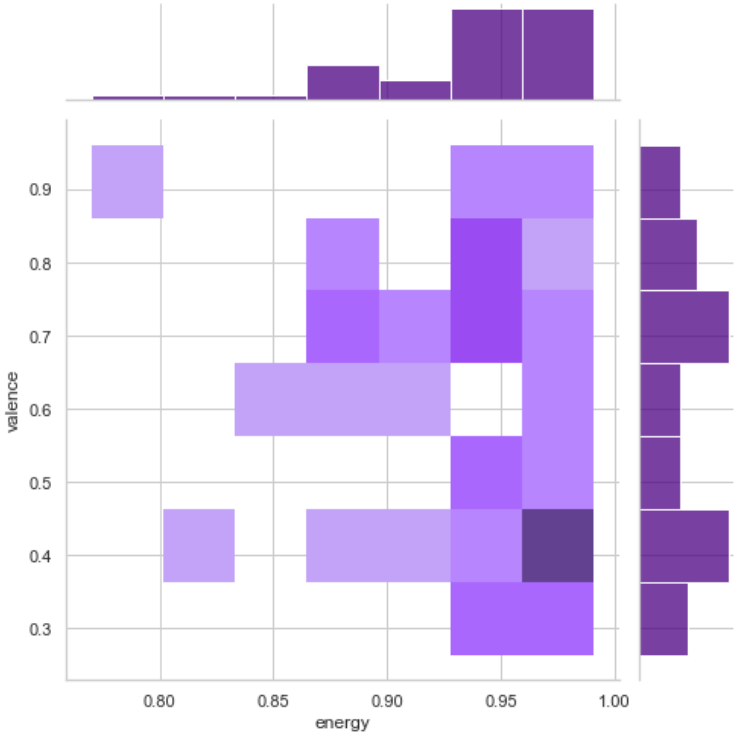

Gráfico 7 - Histograma Bivariado Playlist de Rock

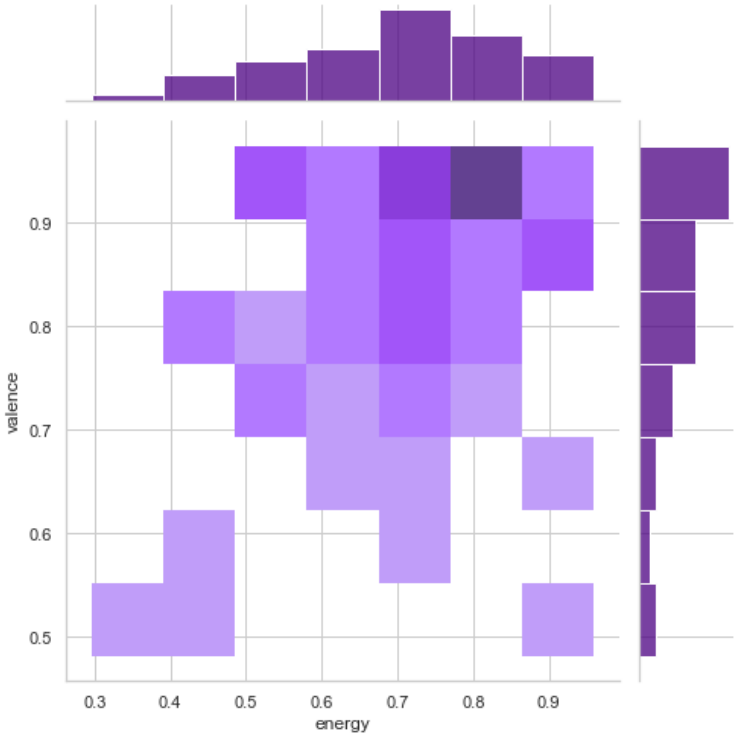

Gráfico 8 - Histograma Bivariado Playlist de Pagode

Analisando o gráfico 8 que corresponde a playlist de pagode é possível notar uma concentração das músicas na extremidade direita do gráfico, correspondendo a músicas com uma alta “valence” e “energy”, a interpretação que se dá é a de que músicas do gênero pagode tendem a ser alegres e rápidas. Em contraste o gráfico 5 da playlist de ópera possui uma maior concentração de músicas na extremidade esquerda do gráfico, correspondendo a músicas calmas e melancólicas.

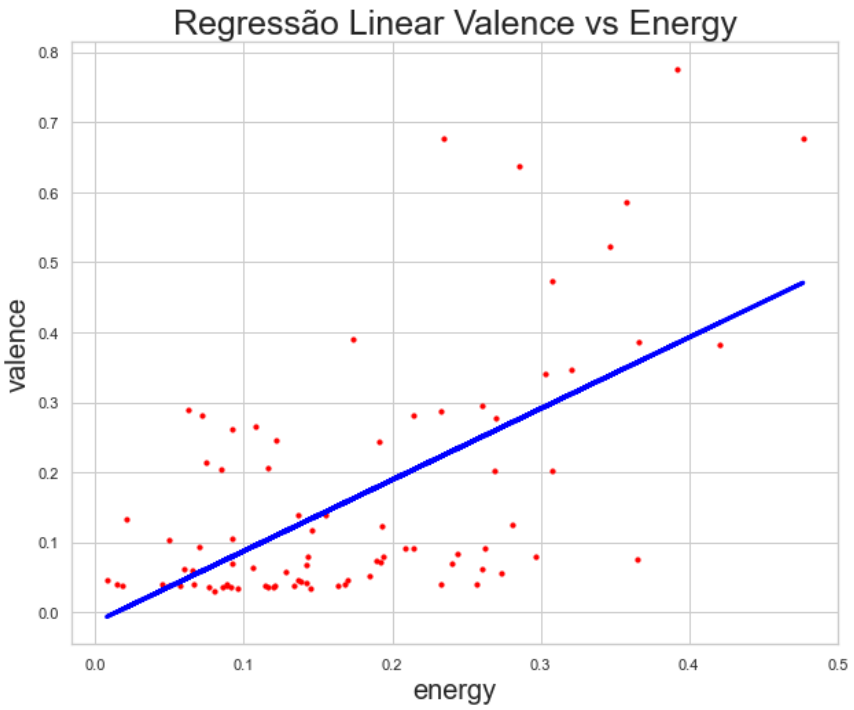

Gráfico 9 - Regressão Linear Playlist de Ópera

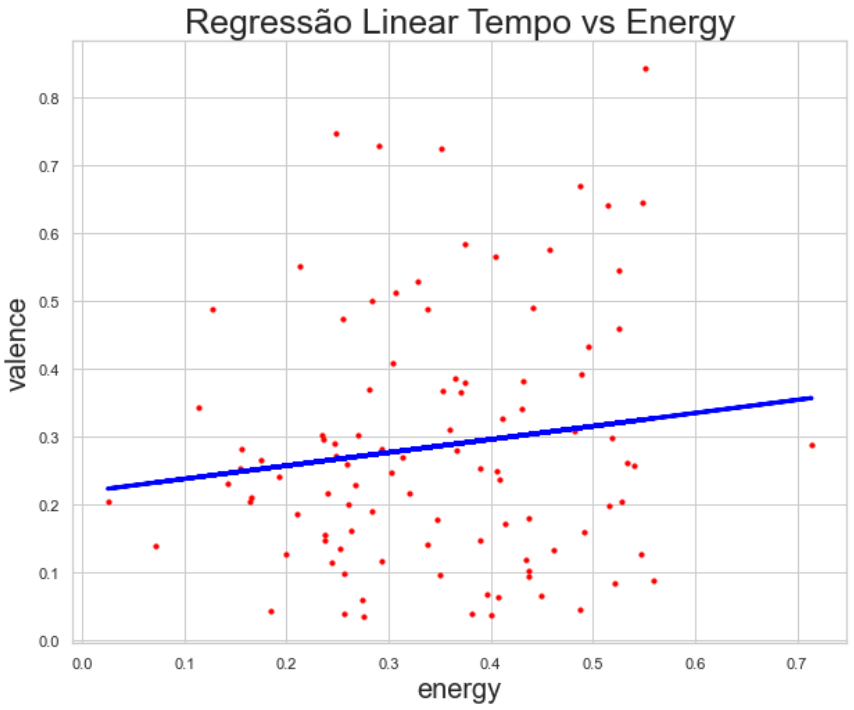

Gráfico 10 - Regressão Linear Playlist de Lo-fi

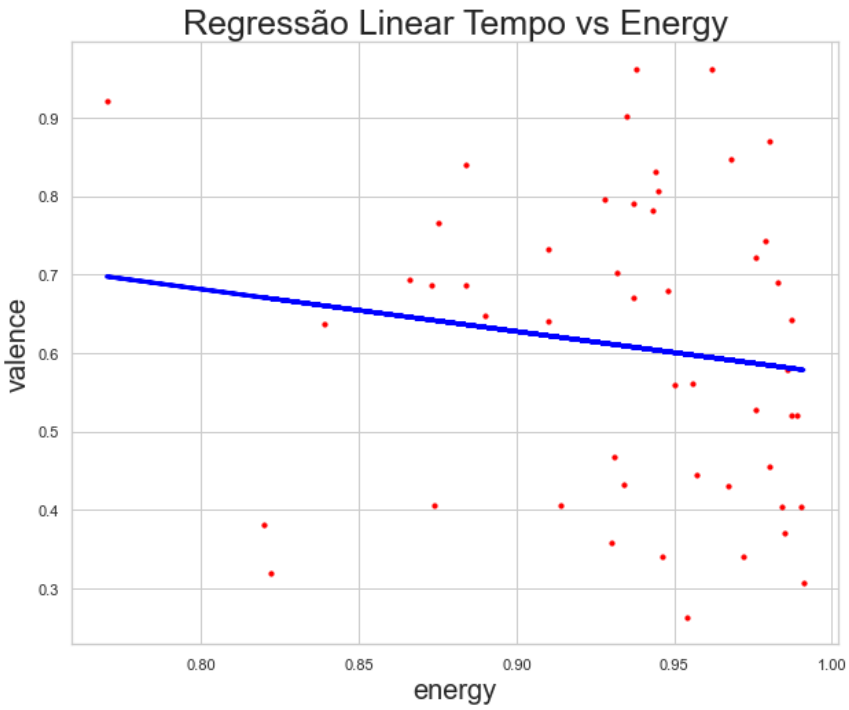

Gráfico 10 - Regressão Linear Playlist de Rock

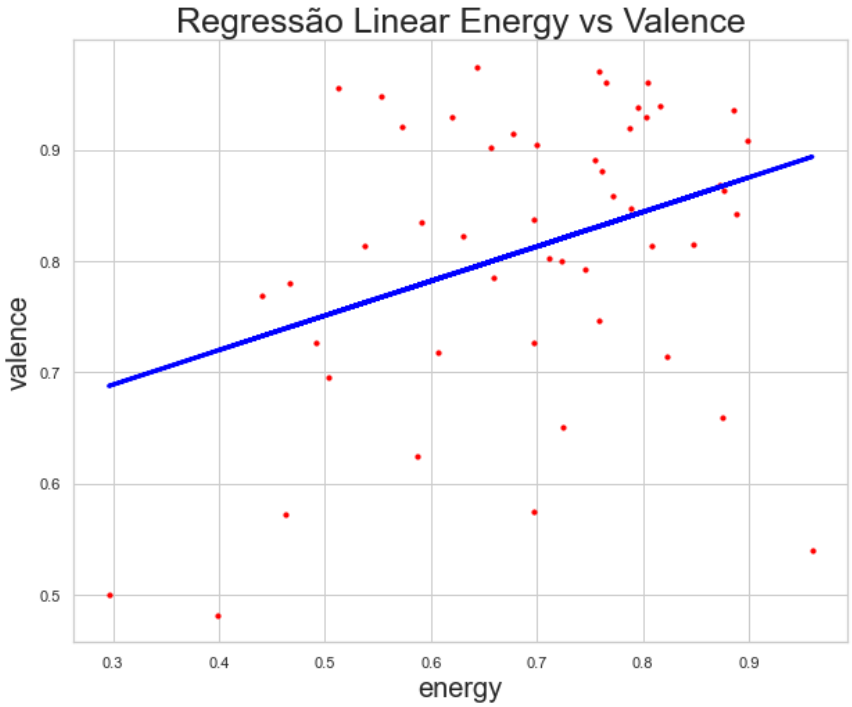

Gráfico 10 - Regressão Linear Playlist de Pagode

Após a análise de cada playlist é preciso juntar as playlists e saber qual é o gênero da música para a comparação futuro. Como a API não disponibiliza o gênero por música e sim por artista, foi feito uma requisição do gênero musical através da ID do artista de cada música e combinado cada tupla da lista de acordo com o gênero musical que o artista canta.

```
frames = [rock, pagode, lofi, opera]
results = pd.concat(frames)
results = results.reset_index(drop=True)
results
```

Figura 2 - Combinação das Playlists em um DataFrame

```
artist_genres = []
for a_id in results.artist_id:
    artist = sp.artist(a_id)
    artist_genres.append(artist['genres'])
```

Figura 3 - Função de Requisição de Gêneros

2.3 Limpeza e seleção dos dados

Como nem todos os artistas disponibilizam o seu gênero musical foi feita a exclusão dessas músicas para não interferir no resultado final. Então com a lista de músicas que passaram por essa filtragem é selecionado as duas características de “valence” e “energy” para aplicar o algoritmo K-Means.

```
results = results.assign(artist_genres=artist_genres)
results['artist_genres_str'] = results['artist_genres'].astype('str')
idxName = results[results['artist_genres_str']=='[]'].index
results.drop(idxName, inplace=True)
results = results.reset_index(drop=True)

dx = results[['energy', 'valence']]
```

Figura 4 - Limpeza e Seleção de Dados

3. Treinamento do algoritmo

3.1 Calculando a quantidade de clusters

O cálculo da quantidade de clusters é necessário em casos de aprendizagem não supervisionada, o modelo implementado neste projeto é de aprendizagem supervisionada, logo é

conhecido que a quantidade de clusters necessários são 4 pois são 4 gêneros musicais, no entanto a função foi implementada para verificar se de fato correspondia a necessidade de 4 clusters.

A quantidade ideal de clusters corresponde ao momento em que o valor do WCSS diminui paralelamente ao eixo x do gráfico, o valor WCC é uma soma da distância ao quadrado entre cada ponto e o centróide em um cluster, o gráfico 11 mostra que de fato 4 clusters são uma quantidade ideal.

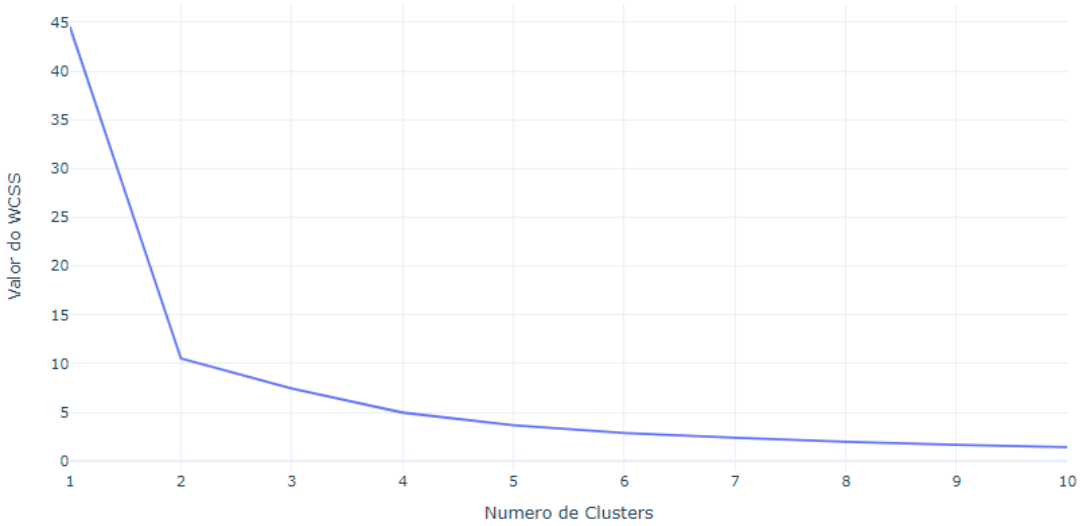

Gráfico 11 - Quantidade de Clusters

3.2 Aplicando K-Means

A aplicação do algoritmo tem como resultado um valor de 0-3 representando o cluster correspondente de cada entrada. O valor final é colocado em um novo DataFrame com os valores de “valence”, “energy”, “artist_genres” e “cluster” correspondentes de cada música analisada para a visualização dos resultados encontrados.

```
kmeans_musicas = KMeans(n_clusters=4, random_state=0)
results['cluster'] = kmeans_musicas.fit_predict(dx)

final_results = results[['valence', 'energy', 'artist_genres', 'cluster']]
```

Código 5 - Aplicação do Método K-Means

4. Resultados

O resultado final pode ser visualizado pelo gráfico 12 que mostra o agrupamento de 4 gêneros musicais de acordo com a sua “valence” e “energy”.

O método definiu como o valor de cluster 0 o gênero de música lo-fi, cluster 1 para o gênero pagode, cluster 2 para ópera e 3 para rock, correspondendo às cores azul, roxo, laranja e amarelo respectivamente.

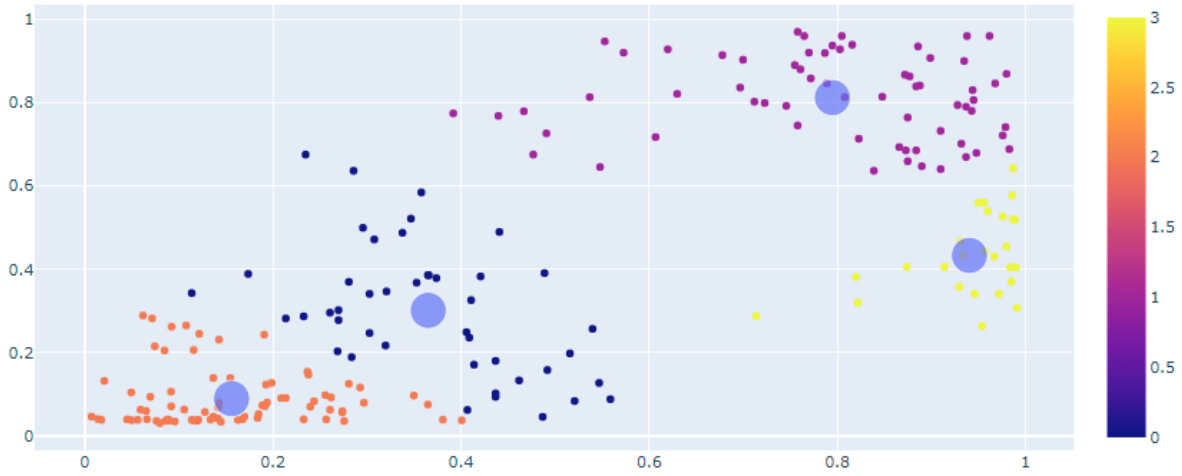

Gráfico 12 - Agrupamento das Músicas por Gênero

Para verificar se o método foi implementado corretamente e se a resposta é condizente com o esperado, é possível visualizar o cluster associado a cada música que se encontra no DataFrame final_results.

112	0.488	0.338	[japanese chillhop]	0
111	0.49	0.441	[new french touch]	0
110	0.391	0.489	[japanese chillhop]	0
109	0.343	0.114	[new french touch]	0
108	0.102	0.437	[japanese chillhop]	0
103	0.302	0.27	[new french touch]	0

Tabela 1 - Músicas Referentes ao Gênero Lo-fi

72	0.915	0.678	[pagode, samba moderno, samba paulista]	1
78	0.822	0.63	[pagode, pagode novo, samba moderno]	1
82	0.718	0.607	[pagode, samba moderno]	1
80	0.803	0.712	[bossa nova, mpb, pagode, partido alto, samba,...	1
81	0.936	0.886	[samba moderno]	1
83	0.938	0.795	[pagode, partido alto, samba, samba-enredo]	1

Tabela 2 - Músicas Referentes ao Gênero Pagode

172	0.0627	0.0596	[classical, french opera, french romanticism, ...	2
171	0.0368	0.116	[classical, italian opera, italian romanticism...	2
163	0.0388	0.134	[classical, italian opera, italian romanticism...	2
164	0.0562	0.274	[classical, italian opera, italian romanticism...	2
170	0.0624	0.261	[classical, italian opera, italian romanticism...	2
180	0.0732	0.189	[classical, italian opera, italian romanticism...	2

Tabela 2 - Músicas Referentes ao Gênero Ópera

18	0.467	0.931	[chicago hardcore, chicago punk, hardcore punk...	3
5	0.341	0.972	[pop punk, punk, rock, socal pop punk]	3
16	0.358	0.93	[punk, skate punk, socal pop punk]	3
15	0.561	0.956	[pop punk, punk, rock, socal pop punk]	3
30	0.308	0.991	[melodic hardcore, pop punk, punk, skate punk]	3
43	0.455	0.98	[glam punk, norwegian punk rock, norwegian roc...	3

Tabela 4 - Músicas Referentes ao Gênero Rock

A maioria das músicas foram classificadas de acordo com o gênero musical correspondente, no entanto dentro de um gênero existem músicas com pontos fora da curva que acabam sendo confundidas com outros grupos como é possível visualizar na tabela 5.

59	0.847	0.789	[pagode, samba]	1
49	0.847	0.968	[celtic punk, celtic rock, punk, skate punk]	1
50	0.961	0.805	[pagode, samba moderno]	1
55	0.793	0.746	[pagode, samba, samba moderno]	1
45	0.689	0.983	[pop punk, punk, ska punk, skate punk]	1

Tabela 5 - Músicas Não Agrupadas Como Esperado

5. Conclusão

Os resultados encontrados se mostraram promissores e condizentes com o que era esperado, o método K-Means foi capaz de agrupar adequadamente os gêneros musicais de acordo com a “valence” e “energy” de cada música. É possível para trabalhos futuros testar outras combinações de características para comparar os resultados, além da possibilidade de aplicar esses dados em algum outro projeto como por exemplo para prever um gênero musical de a partir de uma aplicação que poderia mensurar a movimentação do celular de uma pessoa e indicar o

gênero musical que ela poderia estar dançando levando em consideração a característica “danceability”, outro possibilidade seria desenvolver um aplicativo de recomendação de música que recomendaria músicas através de uma média de distância de clusters feita por várias combinações de características, com isso seria possível maximizar a probabilidade de a música ser parecida com a música colocada como entrada de recomendação semelhante.

6. Repositório

Para uma melhor compreensão é possível ter acesso ao código fonte completo no seguinte repositório do github: https://github.com/RafaelGasparoto/Analise_Musical.git

7. Referências

SAJI, Basil. In-depth Intuition of K-Means Clustering Algorithm in Machine Learning. [*S. l.*]: Analytics Vidhya, 20 jan. 2021. Disponível em: https://www.analyticsvidhya.com/blog/2021/01/in-depth-intuition-of-k-means-clustering-algorithm-in-machine-learning/#:~:text=For%20each%20value%20of%20K,value%20will%20start%20to%20decrease. Acesso em: 24 jun. 2022.

WATTS, Cameron. Extracting Song Data From the Spotify API Using Python: Taking advantage of the data Spotify keeps on its library, and using this for our machine learning projects. [*S. l.*]: Towards Data Science Inc., 17 dez. 2021. Disponível em: https://towardsdatascience.com/extracting-song-data-from-the-spotify-api-using-python-b1e79388d50. Acesso em: 24 jun. 2022.