

**Insper: Instituto de Ensino e Pesquisa**

Ellen Shen

Enzo Neto

Gabriel Huerta

Giulia Castro   
   
   
   
 

**CIÊNCIA DOS DADOS**

 Projeto 3: Análise de dados do Spotify  
   
   
   
 

**São Paulo**

**2019**

**Introdução:**

Falar da pergunta

Falar de como baixou a base de dados

Falar dos 3 parâmetros escolhidos

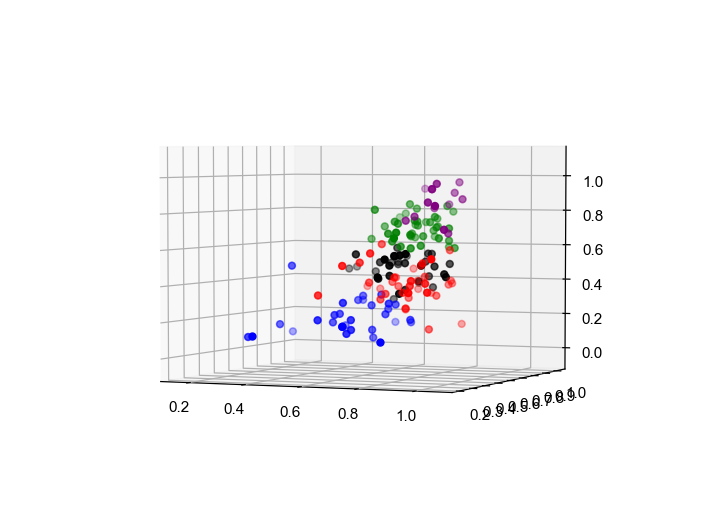
**Clusterização:**

Para a técnica de clusterização inicialmente foram importadas as seguintes bibliotecas: matplotlib.pyplot, pandas, numpy, mpl\_toolkits.mplot3d, seaborn e sklearn. Logo após foram importadas as bases de dados e divididas entre a base que seria utilizada para o treinamento – chamada de top\_2017\_2018\_treinamento – e a base que seria utilizada para o teste do código criado – chamada de top\_2017\_2018\_teste.

Depois desse processo inicial, começamos uma análise exploratória básica dos dados. Desse modo, foi plotado um scatterplot com as três variáveis escolhidas para o nosso estudo, como mencionado na introdução, para a visualização da distribuição das músicas que compunham a base de dados no espaço.

Através do comando *cluster.MiniBatchKMeans* a equipe definiu por trabalhar com 5 grupos distintos, e logo após foi realizado um *fit* passando como argumento as colunas das variáveis que foram definidas anteriormente (danceability, energy e valence). Utilizando o comando *.predict*, o grupo obteve a separação das músicas nos diferentes grupos pré-estabelecidos. O resultado foi guardado na variável *saida\_2017\_2018* e copiados para construir um novo dataset no qual possuía a coluna de saída com os seus respectivos clusters, o qual foi chamado de: *musicas\_2017\_2018*.

Após esse processo foi plotado um outro gráfico de dispersão que separava as músicas pertencentes a cada grupo por cores diferentes. Ou seja, nesse novo gráfico obtivemos a mesma distribuição das músicas no espaço, porém com pontos de cores diversas onde o Grupo 1 foi representado pela cor vermelha, o Grupo 2 pela cor verde, o Grupo 3 pelo azul, Grupo 4 pelo preto e o Grupo 5 pelo roxo. Chegando ao resultado representado na Figura 1 abaixo.



**Figura 1** – Gráfico de dispersão das músicas separadas nos grupos

Apesar do gráfico possibilitar uma ótima observação da distribuição dos Grupos, ele não possibilita uma precisão nos valores utilizados para a construção da divisão. Tendo isso em vista, foi realizado um Pairplot para cada grupo para entendermos a lógica utilizada pelo computador para fazer a divisão. Sendo assim foi concluído que:

**- Grupo 1:** Apresenta dançabilidade alta, porém a valência e a energia médias. Portanto, são músicas dançantes, mas sem muito barulho e não são nem muito alegres nem tristes.

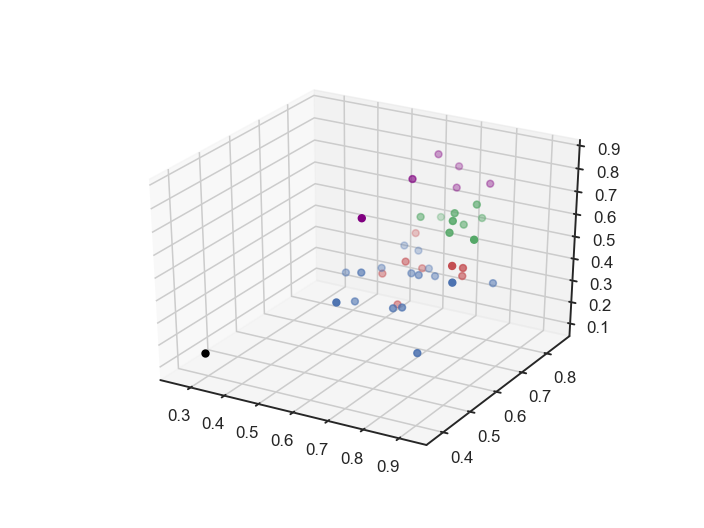
**- Grupo 2:** Possui as três variáveis com valores altos. Conclui-se que são músicas dançantes, barulhentas e alegres.

**- Grupo 3:** Pode-se observar que esse grupo apresenta as músicas com valência muito baixa. Assim, conclui-se que as músicas desse grupo são negativas e mais deprimentes.

**- Grupo 4:** Como pode ser visto as músicas pertencentes a esse grupo apresentam dançabilidade e energia altas e valores de valência medianos. Conclui-se, então, que as músicas são dançáveis, bem rápidas e barulhentas, porém não são tão positivas.

**- Grupo 5:** Nota-se pelo gráfico de dispersão que essas músicas apresentam valência, energia e dançabilidade altos. Ou seja, as músicas desse grupo são no geral bem dançantes, bem energéticas e muito positivas.

Com a conclusão dos resultados obtidos, começamos a trabalhar com a base de dados do teste, repetindo os mesmos processos feitos n treinamento. Ou seja, inicialmente plotamos um gráfico de dispersão das músicas para a visualização dos dados; depois definimos os 5 grupos com os mesmos comandos utilizados anteriormente; e em seguida foi plotado o gráfico final de dispersão com a mesma divisão de cores. Chegando ao resultado representado na Figura 2.



**Figura 2** **-** Gráfico de dispersão das músicas de teste separadas nos grupos

Basta uma análise rápida do gráfico para perceber que os grupos definidos pelo código não correspondem com os Grupos definidos anteriormente; o Grupo 4 (preto), por exemplo, era um Grupo com músicas dançáveis, rápidas e barulhentas, apesar de não serem tão positivas, porém nessa nova distribuição esse mesmo Grupo contém apenas 1 música que é negativa, não dançável e bem pouco energética.

Conclui-se que para essa técnica ser viável seria necessário possuir um gabarito.

**Decision Tree**

Para implementar esta técnica foram utilizadas as bibliotecas do python pandas e sklearn (incluindo as sub-bibliotecas preprocessing e model\_selection). O primeiro passo para fazer a árvore foi coletar a base de dados do spotify: através da conta de desenvolvedor foram obtidas as características de áudio de 100 músicas pertencentes aos top charts de 2018 e 50 outras do “top songs” de outubro de 2019 do próprio aplicativo. É importante apontar que, por motivos práticos, o csv foi manipulado manualmente para retirar algumas variáveis que se julgaram pouco significativas e juntar as duas bases de dados.

A seguir, para poder existir uma variável de input, as 150 músicas foram classificadas em três categorias cada uma representando a preferência de um membro do grupo: músicas de que Enzo gostava recebiam 0; Ellen, 1 e Gabriel, 2.

O programa consistiria, então, em um ‘split’ de treino e teste com um ‘test size’ pequeno para conseguir mais casos para treino; e, em seguida o código principal do classificador da árvore e seu ‘fit’. No entanto, ao checar o score do teste, obteve-se um valor baixo; próximo de 30%.

Por acaso, ocorreu um erro no código e ao tentar consertá-lo foi retirada a variável “instrumentalness”, que mede o quão instrumental é a música. Sem essa variável o teste chegou a obter um acerto de 53%.

Foi possível concluir que a variável estava afetando significativamente a performance do classificador e pôde-se entender a importância de escolher bem as variáveis utilizadas.

Adicionalmente, como um teste extra, foram pegas três músicas; diretamente das playlists de cada um dos três membros da classificação, e suas características foram testadas com a função ‘predict’ da decision tree e, para a surpresa e alegria do grupo todas as três músicas foram classificadas corretamente como sendo de seus respectivos “donos”.

Em conclusão, a técnica da Decision Tree apresentou resultados decentes, mostrando-se viável. Ainda mais se, em futuras iterações fossem estudadas as diferentes variáveis e suas influências mais minuciosamente.

**Naive Bayes**

Para a técnica de naive bayes inicialmente foram importadas as seguintes bibliotecas: matplotlib.pyplot, pandas, numpy e sklearn. Logo após foram importadas a base de dados e dividida entre as variaveis x e y, que servirao como parametros mais tarde. A variavel x e um loc das tres colunas de dados que iremos usar, já a variavel y e a coluna de saida com as classificacoes anteriormente feitas pelo metodo de clusterizacao.

Entre as duas variaveis dividimos a parte que seria utilizada para o treinamento – chamada de x\_treinamento – e a parte que seria utilizada para o teste do código criado – chamada de x\_teste e o mesmo foi feito para a variavel y.

Utilizando o comando MultinomialNB passamos os parametros x e y para fazer um fit, logo apos isso fizemos um predict utilizando a parte teste, tendo assim uma classificacao feita pelo codigo.

Para entender melhor como a classificacao foi feita, fizemos um value counts da coluna saida da base de treinamento, para verificar a proporcao de cada grupo, assim percebemos que os grupos com a maior contagem eram mais suscetiveis a serem classificados na base de testes.

Com os proprios comandos de acuracia e precisao do sklearn passamos o teste com as classificacoes de saida como parametros. Obtivemos uma acuracia de 46% e uma precisao de 23%. Chegando à conclusão de que esta tecnica não é confiavel pois tem um baixo indice de acerto. Para uma futura iteração, faria sentido usar mais variáveis no classificador.