Recomendador de Músicas com dataset do Spotify

Gabriel Meale

[1]Universidade Federal do ABC (UFABC)

Abstract. *This project employs the use of a machine learning algorithm (KNN) to perform personalized music recommendation for a user. Based on the initial Spotify music database, the user will provide an initial marking of their musical preferences and, after appropriate data processing, will receive recommendations for some songs. There is the possibility for the user to provide feedback for model retraining and improvement.*

Resumo. *Este projeto emprega a utilização de um algoritmo de aprendizado de máquina (KNN) para realizar recomendação personalizada de músicas para um usuário. A partir da base inicial de músicas do Spotify, o usuário fará uma marcação inicial de seu gosto musical e, após o tratamento de dados devido, receberá indicação de algumas músicas. Existe a possibilidade do usuário dar feedback para re-treino e aperfeiçoamento do modelo.*

1. Introdução

Com o crescimento exponencial da quantidade de música disponível nas plataformas de streaming, como o Spotify, surge a necessidade de aprimorar os sistemas de recomendação existentes. Embora esses sistemas tenham revolucionado a forma como descobrimos novas músicas, muitas vezes enfrentamos desafios significativos ao tentar encontrar faixas que verdadeiramente correspondam aos nossos gostos individuais e estados de espírito em constante evolução.

Os métodos tradicionais de recomendação musical, muitas vezes baseados apenas em algoritmos de filtragem colaborativa ou conteúdo, podem não capturar totalmente a complexidade e a singularidade dos gostos musicais de cada usuário. Essas abordagens podem resultar em recomendações genéricas, que não refletem com precisão a diversidade e a profundidade de preferências musicais de um indivíduo. Ou ainda, recomendar sempre músicas que o usuário já conhece e não o expondo a novas descobertas.

Portanto, esse projeto é um exercício para entender a recomendação musical e melhorá-la, com técnicas de aprendizado de máquina, como o algoritmo KNN (K-Nearest Neighbors), combinadas com a capacidade de incorporar feedback contínuo do usuário. Essa abordagem visa oferecer recomendações mais personalizadas e relevantes, adaptadas às preferências musicais específicas de cada usuário e capazes de se ajustar dinamicamente às mudanças em seus gostos ao longo do tempo.

2. Metodologia

O projeto consiste em carregar uma base inicial de dados (Spotify Tracks Dataset), realizar um tratamento de dados inicial, normalizar os valores numéricos, tratar as variáveis categóricas e rodar um modelo de aprendizado de máquina. A detalhamento da base pode ser encontrado no seguinte link: `https://www.kaggle.com/datasets/maharshipandya/-spotify-tracks-dataset`.

2.1. Seleção de Features

As seguintes features foram selecionadas para o treinamento, pois carregam dados que podem ser relevantes para a decisão consciente (ou inconsciente) de escolher da música que um usuário quer ouvir.

- artists - Os nomes dos artistas que executaram a faixa
- album_name - O nome do álbum em que a faixa aparece
- track_name - Nome da faixa
- popularity - A popularidade da faixa (valor entre 0 e 100)
- duration_ms - O comprimento da faixa em milissegundos
- danceability - Descreve a adequação da faixa para dançar (valor entre 0.0 e 1.0)
- energy - Medida de intensidade e atividade da faixa (valor entre 0.0 e 1.0)
- key - A tonalidade da faixa
- loudness - A sonoridade geral da faixa em decibéis (dB)
- mode - Indica a modalidade (maior ou menor) da faixa
- speechiness - Detecta a presença de palavras faladas na faixa
- acousticness - Medida de confiança de se a faixa é acústica
- instrumentalness - Prevê se a faixa contém vocais
- liveness - Detecta a presença de uma audiência na gravação
- valence - Descreve a positividade musical transmitida pela faixa
- tempo - O tempo estimado geral da faixa em BPM
- time_signature - Uma assinatura (compasso musical) de tempo estimada
- track_genre - O gênero ao qual a faixa pertence

2.2. Pré-processamento

O primeiro passo do pré-processamento foi dropar os valores nulos da base, que são poucos e, por esse motivo, não afetaria o desempenho do algoritmo caso abordado dessa forma.

Após isso, é feito um filtro de popularidade, pois existem muitos registros na base e, para fazer uma validação de conceitos, podería-se restringir a uma menor variedade de artistas.

2.3. Marcação de dados

Em um sistema real, é necessário que existam APIs e bancos de dados que registrem todos os dados relevantes de um usuário para que seja possível, com esses dados, gerar a melhor indicação musical possível. A oferta inicial de música ao usuário da plataforma, seria feita perguntando seus gostos principais (artistas, músicas e gênero) que mais lhe agradam, através de uma interface agradável e fluida.

Para simular esse comportamento, é feita manualmente uma marcação simples na base, ilustrada pelo pseudo-código do Algorithm 1. Também foi feito o teste marcando diretamente artistas (ao invés de músicas), o que gera mais registros de treinamento, mas pode acabar deixando o modelo muito específico (exploraremos mais na sessão Conclusão).

Algorithm 1 User training

Require: Dataset with tracks and their attributes
Ensure: Target variable y_{target} indicating user's preference for each track
User trains by choosing his favorite artist (or songs)
for each track in dataset **do**
if track == "Like a Rolling Stone" **then**
$y_{\text{target}} \leftarrow 1$
end if
if track == "É Proibido Fumar - Versão remasterizada" **then**
$y_{\text{target}} \leftarrow 1$
end if
if track == "Angie" **then**
$y_{\text{target}} \leftarrow 1$
end if
if track == "Martha My Dear - Remastered 2009" **then**
$y_{\text{target}} \leftarrow 1$
end if
if track == "I Love It Loud" **then**
$y_{\text{target}} \leftarrow 1$
end if
if track == "La Belle de Jour" **then**
$y_{\text{target}} \leftarrow 1$
end if
if track == "O Descobridor dos Sete Mares" **then**
$y_{\text{target}} \leftarrow 1$
end if
if track == "Jump - 2015 Remaster" **then**
$y_{\text{target}} \leftarrow 1$
end if
if track == "Have You Ever Seen The Rain" **then**
$y_{\text{target}} \leftarrow 1$
end if
if track == "White Room" **then**
$y_{\text{target}} \leftarrow 1$
end if
end for
for each track in dataset **do**
if $y_{\text{target}} == -1$ **then**
$y_{\text{target}} \leftarrow 0$
end if
end for

2.4. Normalização dos dados

Muitos algoritmos de aprendizado de máquina são sensíveis à escala dos dados. Normalizar as características ajuda a garantir que os algoritmos funcionem de forma mais eficaz e eficiente, especialmente quando as características têm escalas muito diferentes. Isso pode levar a um melhor desempenho do modelo e a uma convergência mais rápida durante o treinamento.

Para realizar a normalização dos dados, utilizamos a normalização z-score. Ela transforma as características de forma que tenham média zero e variância unitária. Isso significa que após a transformação, a média de cada característica será 0 e o desvio padrão será 1.

A normalização foi aplicada nas seguintes features:

- popularity
- duration_ms
- danceability
- energy - Medida de intensidade e atividade da faixa (valor entre 0.0 e 1.0)
- loudness - A sonoridade geral da faixa em decibéis (dB)
- speechiness - Detecta a presença de palavras faladas na faixa
- acousticness - Medida de confiança de se a faixa é acústica
- instrumentalness - Prevê se a faixa contém vocais
- liveness - Detecta a presença de uma audiência na gravação
- valence - Descreve a positividade musical transmitida pela faixa
- tempo - O tempo estimado geral da faixa em BPM

2.5. Balanceamento

Já que temos poucos registros positivos e muito mais exemplos ”negativos”, se faz necessário balancear o conjunto de treinamento para que não ocorra overfitting, como mostra o Algorithm 2. Dessa forma teremos a mesma quantidade de dados marcados como True e False.

Algorithm 2 Seleção de Exemplos Positivos e Negativos

1: $true\_examples \leftarrow$ dataset[dataset[”y_target”] $== 1$]
2: $false\_examples \leftarrow$ dataset[dataset[”y_target”] $== 0$]
3: $false\_examples\_sampled \leftarrow$ false_examples.sample($n =$ true_examples.shape[0])

2.6. Get_Dummies

Já que muitos algoritmos de aprendizado de máquina não podem lidar diretamente com variáveis categóricas, vamos transformá-las em valores numéricos. O one-hot encoding (implementado pelo pandas como get_dummies é uma técnica comum para fazer isso, pois preserva a natureza categórica das variáveis enquanto as transforma em um formato que os algoritmos podem entender e processar adequadamente. Isso ajuda a melhorar o desempenho e a precisão dos modelos.

2.7. Separação para Treino e Teste

Nesse projeto tivemos duas abordagens. A primeira é simplesmente separar uma porcentagem da base para treino e outra para teste. Foi aplicada quando o usuário marcou seu gosto musical através de nomes de artistas, o que gerou mais registros para treinamento. Já para a marcação de apenas 10 músicas, foi utilizado o método Leave-One-Out (LOO).

No LOO, o modelo é treinado usando o conjunto de dados menos uma observação. Isso significa que o modelo é treinado repetidamente em subconjuntos de dados ligeiramente diferentes, cada um contendo todos os dados, exceto uma observação.

Após treinar o modelo em cada conjunto de dados menos um, o modelo é testado na observação que foi deixada de fora.

2.8. KNN

Foi escolhido o algoritmo KNN para essa aplicação, com valores de K=11.

2.9. Avaliação do modelo

Para validação da assertividade do modelo, utilizou-se as métricas de acurácia, recall e f1-score.

3. Resultados e Observações

Após a implementação do algoritmo obtivemos os seguintes resultados.

3.1. Marcação através de artista

Número de registros:

- Treino - 507
- Teste - 12

Table 1. Resultados do Modelo

Classe	Precisão	Recall	F1-Score
0	0.93	0.69	0.79
1	0.77	0.95	0.85

Podemos observar através de tabela 1 que o algoritmo teve uma precisão bem alta, principalmente quando se trata dos exemplos negativos. Porém ele sabe distinguir muito melhor o que o usuário quer ouvir do que o que não quer. De certa forma, o modelo está enviesado. Essa abordagem trará novas sugestões de músicas apenas quando o modelo errar uma predição negativa e sempre indicará músicas que, provavelmente, o usuário já conhece.

3.2. Marcação através de música

Vale relembrar que nessa abordagem foi utilizado o método LOO, com uma base inicial de 44 registros.

Nessa abordagem, de acordo com a 2, é possível perceber que o algoritmo erra mais. Mas o recall está muito mais próximo do que no resultado por artista. Isso de certa forma indica um algoritmo mais equilibrado que pode ter a chance de oferecer ao usuário músicas diferentes.

Table 2. Resultados do Modelo

Classe	Precisão	Recall	F1-Score
0	0.62	0.59	0.60
1	0.61	0.64	0.62

3.3. Conclusão

Independente das duas abordagens, se faz necessário equilibrar os registros de marcação do usuário de alguma forma. O primeiro contato do usuário com o aplicativo ou plataforma, deve favorecer a criação desses dados. Após o uso do aplicativo essa base deve ser retroalimentada com as novas marcações que o usuário fará, ativamente ou de maneira passiva. Por exemplo, o aplicativo deve ter uma opção de um usuário marcar uma sugestão ruim como negativa e uma sugestão que o agrada como positiva. Dessa forma o algoritmo será retroalimentado e treinado novamente.

Porém, existe um ponto muito curioso extraído dessa análise. A tendência é que a assertividade geral do algoritmo aumente na medida que o usuário continue retroalimentando-o (é possível extrair uma análise muito semelhante através dos resultados de marcações por artista que cria uma base de dados maior para treino). Isso acabaria gerando um algoritmo de ”bolha” que deixaria de fazer recomendações fora dessa ”bolha”. Então, de certa forma, parece que é necessário que o algoritmo esteja balanceado de uma forma que ele não busque a perfeição.

Além disso, é interessante pensar que a seleção de features mais simples, como ”Artists”, tendem a enviesar e deixar o algoritmo ainda mais com essa caracteristica.

4. Trabalhos Futuros

- Exploração de outros algoritmos de aprendizado de máquina
- Feature Selection diferente, desconsiderando caracteristica que tendem a enviesar o algoritmo
- Testes de retroalimentação

References

Medeiros, Débora. (Q1.2024). Aulas de Mineração de Dados. Curso de Bacharelado em Ciência da Computação, Universidade Federal do ABC.