Universidade Presbiteriana Mackenzie

PROJETO DE ANÁLISE DE DADOS

DE

MÚSICA NO SPOTIFY

ALEX JUNIOR MOURA DA SILVA

ENZO VEMADO

JOAO PEDRO BARRETO DE MELO

LUCIANO GUIMARAES COSTA

Trabalho apresentado como critério de avaliação da disciplina

PROJETO APLICADO III

São Paulo

2024

Universidade Presbiteriana Mackenzie

SUMÁRIO

1. INTRODUÇÃO

1.1. CONTEXTO DO TRABALHO

A música, além de ser uma expressão cultural, é uma indústria em constante evolução, influenciada por tendências sociais, avanços tecnológicos e preferências de consumo. Com o avanço das plataformas de streaming, como o Spotify, houve uma mudança significativa na forma como a música é consumida e distribuída. Essas plataformas não apenas ampliaram o acesso à música, mas também geraram uma grande quantidade de dados que podem ser explorados para entender melhor os padrões dos usuários.

1.2. MOTIVAÇÃO E JUSTIFICATIVA

A relevância do tema se manifesta na sua capacidade de fornecer insights sobre o comportamento dos consumidores e o impacto das tendências musicais em escala global. A escolha do tema foi motivada por interesses pessoais e profissionais em análise de dados e na música. A análise desses dados pode revelar padrões comportamentais e preferências.

Desenvolver este projeto oferece diversas vantagens e benefícios. Primeiramente, ele contribuirá para aumentar o conhecimento sobre o comportamento. Em segundo lugar, o projeto pode contribuir para a aplicação de técnicas de análise de dados em um contexto prático, permitindo a aplicação de novos algoritmos e abordagens analíticas.

Esse desenvolvimento poderá contribuir significativamente para o campo de estudo ao oferecer uma visão detalhada do comportamento de consumo de música e propor modelos preditivos robustos baseados em dados acústicos e demográficos.

1.3. OBJETIVOS

1.3.1. OBJETIVO GERAL:

Analisar e modelar dados musicais do Spotify, explorando características acústicas e padrões de consumo para identificar tendências e propor um sistema de recomendação personalizado.

1.3.2. OBJETIVOS ESPECÍFICOS:

1. Analisar as características acústicas das faixas, como dançabilidade, energia, valência e instrumentalidade.
2. Avaliar o impacto de variáveis como popularidade, ano de lançamento e letras explícitas no consumo musical.
3. Desenvolver modelos preditivos para recomendar músicas com base no perfil de consumo dos usuários.
4. Explorar o comportamento de diferentes grupos demográficos em relação ao consumo de músicas específicas.

2.0. DEFINIÇÕES

2.1. BIBLIOTECAS PYTHON:

Utilizamos as seguintes bibliotecas:

- **PySpark**: Para manipulação e processamento de grandes volumes de dados de forma distribuída, essencial para tratar a base de dados do Spotify em larga escala.
- **Plotly**: Para criar visualizações interativas que ajudarão na análise exploratória e na apresentação dos resultados.
- **NumPy** e **SciPy**: Para operações matemáticas e estatísticas avançadas, facilitando cálculos complexos e análises quantitativas.
- **Spotipy**: Para interação com a API do Spotify, possibilitando a coleta e manipulação de dados diretamente da plataforma.
- **Matplotlib**: Para gerar gráficos estáticos que auxiliam na análise exploratória dos dados.
- **Scikit-Image (skimage)**: Pode ser utilizada para processamento de imagens caso você decida explorar capas de álbuns ou outras imagens relacionadas às músicas.

2.2 BASE DE DADOS:

2.2.1. DESCRIÇÃO DA BASE DE DADOS:

A base de dados utilizada para este projeto foi extraída da API pública do Spotify, conforme descrito na documentação oficial da plataforma. A base abrange uma série de variáveis que descrevem tanto as faixas musicais quanto o comportamento dos ouvintes. Entre as principais variáveis estão:

- **Acousticness (Acústica):** Mede a confiança de que a faixa é acústica, em uma escala de 0,0 a 1,0.
- **Danceability (Dançabilidade):** Avalia quão adequada uma faixa é para dançar, com valores entre 0,0 e 1,0.
- **Energy (Energia):** Mede a intensidade da faixa, em uma escala de 0,0 a 1,0.
- **Valence (Valência):** Indica a positividade emocional de uma faixa, variando de 0,0 a 1,0.
- **Instrumentalness (Instrumentalidade):** Prediz a ausência de vocais em uma faixa.
- **Tempo:** Refere-se ao ritmo da música em batidas por minuto (BPM).

- **Loudness (Volume em dB):** Representa o volume médio da faixa em decibéis.
- **Popularity (Popularidade):** Avalia a popularidade da música, com valores entre 0 e 100.
- **Explicit (Explícito):** Indica se a faixa contém ou não letras explícitas.
- **Key (Chave):** Representa a tonalidade da faixa.
- **Mode (Modo):** Define se a música está em escala maior (1) ou menor (0).

2.2.2. LIMITAÇÕES DA BASE DE DADOS:

Podemos ter algumas limitações na base, por exemplo as métricas de popularidade podem variar de acordo com o número de reproduções recentes, o que pode introduzir viés temporal. Além disso, a previsão de atributos como instrumentalidade ou vivacidade pode não ser precisa em todos os casos, especialmente para faixas com características ambíguas.

2.2.3. ORIGEM DOS DADOS:

Os dados foram extraídos diretamente da API do Spotify e disponibilizados em formato como tabela no Github (de onde consultamos eles). Cada faixa é identificada por um ID único fornecido pela plataforma, permitindo o rastreamento e a análise de todas as suas características associadas.

3.0 LEITURA E ANALISE DOS DADOS:

3.1 LEITURA DOS DADOS:

Nesta etapa, buscamos importar a base de dados do Spotify, acessando tanto os dados previamente baixados quanto os dados obtidos por meio da API oficial da plataforma, utilizando a biblioteca Spotipy para integração.

Durante a leitura, foi necessário realizar verificações de integridade e qualidade dos dados, como a identificação de valores ausentes, inconsistências e possíveis duplicatas. A combinação das bibliotecas PySpark e Pandas foi essencial para processar eficientemente a grande quantidade de informações, permitindo o carregamento e a manipulação dos dados de forma rápida e eficaz.

```
dados.printSchema()
```

```
root
 |-- valence: double (nullable = true)
 |-- year: integer (nullable = true)
 |-- acousticness: double (nullable = true)
 |-- artists: string (nullable = true)
 |-- danceability: double (nullable = true)
 |-- duration_ms: integer (nullable = true)
 |-- energy: double (nullable = true)
 |-- explicit: integer (nullable = true)
 |-- id: string (nullable = true)
 |-- instrumentalness: double (nullable = true)
 |-- key: integer (nullable = true)
 |-- liveness: double (nullable = true)
 |-- loudness: double (nullable = true)
 |-- mode: integer (nullable = true)
 |-- name: string (nullable = true)
 |-- popularity: integer (nullable = true)
 |-- speechiness: double (nullable = true)
 |-- tempo: double (nullable = true)
 |-- artists_song: string (nullable = true)
```

3.2 ANALISE EXPLORATORIA

O processo de análise exploratória tem como objetivo identificar padrões, relações e possíveis inconsistências dentro do conjunto de dados. A partir dessa análise, será possível extrair insights preliminares sobre as características das músicas, como dançabilidade, energia, valência e popularidade, além de identificar variáveis que podem ser relevantes para o desenvolvimento do modelo de recomendação.

Foi realizada a leitura do arquivo CSV contendo dados sobre músicas utilizando a biblioteca PySpark. Inicialmente, o arquivo foi adicionado ao contexto do Spark a partir de uma URL externa, hospedada no GitHub. O caminho do arquivo foi recuperado através da função SparkFiles.get, e em seguida, a leitura do arquivo foi feita com o método read.csv, configurando-se para considerar o cabeçalho e inferir automaticamente o esquema dos dados. Essa abordagem permite o processamento eficiente do conjunto de dados em grandes volumes, utilizando o poder de processamento distribuído do PySpark.

```
url_anos_dados = 'https://github.com/IgorNascAlves/dados/blob/main/dados_musicas_ano.csv?raw=true'

sessao_spark.sparkContext.addFile(url_anos_dados)
path_dados_file = 'file://' + SparkFiles.get('dados_musicas_ano.csv')

dados_anos = sessao_spark.read.csv(path_dados_file, header=True, inferSchema=True)
```

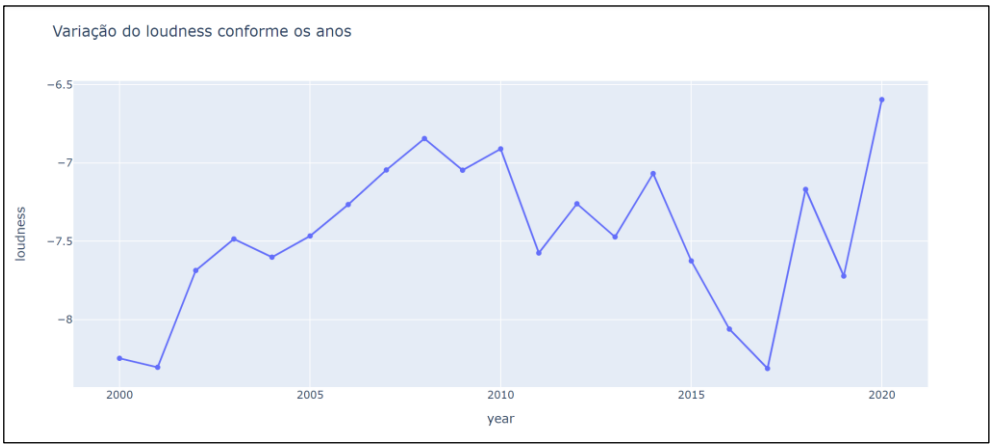
Com isso, realizamos a leitura e filtragem dos dados, utilizando PySpark, para considerar apenas as músicas lançadas a partir do ano 2000. A função filter foi aplicada ao conjunto de dados, restringindo a análise a esse período específico. Em seguida, os dados filtrados foram exibidos por meio do método show(), apresentando as primeiras 20 linhas, com variáveis como *acousticness* (acústica), *danceability* (dançabilidade), *energy* (energia), e outras características musicais. Esse passo é essencial para refinar o escopo da análise e focar nas faixas mais recentes.

```
dados_anos = dados_anos.filter('year >= 2000')
dados_anos.show()
```

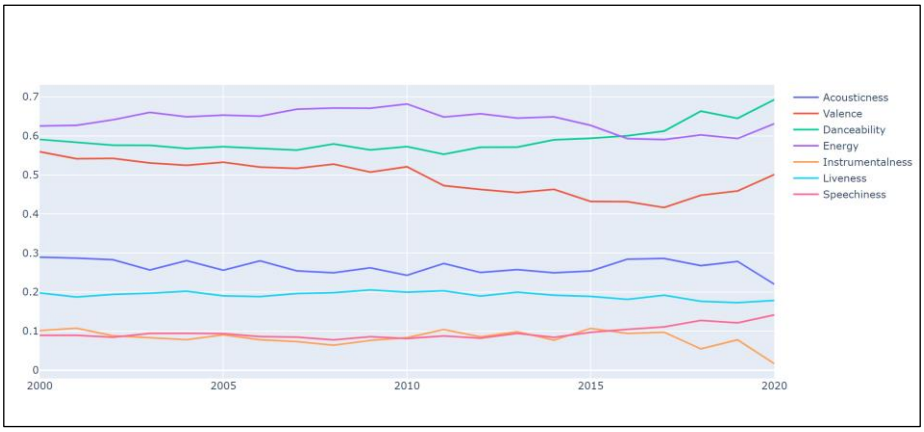
mode	year	acousticness	danceability	duration_ms	energy	instrumentalness	liveness	loudness	speechiness	tempo	valence	popularity	key
1	2000	0.28932270051635994	0.590918047034764	242724.6426380368	0.6254128323108387	0.10116776879345596	0.1976860429447853	-8.247765848670758	0.0892054192290394	118.9993231083843	0.5594754601226991	46.6840490797546	7
1	2001	0.2868424784828934	0.5833178553615969	240307.79600997505	0.6269855221945144	0.10721401899251867	0.18702563591022486	-8.305095261845384	0.0891822942643916	117.765399002493781	0.5414791072231919	48.750124688279291	7
1	2002	0.28262428988500105	0.5761602999999997	239503.283	0.6412697960000012	0.08804835031499995	0.1939111999999998	-7.686639500000013	0.08430759999999995	119.239738499999931	0.5423971500000008	48.6555	7
1	2003	0.25647051817297833	0.5757633060388944	244670.57523029681	0.6601652610030712	0.08304927466734914	0.19697630501535301	-7.485545035823955	0.0939257420675538	120.914621801432811	0.5305042476970324	48.626407369498471	7
1	2004	0.2885589466225845	0.5676803662258397	237378.78083662261	0.6488679450661226	0.07793403490884357	0.2021994040883014	-7.601555137334684	0.0942388097660224	121.290345879959961	0.5244885554425227	49.271343438453721	7
1	2005	0.25576350686666651	0.5722805641025652	237229.5882051282	0.6532085112820511	0.09019436118461543	0.19008162564102585	-7.46615897435897	0.0933336923076923	121.617966666666449	0.5325309230769236	50.953333333333331	0
1	2006	0.2799863502564088	0.5682301538461539	234042.91435897435	0.6503262820512825	0.07770147187692314	0.1882892820512821	-7.265500512820514	0.0858467692307692	121.798615384615361	0.5200283076923078	51.313846153846161	7
1	2007	0.2540809575538463	0.5634143589743592	241049.962564102551	0.6683047743589737	0.07295726805128197	0.1961265641025642	-7.044535897435892	0.0843473333333336	124.08751641025621	0.5167938461538473	51.075897435897441	7
1	2008	0.2491917627212275	0.5791928388746803	240107.31560102306	0.6714680207672623	0.06366209031713561	0.19843140664961625	-6.843804092071661	0.0773563682864507	123.509933508336411	0.5275418925831203	50.630179028132991	0
1	2009	0.26192861909693904	0.5641903571428577	238140.0132653061	0.6707487551020408	0.07587207368367352	0.2052522959183674	-7.046014795918364	0.0854578061224899	123.463807653061251	0.5071696617346935	51.440816326530616	0
1	2010	0.2426866435515872	0.5724883432539687	242811.8045634921	0.6817778026289684	0.08298056821924611	0.19970064484126995	-6.909904265873012	0.0810314980158731	123.570215777779	0.5208951587301589	52.730158730158731	0
1	2011	0.27318266251859197	0.5528669806643526	236998.787307883	0.6483008934064099	0.10377233216658402	0.20330882498760566	-7.574986117997006	0.0874792761270201	121.483997025284881	0.4724536787307882	53.307387208725821	2
1	2012	0.2499530443804624	0.5708818588097431	245807.4575835476	0.6565714601542408	0.08528552067866319	0.1897330077120824	-7.260549613959888	0.0817424678663405	121.781736246786691	0.4627090128534706	52.655012853470441	7
1	2013	0.2574880859564779	0.5711480263157896	242267.66143724695	0.6455968914473688	0.09836505391700387	0.19963081983805714	-7.472039473684207	0.0938488866396761	120.806829453441181	0.4547411943319834	54.047064777327925	1
1	2014	0.2493126458802906	0.5899476807980057	233728.31471321694	0.6487954437905223	0.07656955483790515	0.1918215960099752	-7.06743990024938	0.0840605486284296	122.305262842892961	0.4630487780548628	55.543142144638395	0
1	2015	0.2539522577102332	0.5937740628166152	230029.0466058764	0.6270642715298895	0.10678678054771141	0.18885643363728474	-7.625639311043569	0.0967792299898684	120.115411347517681	0.4320983687943254	56.700607902735571	7
1	2016	0.28417102995548144	0.6002023928770179	221396.51029493602	0.5928552316082347	0.09398438711741793	0.18116989426822505	-8.061056204785759	0.1043132999443518	118.652629938786931	0.4315320589872012	59.647189760712301	0
1	2017	0.2860990665261043	0.6122170180722886	211115.6967871486	0.5904210208835337	0.09709060906626496	0.1917126004016064	-8.312629518072281	0.1105364959839356	117.202739959839361	0.4164763112449793	63.263554216867471	1
1	2018	0.2676329907750836	0.6635004755111744	206001.0071326676	0.6024346220161672	0.05421712166904419	0.17632553495007114	-7.168785068949124	0.1271755587253022	121.922307655729601	0.4479212743699474	63.296243461721351	1
1	2019	0.27829858633658241	0.6448141097998967	201024.78809645973	0.5932240360184717	0.07764024697280643	0.17261641867624464	-7.722191893278596	0.1218433555669573321	120.235644433042721	0.4588176295536167	65.256541816316061	1

only showing top 20 rows

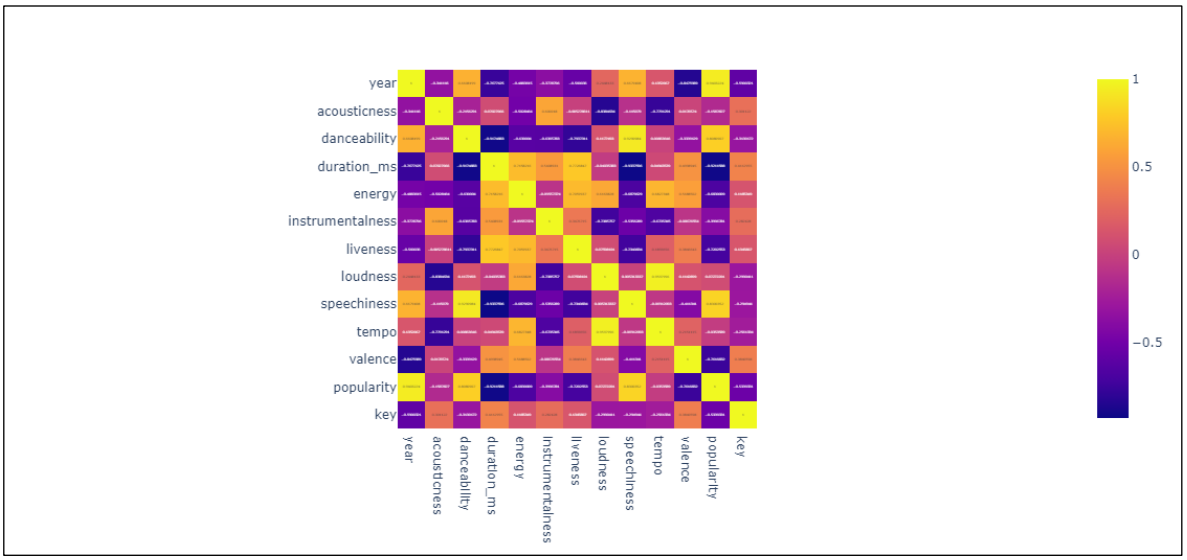
Foi gerado um gráfico que mostra a variação do *loudness* (volume) das músicas ao longo dos anos, utilizando a biblioteca Plotly para visualização interativa dos dados. A linha do gráfico representa a média do volume das músicas em decibéis (dB) para cada ano, desde 2000 até 2020. Observa-se que o *loudness* teve uma variação significativa ao longo do tempo, com períodos de aumento e diminuição. Esse tipo de análise ajuda a identificar tendências de sonoridade nas produções musicais ao longo das duas últimas décadas.



Também geramos um gráfico de linhas comparando a evolução de diversas características musicais entre os anos 2000 e 2020. As variáveis analisadas incluem *acousticness* (acústica), *valence* (positividade), *danceability* (dançabilidade), *energy* (energia), *instrumentalness* (instrumentalidade), *liveness* (vivacidade) e *speechiness* (fala). O gráfico mostra que, ao longo do tempo, algumas dessas variáveis apresentaram pouca variação, como *speechiness* e *liveness*, enquanto outras, como *acousticness* e *valence*, exibem flutuações mais evidentes. Essas informações são importantes para identificar mudanças nas tendências de produção musical ao longo dos anos, refletindo a evolução das preferências musicais e a produção das músicas no mercado global.



Em seguida, foi realizada uma análise de correlação entre as variáveis do conjunto de dados musicais, utilizando um *heatmap*. A matriz de correlação mostra as relações entre diferentes características musicais. As cores variam de -1 a 1, representando correlações negativas e positivas, respectivamente. Essa análise é crucial para identificar interdependências entre as variáveis, permitindo observar, por exemplo, uma correlação positiva entre *loudness* e *energy*, sugerindo que faixas com maior volume tendem a ser mais energéticas.



4.0 TRATAMENTO E PREPARACAO DA BASE

Nesse processo de tratamento, a base de dados do Spotify, é preparada para que possamos obter resultados confiáveis. As atividades principais envolvem a limpeza de dados, que inclui a remoção de duplicatas e o tratamento de valores ausentes, além da normalização de variáveis numéricas e a conversão de dados categóricos. Essas etapas garantem que os dados estejam consistentes e prontos para a aplicação das técnicas de machine learning, permitindo uma análise eficiente e precisa.

Neste trecho de código, após a importação e leitura, a preparação dos dados para o treinamento do modelo de machine learning começa com a remoção da coluna "genres" da lista de variáveis a serem utilizadas no treinamento. A coluna "genres" é categórica, e neste momento, o foco está nas variáveis numéricas, como *acousticness*, *danceability*, *energy*, entre outras. A remoção ocorre para que apenas as características numéricas sejam vetorizadas e, posteriormente, processadas pelo modelo.

```
[ ]  X = dados_generos.columns
     X.remove('genres')
     X

     ['mode',
      'acousticness',
      'danceability',
      'duration_ms',
      'energy',
      'instrumentalness',
      'liveness',
      'loudness',
      'speechiness',
      'tempo',
      'valence',
      'popularity',
      'key']
```

Em seguida, a função VectorAssembler é utilizada para combinar todas as variáveis numéricas em um único vetor, chamado "features". Esse vetor representa um conjunto unificado das características de cada música, que será utilizado pelo modelo. Após essa transformação, o DataFrame é ajustado para conter apenas as colunas "features" e "genres", permitindo que as variáveis categóricas de gênero possam ser mantidas para futuros usos, como na categorização dos dados ou avaliação do modelo.

```
[ ]  dados_generos_vector = VectorAssembler(inputCols=X, outputCol='features').transform(dados_generos).select(['features', 'genres'])

     dados_generos_vector.show(truncate=False, n=5)

     +-----------------------------------------------------------------------------------------------------------------------
     |features
     +-----------------------------------------------------------------------------------------------------------------------
     |[1.0,0.9793333333333332,0.16288333333333335,160297.66666666663,0.07131666666666665,0.60683367,0.3616,-31.514333333333337,0.040566666666666
     |[1.0,0.49478,0.2993333333333333,1048887.333333333,0.4506783333333333,0.4777616666666668,0.131,-16.854,0.07681666666666667,120.285666666666
     |[1.0,0.762,0.7120000000000001,115177.0,0.818,0.8759999999999999,0.126,-9.18,0.047,133.444,0.975,48.0,7.0]
     |[1.0,0.6514170195595453,0.5290925603549332,232880.8902503945,0.4191460727353524,0.2053091895111363,0.21869585415040735,-12.288964675489455
     |[1.0,0.676557304985755,0.5389612464387464,190628.5408867521,0.3164335701566952,0.003003441440420227,0.1722541371082621,-12.479387421652426
     +-----------------------------------------------------------------------------------------------------------------------
     only showing top 5 rows
```

Entao os dados foram normalizados usando a técnica de *Standard Scaling*, que padroniza as variáveis numéricas para terem média 0 e desvio padrão 1. Isso é feito para garantir que todas as variáveis estejam na mesma escala, evitando que alguma variável influencie mais o modelo devido à sua magnitude. O processo é realizado através da função StandardScaler, que transforma a coluna "features" em "scaled_features", onde as variáveis numéricas normalizadas são armazenadas. Após essa transformação, o DataFrame mostra as colunas "features" originais, os "genres" e as "scaled_features", prontas para o uso no treinamento do modelo de machine learning.

```
[ ] scaler = StandardScaler(inputCol='features', outputCol='scaled_features')
    scaler_model = scaler.fit(dados_generos_vector)
    dados_generos_scaler = scaler_model.transform(dados_generos_vector)

    dados_generos_scaler.show()

+--------------------+--------------------+--------------------+
|            features|              genres|     scaled_features|
+--------------------+--------------------+--------------------+
|[1.0,0.9793333333...|21st century clas...|[2.68174831000279...|
|[1.0,0.49478,0.29...|               432hz|[2.68174831000279...|
|[1.0,0.762,0.7120...|              8-bit|[2.68174831000279...|
|[1.0,0.6514170195...|                  []|[2.68174831000279...|
|[1.0,0.6765573049...|         a cappella|[2.68174831000279...|
|[1.0,0.45921,0.51...|            abstract|[2.68174831000279...|
|[1.0,0.3421466666...|      abstract beats|[2.68174831000279...|
|[1.0,0.2438540633...|    abstract hip hop|[2.68174831000279...|
|[0.0,0.3229999999...|           accordeon|[0.0,1.0101313736...|
|[1.0,0.446125,0.6...|           accordion|[2.68174831000279...|
|[0.0,0.0679505384...|          acid house|[0.0,0.2125045534...|
|[1.0,0.2569145079...|            acid rock|[2.68174831000279...|
|[1.0,0.00683,0.66...|         acid trance|[2.68174831000279...|
|[1.0,0.9170191062...|          acousmatic|[2.68174831000279...|
|[1.0,0.7617235394...|      acoustic blues|[2.68174831000279...|
|[1.0,0.4902350260...|        acoustic pop|[2.68174831000279...|
|[1.0,0.4049000220...|       acoustic punk|[2.68174831000279...|
|[1.0,0.6132007936...|       acoustic rock|[2.68174831000279...|
|[0.0,0.229,0.412,...|         action rock|[0.0,0.7161612525...|
|[1.0,0.4328571428...|           adoracion|[2.68174831000279...|
+--------------------+--------------------+--------------------+
only showing top 20 rows
```

Foi utilizada a técnica de Análise de Componentes Principais (PCA), que reduz a dimensionalidade dos dados ao encontrar as combinações lineares mais importantes das variáveis originais. Neste caso, a função PCA foi aplicada sobre as variáveis normalizadas na coluna "scaled_features", com a intenção de reduzir os dados para dois componentes principais (k=2), que são armazenados na nova coluna "pca_features". Após o ajuste do modelo com o método fit, a transformação é realizada e os dados são exibidos com suas novas representações reduzidas. O PCA ajuda a simplificar a complexidade dos dados, mantendo o máximo de variância explicada com um número reduzido de variáveis, o que facilita o treinamento do modelo de machine learning.

```
[ ] from pyspark.ml.feature import PCA

[ ] pca = PCA(k=2, inputCol='scaled_features', outputCol='pca_features')
    model_pca = pca.fit(dados_generos_scaler)
    dados_generos_pca = model_pca.transform(dados_generos_scaler)

[ ] dados_generos_pca.select('pca_features').show(truncate=False)

+----------------------------------------+
|pca_features                            |
+----------------------------------------+
|[2.5070953668885667,0.43816913737697943]|
|[-0.5969679056633488,4.981612052751348] |
|[-4.158460276223561,-0.8366525081079943]|
|[-2.387344878512217,-0.4877989015663405]|
|[-2.6501218371679083,-0.5756819768820474]|
|[-1.49650912033676 3,1.8644183183717797] |
|[-3.923520772157324,0.2851835002352836] |
|[-4.611011109831114,-0.6783790472312378]|
|[-2.837690063084229,-0.5712993716580518]|
|[-2.706690139892783,-1.25937880797083]  |
|[-4.6983313839242875,1.2765569680619446]|
|[-3.375987496679868,0.7560741064307471] |
|[-5.608998877066021,1.0427311644393213] |
|[0.2954946352117687,-0.2763864586236301]|
|[-2.5725591062870428,-1.3169815431109795]|
|[-3.4008228020493454,0.5073029625781897]|
|[-4.366720316263419,-0.3364827059771091]|
|[-2.7254698167724003,0.5058604987046365]|
|[-4.958112358381605,1.2627579957290729] |
|[-3.6934951846422712,1.3822762065335454]|
+----------------------------------------+
only showing top 20 rows
```

Na segunda parte, foi construída uma pipeline de transformação, que integra as etapas anteriores: a criação do vetor de características com VectorAssembler, a normalização com StandardScaler, e a redução de dimensionalidade com PCA. A função Pipeline permite encadear essas operações em um fluxo contínuo, garantindo que todos os passos sejam aplicados sequencialmente aos dados. O pipeline é ajustado aos dados originais com o comando fit, resultando em um modelo capaz de realizar todas as etapas de preparação dos dados de forma automática.

```
[ ]  pca_pipeline = Pipeline(stages=[VectorAssembler(inputCols=X, outputCol='features'),
                                     StandardScaler(inputCol='features', outputCol='scaled_features'),
                                     PCA(k=2, inputCol='scaled_features', outputCol='pca_features')])

[ ]  pca_pipeline_model = pca_pipeline.fit(dados_generos)

[ ]  dados_generos_pca = pca_pipeline_model.transform(dados_generos)
```

5.0 DEFINIÇÃO E TESTES DO MODELO DE RECOMENDAÇÃO

Para o desenvolvimento do modelo de recomendação, escolhemos o algoritmo de K-means, uma técnica de *clusterização* não supervisionada que agrupa os dados em diferentes clusters com base em suas similaridades. A escolha do K-means se justifica pelo fato de que ele permite identificar padrões subjacentes nos dados, agrupando músicas com características acústicas e comportamentais semelhantes. Como o Spotify disponibiliza diversas variáveis numéricas relacionadas às músicas o K-means é capaz de explorar essas relações e criar grupos de faixas que compartilham perfis musicais semelhantes. Isso é ideal para a recomendação de músicas, pois faixas dentro de um mesmo cluster podem ser sugeridas aos usuários com base em suas preferências anteriores.

Outra vantagem do K-means é sua simplicidade e eficiência em lidar com grandes volumes de dados, como os que estamos analisando neste projeto. Como a base de dados do Spotify contém muitas faixas e atributos, o K-means permite segmentar esses dados de maneira escalável, gerando grupos coesos de músicas que podem ser facilmente explorados para recomendações. Além disso, o K-means é um algoritmo amplamente utilizado em sistemas de recomendação, especialmente quando se deseja fornecer sugestões baseadas em similaridade de atributos e não apenas no histórico de consumo do usuário. Isso faz dele uma escolha robusta e adequada para o nosso modelo de recomendação.

Com o K-means, podemos então sugerir músicas pertencentes ao mesmo cluster de faixas que o usuário já ouviu e gostou, aumentando a probabilidade de que as recomendações sejam bem recebidas. Essa abordagem baseada em similaridades explícitas entre características acústicas permite criar um sistema de recomendação mais intuitivo e alinhado às preferências do usuário, mesmo sem precisar do histórico exato de consumo.

5.1 IMPLEMENTANDO O ALGORITIMO

O algoritmo K-means foi implementado para realizar a clusterização das músicas com base nas suas características principais, que foram previamente reduzidas através da técnica de PCA. Inicialmente, o número de clusters foi definido como 5 (setK(5)), o que significa que o algoritmo K-means irá agrupar as músicas em cinco diferentes clusters, baseando-se nas similaridades entre suas características condensadas nas colunas de "pca_features". Foi também definido um valor de semente (setSeed(SEED)) para garantir que os resultados sejam reprodutíveis.

```
[ ] from pyspark.ml.clustering import KMeans

[ ] SEED = 1224

[ ] kmeans = KMeans(featuresCol='pca_features', predictionCol='cluster_pca').setK(5).setSeed(SEED)

[ ] model_kmeans = kmeans.fit(dados_generos_pca)

[ ] prections_kmeans = model_kmeans.transform(dados_generos_pca)

[ ] prections_kmeans.select('pca_features', 'cluster_pca').show(truncate=False)

+----------------------------------------+-----------+
|pca_features                            |cluster_pca|
+----------------------------------------+-----------+
|[2.5070953668885667,0.4381691373697943] |2          |
|[-0.5969679056633488,4.981612052751348] |2          |
|[-4.158460276223561,-0.8366525081079943]|4          |
|[-2.387344878512217,-0.4877989015663405]|0          |
|[-2.6501218371679083,-0.5756819768820474]|0         |
|[-1.496509120336763,1.8644183183717797] |2          |
|[-3.923520772157324,0.2851835002352836] |4          |
|[-4.611011109831114,-0.6783790472312378]|1          |
|[-2.837690063084229,-0.5712993716580518]|4          |
|[-2.706690139892783,-1.25937880797083]  |0          |
|[-4.6983313839242875,1.2765569680619446]|3          |
|[-3.375987496679868,0.7560741064307471] |4          |
|[-5.608998877066021,1.0427311644393213] |1          |
|[0.2954946352117687,-0.2763864586236301]|2          |
|[-2.5725591062870428,-1.3169815431109795]|0         |
|[-3.4008228020493454,0.5073029625781897]|4          |
|[-4.366720316263419,-0.3364827059771091]|4          |
|[-2.7254698167724003,0.5058604987046365]|4          |
|[-4.958112358381605,1.2627579957290729] |3          |
|[-3.6934951846422712,1.3822762065335454]|3          |
+----------------------------------------+-----------+
only showing top 20 rows
```

Após ajustar o modelo K-means com os dados de PCA, o comando transform foi utilizado para prever a qual cluster cada música pertence. O resultado é exibido na coluna "cluster_pca", que mostra o número do cluster ao qual cada conjunto de características principais foi atribuído. A visualização dos dados mostra as 20 primeiras músicas com suas respectivas características agrupadas, o que permite analisar a coesão de cada cluster. Esses clusters serão usados posteriormente para realizar as recomendações, sugerindo músicas de um mesmo grupo para usuários com preferências semelhantes.

5.2 VISUALIZACAO DOS TESTES

Com o algoritmo implementado foi realizada a preparação dos dados para a criação de um gráfico de dispersão que visualiza os clusters de gêneros de músicas formados pelo algoritmo K-means. O comando vector_to_array foi utilizado para converter as componentes principais de PCA (armazenadas na coluna "pca_features") em valores separados para as coordenadas X e Y, permitindo a plotagem dos dados em um gráfico 2D.

Primeiramente, as colunas "x" e "y" são criadas a partir das componentes principais, representando as duas dimensões geradas pelo PCA. Em seguida, os dados dos clusters e gêneros das músicas são mantidos nas colunas "cluster_pca" e "genres", respectivamente. Essa estrutura será usada para gerar o gráfico de dispersão, onde as músicas serão plotadas de acordo com suas coordenadas X e Y, e a cor ou outro elemento visual indicará o cluster a que pertencem. Essa visualização facilitará a interpretação da separação dos clusters e das similaridades entre os grupos de músicas.

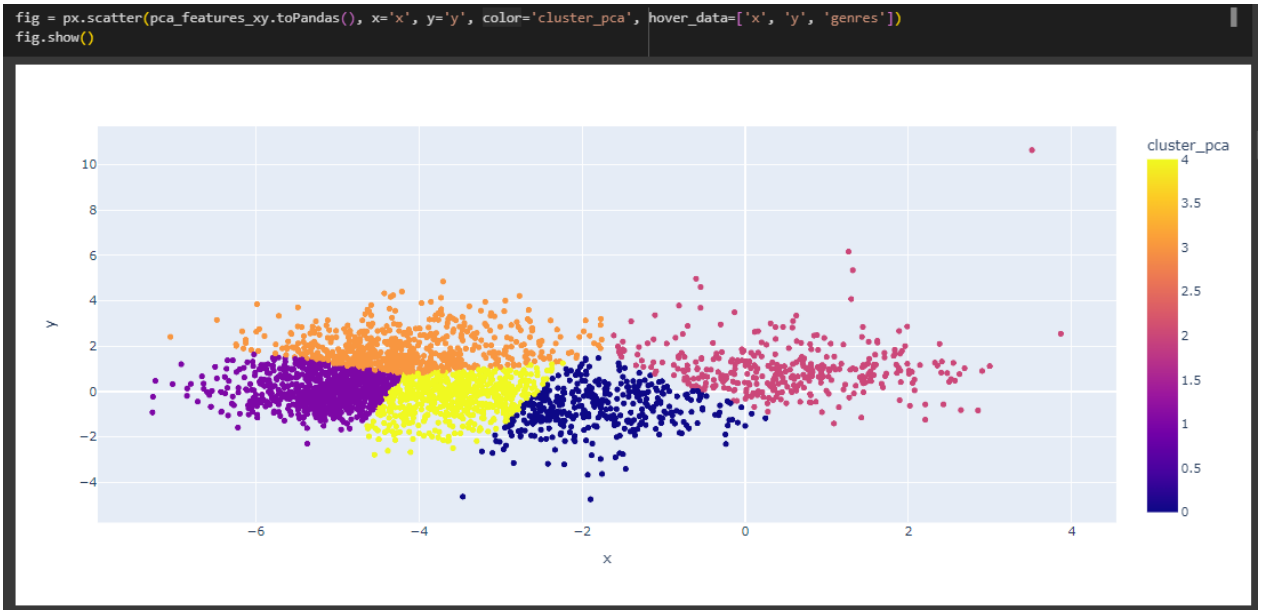
```
[ ] from pyspark.ml.functions import vector_to_array

[ ] pca_features_xy = prections_kmeans.withColumn('x', vector_to_array('pca_features')[0])\
                    .withColumn('y', vector_to_array('pca_features')[1])\
                    .select(['x', 'y', 'cluster_pca', 'genres'])

[ ] pca_features_xy.show()
```

```
+-------------------+-------------------+-----------+--------------------+
|                  x|                  y|cluster_pca|              genres|
+-------------------+-------------------+-----------+--------------------+
| 2.5070953668885667|0.43816913737697943|          2|21st century clas...|
|-0.5969679056633488|  4.981612052751348|          2|               432hz|
| -4.158460276223561|-0.8366525081079943|          4|               8-bit|
| -2.387344878512217|-0.4877989015663405|          0|                  []|
|-2.6501218371679083|-0.5756819768820474|          0|          a cappella|
| -1.496509120336763| 1.8644183183717797|          2|            abstract|
| -3.923520772157324| 0.2851835002352836|          4|       abstract beats|
| -4.611011109831114|-0.6783790472312378|          1|     abstract hip hop|
| -2.837690063084229|-0.5712993716580518|          4|           accordeon|
| -2.706690139892783|  -1.25937880797083|          0|           accordion|
|-4.6983313839242875| 1.2765569680619446|          3|          acid house|
| -3.375987496679868| 0.7560741064307471|          4|           acid rock|
| -5.608998877066021| 1.0427311644393213|          1|         acid trance|
| 0.2954946352117687|-0.2763864586236301|          2|          acousmatic|
|-2.5725591062870428|-1.3169815431109795|          0|       acoustic blues|
|-3.4008228020493454| 0.5073029625781897|          4|         acoustic pop|
| -4.366720316263419|-0.3364827059771091|          4|        acoustic punk|
|-2.7254698167724003| 0.5058604987046365|          4|        acoustic rock|
| -4.958112358381605| 1.2627579957290729|          3|         action rock|
|-3.6934951846422712| 1.3822762065335454|          3|           adoracion|
+-------------------+-------------------+-----------+--------------------+
only showing top 20 rows
```

Em seguida, construímos um gráfico de dispersão com esses valores, ele apresenta a visualização dos clusters criados pelo algoritmo K-means, com base nas duas primeiras componentes principais (PCA) das características acústicas das músicas. Cada ponto no gráfico representa uma música, posicionada de

acordo com suas coordenadas "x" e "y", que são as dimensões reduzidas resultantes da aplicação do PCA. As cores dos pontos indicam os diferentes clusters, variando de 0 a 4.

Além disso, as informações adicionais, como o gênero da música e o número do cluster ao qual ela pertence, são exibidas ao passar o mouse sobre os pontos, o que facilita a interpretação dos grupos e suas características. Esse tipo de visualização é útil para observar como as músicas foram agrupadas com base em suas similaridades, permitindo identificar padrões e possíveis outliers nos dados de forma visual e intuitiva.



6.0 AVALIAÇÃO DO MODELO

A avaliação do modelo de recomendação será feita utilizando a métrica de distância euclidiana, que permitirá medir a proximidade dos pontos dentro de cada cluster em relação ao centróide correspondente. Essa métrica será fundamental para verificar a coesão dos agrupamentos formados pelo K-means, uma vez que, quanto menores forem as distâncias entre as músicas e o centróide, mais homogêneo será o cluster. O objetivo é assegurar que o modelo consiga formar grupos com características musicais semelhantes, garantindo a eficácia das recomendações.

Utilizaremos a biblioteca scipy.spatial.distance para calcular as distâncias euclidianas entre os pontos e os centróides. A dispersão dos clusters com base nas distâncias calculadas. Clusters com menores distâncias indicarão uma maior similaridade entre as músicas agrupadas, enquanto clusters com distâncias maiores poderão sugerir uma segmentação menos eficaz. Essa abordagem fornecerá uma avaliação robusta da qualidade dos agrupamentos gerados pelo modelo de recomendação.

7.0 REFERENCIAL TEÓRICO

Sistemas de Recomendação

Os sistemas de recomendação são amplamente utilizados em plataformas como Spotify, Netflix e Amazon para fornecer sugestões personalizadas aos usuários com base em seus hábitos de consumo. Segundo Ricci, Rokach e Shapira (2015), os sistemas de recomendação podem ser classificados em três grandes abordagens: baseada em conteúdo, colaborativa e híbrida. No nosso projeto, adotamos uma abordagem baseada em conteúdo, onde as características das músicas, como *acousticness* e *danceability*, são usadas para formar agrupamentos de faixas semelhantes.

Além disso, as técnicas de *clusterização*, como o K-means, são amplamente utilizadas para agrupar itens com características semelhantes. O K-means é um algoritmo não supervisionado que visa particionar um conjunto de dados em "k" clusters, minimizando a distância euclidiana entre os pontos de dados e os centróides dos clusters. O K-means é conhecido por sua simplicidade e eficiência computacional, o que o torna uma escolha comum para problemas de grande escala, como a recomendação de músicas. Segundo Xu e Tian (2015), o K-means é particularmente útil para a formação de grupos em grandes volumes de dados de alto dimensionalidade, como o utilizado neste projeto.

Redução de Dimensionalidade com PCA

A técnica de *Principal Component Analysis* (PCA) foi utilizada neste projeto para reduzir a dimensionalidade dos dados, facilitando a visualização e o processamento das características musicais. De acordo com Jolliffe (2002), o PCA é uma ferramenta matemática eficaz para transformar variáveis altamente correlacionadas em um conjunto de componentes principais não correlacionados, o que simplifica a estrutura dos dados e melhora a eficiência computacional de modelos complexos. No contexto de sistemas de recomendação, o PCA é utilizado para manter as informações mais relevantes e reduzir o ruído dos dados, como observado por Aggarwal (2016).

Ao aplicar o PCA, conseguimos visualizar os dados em duas dimensões principais, o que facilita a interpretação dos clusters formados. Essa abordagem ajuda a destacar como as músicas se relacionam umas com as outras com base em suas características acústicas, como descrito por Benesty(2009) .

Avaliação de Modelos de Clusterização

A avaliação de modelos de clusterização é uma etapa fundamental para garantir que os agrupamentos formados são coerentes e refletem a similaridade dos dados. Para isso, utilizaremos a métrica de distância euclidiana, que é uma das formas mais comuns de medir a proximidade entre pontos em um espaço

multidimensional. Conforme detalhado por Hastie, Tibshirani e Friedman (2009), a distância euclidiana é uma métrica amplamente adotada para avaliar a coesão dos clusters, garantindo que os pontos de dados estejam devidamente agrupados.

Além disso, utilizaremos a métrica de Silhouette Score para medir a qualidade dos clusters formados. O Silhouette Score avalia a separação entre clusters e a compactação dos pontos dentro de cada cluster. Segundo Kaufman e Rousseeuw (2005), essa métrica é particularmente útil para determinar se os agrupamentos formados são apropriados e para escolher o número ideal de clusters em problemas de *machine learning* não supervisionado.

Aplicações em Recomendação Musical

No contexto da recomendação musical, várias abordagens têm sido propostas para melhorar a personalização das recomendações. Segundo Celma (2010), sistemas de recomendação baseados em características musicais, como análise de áudio e atributos acústicos, têm mostrado eficácia em fornecer recomendações mais precisas, em comparação com sistemas que se baseiam apenas no histórico de consumo dos usuários. Nossa abordagem foca na similaridade entre as músicas, agrupando-as com base em suas características acústicas e comportamentais, o que possibilita uma recomendação mais assertiva e personalizada.

A utilização de técnicas de aprendizado de máquina para análise de grandes volumes de dados musicais tem crescido nos últimos anos. Em seu estudo, Westermann et al. (2020) exploraram o uso de sistemas de recomendação baseados em conteúdo e *clusterização* para melhorar a experiência dos usuários em plataformas de streaming de música. Esse estudo reforça a importância de modelos híbridos que combinem atributos musicais com o comportamento do usuário, o que se alinha com os objetivos do nosso projeto.

REFERENCIAS:

BHOLLOWALIA, P.; KUMAR, A. EBK-means: A clustering technique based on elbow method and k-means in WSN. *International Journal of Computer Applications*, 2014.

HARTIGAN, J. A.; WONG, M. A. Algorithm AS 136: A k-means clustering algorithm. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, v. 28, n. 1, p. 100-108, 1979.

JOLLIFFE, I. T. *Principal Component Analysis*. 2. ed. New York: Springer, 2002.

LI, Y.; LIU, B.; XIE, L. Dimensionality reduction for clustering. *Journal of Machine Learning Research*, v. 21, p. 1-28, 2020.

RENDLE, S. et al. Factorization machines. In: *2010 IEEE International Conference on Data Mining*. Proceedings… Sydney, Australia: IEEE, 2010. p. 995-1000.

SCHEDL, M. et al. Current challenges and visions in music recommender systems research. *International Journal of Multimedia Information Retrieval*, v. 7, n. 2, p. 95-116, 2018.

SAMMUT, C.; WEBB, G. I. *Encyclopedia of Machine Learning*. New York: Springer, 2011.

Ricci, F., Rokach, L., & Shapira, B. (2015). *Recommender Systems Handbook*. Springer.

Xu, R., & Tian, Y. (2015). *Clustering Techniques and Applications*. Springer.

Jolliffe, I. (2002). *Principal Component Analysis*. Springer.

Aggarwal, C. C. (2016). *Recommender Systems: The Textbook*. Springer.

Benesty, J., Chen, J., Huang, Y., & Cohen, I. (2009). *Noise Reduction in Speech Processing*. Springer.

Hastie, T., Tibshirani, R., & Friedman, J. (2009). *The Elements of Statistical Learning*. Springer.

Kaufman, L., & Rousseeuw, P. J. (2005). *Finding Groups in Data: An Introduction to Cluster Analysis*. Wiley.

Celma, Ò. (2010). *Music Recommendation and Discovery in the Long Tail*. Springer.

Westermann, J., Schmidt, C., & Lischka, M. (2020). Improving User Experience through Cluster-Based Music Recommendation Systems. *International Journal of Music Information Retrieval*, 5(2), 101-112.