Processamento e analise

Contexto

Num mundo onde a indústria musical é extremamente competitiva e em constante evolução, a capacidade de tomar decisões baseadas em dados tornou-se um ativo inestimável.

Uma gravadora enfrenta o emocionante desafio de lançar um novo artista no cenário musical global.

Felizmente, ela tem uma ferramenta poderosa em seu arsenal: um extenso conjunto de dados do Spotify com informações sobre as músicas mais ouvidas em 2023.

Hipóteses

Músicas com BPM (Batidas Por Minuto) mais altos fazem mais sucesso em termos de número de streams no Spotify.

As músicas mais populares no ranking do Spotify também possuem um comportamento semelhante em outras plataformas, como a Deezer.

A presença de uma música em um maior número de playlists está correlacionada com um maior número de streams.

Artistas com um maior número de músicas no Spotify têm mais streams.

As características da música influenciam o sucesso em termos de número de streams no Spotify.

Objetivo

Validar as hipóteses através da análise de dados e fornecer recomendações estratégicas com base nas descobertas e tomar decisões informadas que aumentem as chances de alcançar o “sucesso”.

Estrutura dos dados

Neste banco de dados temos 3 tabelas: Competition, Spotify e Technical.

Etapas do projeto

Primeiros passos

1. Coleta de dados
2. Importações
3. Tratamento dos daddos

Análise dos dados

4. Criar variáveis categóricas
5. Unir tabelas
6. Aplicar medidas de tendência central
7. Aplicar medidas de dispersão
8. Calcular quartis, decis ou percentis
9. Calcular correlação entre variáveis

Validando hipóteses

10. Aplicar segmentação Power Bi
11. Validar hipótese Power Bi

Etapas do projeto

1. **Coleta de dados (Big Query)**

   A base de dados do Spotify foi disponibilizada pela Laboratoría.

   Coleta de dados (Colab)

```
df = pd.read_csv('hipotese.csv')
```

2. **Importações**

   Importamos 3 arquivos em formato CSV para o BIGQUERY.

3. **Tratamento dos dados**

   Identificar e tratar valores nulos

   Utilizamos os códigos abaixo para encontrar valores nulos nas tabelas.

   Na tabela *track_in_competition* encontramos 50 nulos.

```
SELECT COUNT (*)
FROM `projeto-hipotese.origem.track_in_competition`
WHERE track_id IS NULL OR in_apple_playlists IS NULL OR in
```

   Na tabela *track_technical_info* encontramos 95 nulos.

```
SELECT COUNT(*)
FROM `projeto-hipotese.origem.track_technical_info` AS tec
WHERE track_id IS NULL OR bpm IS NULL OR technical.key IS
```

   Na tabela *track_in_spotify não* encontramos nulos.

```
SELECT COUNT(*)
FROM `projeto-hipotese.origem.track_in_spotify`
WHERE track_id IS NULL OR track_name IS NULL OR artist_s__
```

> ! Deixamos os valores nulos ao reparar que o total de valores nulos correspondem cerca de 10% dos dados totais e a retirada das músicas podem influenciar na tomada de decisão.

Identificar e tratar valores duplicados

Utilizamos os códigos abaixo para encontrar valores duplicados nas tabelas.

Na tabela *track_in_competition não* encontramos valores duplicados.

```
SELECT track_id, COUNT(*)
FROM `projeto-hipotese.origem.track_in_competition`
GROUP BY track_id
HAVING COUNT(*);
```

Na tabela *track_technical_info não* encontramos valores duplicados.

```
SELECT COUNT (*)
FROM `projeto-hipotese.origem.track_technical_info`
GROUP BY track_id
HAVING COUNT(*) >1;
```

Na tabela *track_in_spotify* encontramos 4 valores duplicados.

```
SELECT  track_name,artist_s__name,released_year,
COUNT(*)  AS Conte, max(track_id)id1, min(track_id) AS id2
FROM `projeto-hipotese.origem.track_in_spotify`
GROUP BY  track_name,artist_s__name, released_year
HAVING COUNT(*) >1;
```

! Consideramos dados duplicados para os dois primeiros artistas. Mantemos as músicas dos 2 últimos por terem dados diferentes na base e/ou não terem dados nulos em colunas relevantes.

Linha	track_name	artist_s__name	quantidade	id2	id1
1	SNAP	Rosa Linn	2	5675634	3814670
2	About Damn Time	Lizzo	2	7173596	5080031
3	Take My Breath	The Weeknd	2	4586215	1119309
4	SPIT IN MY FACE!	ThxSoMch	2	8173823	4967469

Músicas excluídas da tabela *track_in_spotify*.

Identificar e tratar dados fora do escopo de análise

> ! Mantivemos todas as variáveis para a análise. Chegamos a conclusão que excluir uma variável neste momento pode comprometer análises posteriores.

Identificar e tratar dados discrepantes em variáveis categóricas

Utilizamos o código abaixo nas para encontrar valores discrepantes na tabela *track_in_spotify*.

```
SELECT track_id,REGEXP_REPLACE(track_name,r'[^A-Za-z0-9\s]
FROM `projeto-hipotese.origem.track_in_spotify`;
```

> ! Foram encontramos caracteres no nome de algumas musicas e artistas, usamos a formula acima para remover os caracteres especiais.

Depois de utlizar a formula encontramos 2 linhas em branco, optamos por deixar devido a somente o track_name_clean estar em branco.

track_id	track_name_clean	artist_s__name_clean	artist_count	released_year
5865058		Fujii Kaze	1	2020
6720570		YOASOBI	1	2023

Verificar e alterar o tipo de dados

Utilizamos o código abaixo para encontrar tipos de dados incorretos na tabela *track_in_spotify*.

```
SELECT * EXCEPT(streams), SAFE_CAST(streams AS INT64) AS s
FROM `projeto-hipotese.origem.track_in_spotify`;
```

> ! Encontramos uma variável STRING onde deveria ser INTEGGER. Alteramos o dado da variável para NULL.

Identificar e tratar dados discrepantes em variáveis numéricas com MAX, MIN e AVG.

Utilizamos o código abaixo para encontrar valores mínimo, máximo e média nas tabelas.

Código para a tabela *spotify*.

```
SELECT
MAX(artist_count),
MAX(released_year),
MAX(released_month),
MAX(released_day),
MAX(in_spotify_playlists),
MAX(in_spotify_charts),
MAX (streams_clean) ,

MIN(artist_count),
MIN(released_year),
MIN(released_month),
MIN(released_day),
MIN(in_spotify_playlists),
MIN(in_spotify_charts),
MIN (streams_clean) ,
```

```
AVG(artist_count),
AVG(released_year),
AVG(released_month),
AVG(released_day),
AVG(in_spotify_playlists),
AVG(in_spotify_charts),
AVG (streams_clean),

FROM `projeto-hipotese.tratados.spotify`;
```

Código para a tabela *track_in_competition.*

```
SELECT
MAX(in_apple_playlists),
MAX(in_apple_charts),
MAX(in_deezer_playlists),
MAX(in_deezer_charts),
MAX(in_shazam_charts) ,

MIN(in_apple_playlists),
MIN(in_apple_charts),
MIN(in_deezer_playlists),
MIN(in_deezer_charts),
MIN(in_shazam_charts),

AVG(in_apple_playlists),
AVG(in_apple_charts),
AVG(in_deezer_playlists),
AVG(in_deezer_charts),
AVG(in_shazam_charts) ,

FROM  `projeto-hipotese.tratados.track_in_competition`;
```

Código para a tabela *technical.*

```
SELECT
MAX(bpm),
MAX(danceability__),
MAX(valence__),
MAX(energy__),
MAX(acousticness__),
MAX(instrumentalness__),
MAX(liveness__),
MAX(speechiness__),

MIN(bpm),
MIN(danceability__),
MIN(valence__),
MIN(energy__),
MIN(acousticness__),
MIN(instrumentalness__),
MIN(liveness__),
MIN(speechiness__),

AVG(bpm),
AVG(danceability__),
AVG(valence__),
AVG(energy__),
AVG(acousticness__),
AVG(instrumentalness__),
AVG(liveness__),
AVG(speechiness__),

FROM `projeto-hipotese.tratados.technical`;
```

> ! Encontramos uma linha na tabela spotify que continha o ano de lançamento 1930, mas o lançamento da musica foi em 2022.

4. **Criar novas variáveis**

No código abaixo criamos a variável ano de lançamento no formato ano/mês/dia

```
SELECT track_id, DATE(CONCAT(CAST(released_year AS STRING)
FROM `projeto-hipotese.tratados.spotify`;
```

No código abaixa criamos a variável que soma as playlists de todos os streams.

```
SELECT track_id,in_apple_playlists,in_deezer_playlists, in
FROM `projeto-hipotese.tratados.tratatos_track_in_competit
```

No código abaixo criamos uma variável para contar a quantidade total de música por artista.

```
SELECT artist_s__name_clean, COUNT(*) AS counta
FROM `projeto-hipotese.tratados.spotify`
GROUP BY artist_s__name_clean;
```

5. **Unir Tabelas**

   No código unimos as 3 tabelas e incluímos as variáveis que somam as playlists e a de data.

```
SELECT spotify.track_id, track_name_clean, artist_s__name_
in_spotify_charts,streams_clean,in_apple_playlists, in_app
valence__, energy__,acousticness__, instrumentalness__, li
spotify.in_spotify_playlists + competition.in_apple_playli

FROM `projeto-hipotese.tratados.spotify` AS spotify
LEFT JOIN `projeto-hipotese.track_in_competition.track_in_
LEFT JOIN `projeto-hipotese.track_in_competition.track_tec
LEFT JOIN `projeto-hipotese.tratados.realeased_date` AS re
```

> Agora no Power BI, começamos a aplicar tabelas e gráficos para a visualização dos dados.

6. **Aplicar medidas de tendência central**

   Aplicamos medidas como média, mediana para variável streams e soma_playlist.

Média de soma_playlist	Mediana de soma_playlist
5654,07	2302

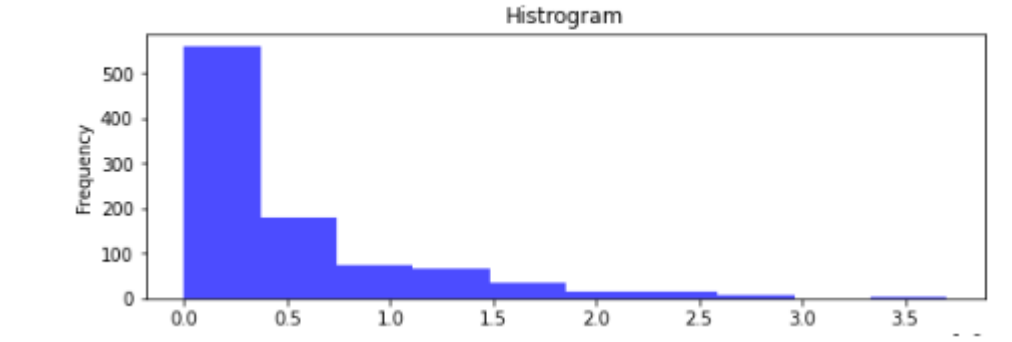
Média de streams_clean	Mediana de streams_clean
513693291,16	289165139

> ! A média está consideravelmente distante da mediana, é um sinal de que os dados podem conter valores extremos, ou seja, outliers.

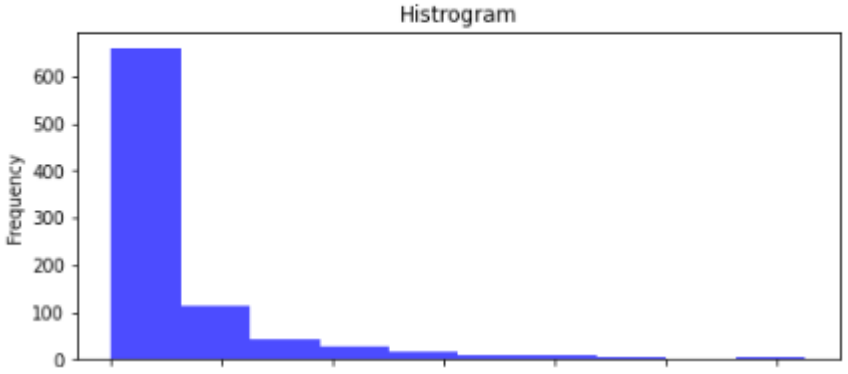
Ver distribuição

Através do histrograma criado em python podemos ver a distrubuição das variaveis abaixo

streams_clean



soma_playlist



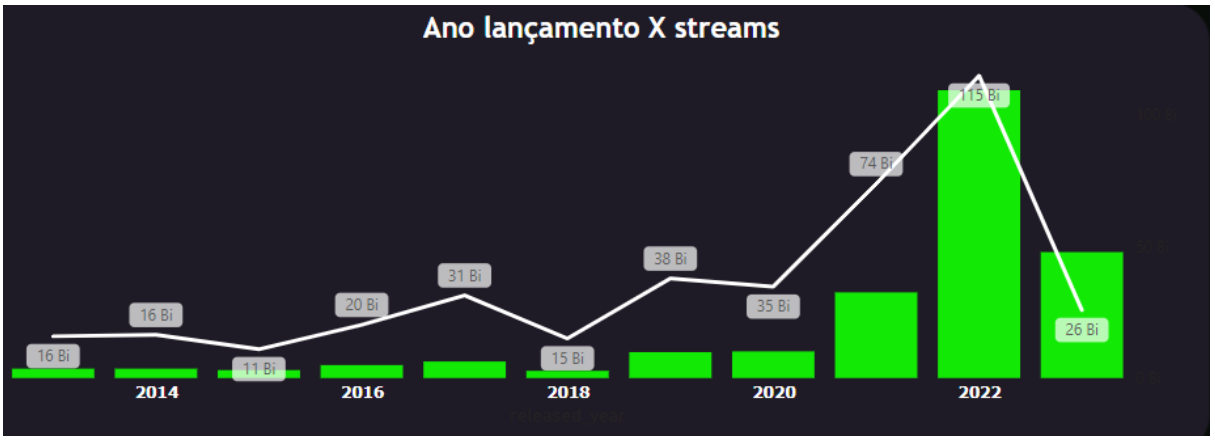
7.Aplicar medidas de dispersão

Soma de soma_playlist	Média de soma_playlist	Mediana de soma_playlist	Desvio padrão de soma_playlist	Variação de soma_playlist
5371370	5654,07	2302	8922,64	79613479,06

Média de streams_clean	Soma de streams_clean	Mediana de streams_clean	Desvio padrão de streams_clean	Variação de streams_clean
513693291,16	488008626601	289165139	567072431,93	321571143054950500,00

❗ O desvio padrão nas duas variáveis aponta para uma dispersão significativa em relação à mediana, indicando uma ampla diversidade nos dados da amostra. Isso sugere que os valores individuais estão consideravelmente distantes da mediana, o que reflete uma considerável variabilidade e uma ampla gama de valores na amostra.

Visualizar dados ao longo do tempo



❗ No gráfico acima conseguimos visualizar como as plataformas de streams se comportaram ao longo do tempo e como o ano de lançamento da música afeta a visibilidade que a plataforma tem.

8.Calcular quartis, decis ou percentis

Query para calcular os quartis para as variáveis dos atributos musicais e para as streams, categorizando-os posteriormente.

```
WITH Quartiles AS (
  SELECT
    streams_clean,
```

```
    bpm,
   danceability__,
   valence__,
   energy__,
   acousticness__,
   instrumentalness__,
   liveness__,
   speechiness__,
    NTILE(4) OVER (ORDER BY streams_clean) AS quartile_stre
    NTILE(4) OVER (ORDER BY bpm) AS quartile_bpm,
    NTILE(4) OVER (ORDER BY danceability__) AS quartile_dar
    NTILE(4) OVER (ORDER BY valence__) AS quartile_valence_
    NTILE(4) OVER (ORDER BY energy__) AS quartile_energy__,
    NTILE(4) OVER (ORDER BY acousticness__) AS quartile_acc
    NTILE(4) OVER (ORDER BY instrumentalness__) AS quartile
    NTILE(4) OVER (ORDER BY liveness__) AS quartile_livenes
    NTILE(4) OVER (ORDER BY speechiness__) AS quartile_spee

    ROW_NUMBER() OVER (PARTITION BY streams_clean ORDER BY
  FROM `projeto-hipotese.tratados.teste`
)

SELECT
  a.*,
  Quartiles.quartile_streams_clean,
   CASE
    WHEN quartile_streams_clean<= 2 THEN 'Baixo'
    WHEN quartile_streams_clean >= 3 THEN 'Alto'
END AS categoria_streams_clean,

  Quartiles.quartile_bpm,
  CASE
    WHEN quartile_bpm <= 2 THEN 'Baixo'
    WHEN quartile_bpm >= 3 THEN 'Alto'
END AS categoria_bpm,

  Quartiles.quartile_danceability__,
   CASE
```

```
    WHEN quartile_danceability__ <= 2 THEN 'Baixo'
    WHEN quartile_danceability__>= 3 THEN 'Alto'

END AS categoria_danceability__,

  Quartiles.quartile_valence__,
     CASE
    WHEN quartile_valence__ <= 2 THEN 'Baixo'
    WHEN quartile_valence__ >= 3 THEN 'Alto'

END AS categoria_valence__ ,

  Quartiles.quartile_energy__,
       CASE
    WHEN quartile_energy__ <= 2 THEN 'Baixo'
    WHEN quartile_energy__ >= 3 THEN 'Alto'

END AS categoria_energy__ ,

  Quartiles.quartile_acousticness__,
       CASE
    WHEN quartile_acousticness__<=2 THEN 'Baixo'
    WHEN quartile_acousticness__>=3 THEN 'Alto'

END AS categoria_acousticness__ ,

  Quartiles.quartile_instrumentalness__,
        CASE
    WHEN quartile_instrumentalness__<= 2 THEN 'Baixo'
    WHEN quartile_instrumentalness__>= 3 THEN 'Alto'

END AS categoria_instrumentalness__ ,

  Quartiles.quartile_liveness__,
 CASE
    WHEN quartile_liveness__<= 2 THEN 'Baixo'
    WHEN quartile_liveness__>= 3 THEN 'Alto'
```

```
END AS categoria_liveness__ ,

  Quartiles.quartile_speechiness__,
   CASE
    WHEN quartile_speechiness__<= 2 THEN 'Baixo'
    WHEN quartile_speechiness__>= 3 THEN 'Alto'

END AS categoria_speechiness__ ,


FROM `projeto-hipotese.tratados.teste` a
LEFT JOIN (
  SELECT * FROM Quartiles WHERE rn = 1
) Quartiles
ON a.streams_clean = Quartiles.streams_clean;
```

> ! Criamos os quartis, para compreender a distribuição dos dados e aplicamos categorias, essas categorias serão essenciais para uma análise mais aprofundada posteriormente.

9. **Calcular correlação entre variáveis**

   No código abaixo calculamos a correlação da soma das playlists e de stream.

```
SELECT CORR(streams_clean, soma_playlist) AS correlacao_pl
FROM `projeto-hipotese.tratados.tabelas_unificadas`;
```

INFORMAÇÕES DO JOB | RESULTADOS

Linha	correlacao_playlist	correlacao_danceabj
1	0.783176141098...	-0.10573626604...

> ! Na análise da correlação entre streams e playlists, observamos uma associação, indicando uma relação significativa entre essas duas variáveis. Por outro lado, ao examinarmos a correlação com a de streams e danceability, notamos uma associação mais fraca, sugerindo uma ligação menos marcante entre as variaveis.

10. **Aplicar segmentação**

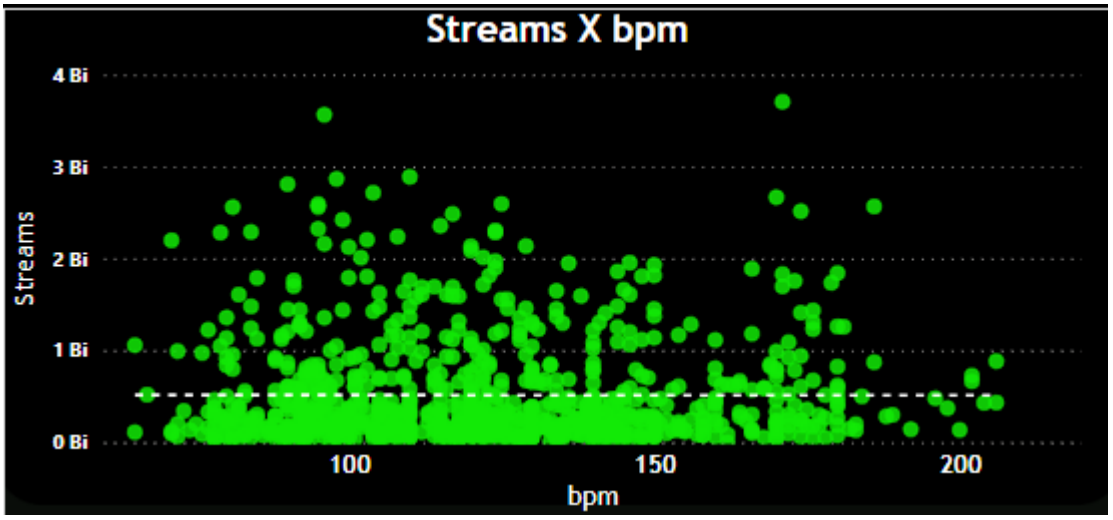
categoria_streams_clean	Média de danceability__	Média de energy__	Média de valence__	Média de acousticness__	Média de liveness__	Média de speechiness__	Média de instrumentalness__
Baixo	68,06	64,98	52,46	27,96	18,80	11,42	1,62
Alto	65,89	63,55	50,30	26,27	17,53	8,87	1,56
Total	**66,98**	**64,26**	**51,38**	**27,11**	**18,17**	**10,15**	**1,59**

> ! Aplicamos a segmentação efetuada com os quartis nas categorias musicais

11. **Validar hipótese**

Abaixo apresentaremos as hipóteses e as análises feitas para validar ou revogar uma hipótese.

> *Músicas com BPM (Batidas Por Minuto) mais altos fazem mais sucesso em termos de número de streams no Spotify.*



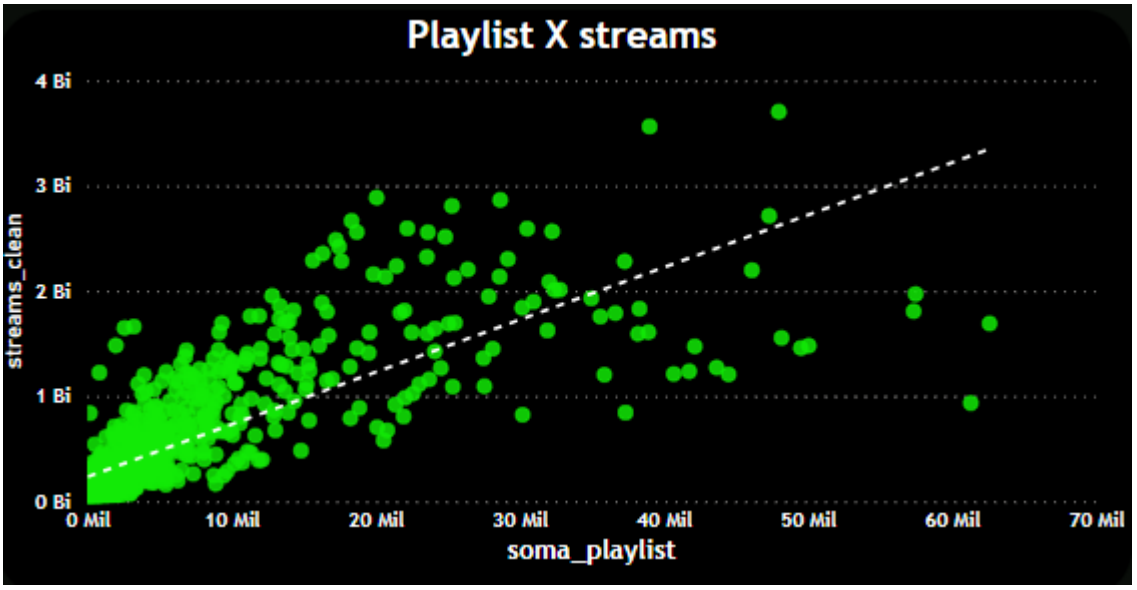
> **!** A hipótese foi anulada após a análise do gráfico de dispersão, que revelou a ausência de correlação direta entre o número de streams e o BPM (Batidas por minuto). Os dados sugerem que outros fatores podem influenciar o número de streams, enquanto o BPM não parece desempenhar um papel significativo neste contexto.

> *As músicas mais populares no ranking do Spotify também possuem um comportamento semelhante em outras plataformas, como a Deezer.*

track_name_clean	Soma de in_spotify_charts	Soma de in_apple_charts	Soma de in_deezer_charts
Adore You	1	71	2
Cay La Noche feat Cruz Cafun Abhir Hathi Bejo EL IMA	1	5	0
Daydreaming	1	1	0
ELEVEN	1	89	0
Golden	1	24	0
HUMBLE	1	114	0
Hummingbird Metro Boomin James Blake	1	20	0
I Really Want to Stay at Your House	1	0	0
Keep Driving	1	1	0
Malvad	1	3	0
Mejor Que Yo	1	13	0
Music For a Sushi Restaurant	1	11	0
San Lucas	1	0	0
Sweet Child O Mine	1	151	3
Un Verano Sin Ti	1	12	0
X LTIMA	1	3	0
Total	**16**	**518**	**5**

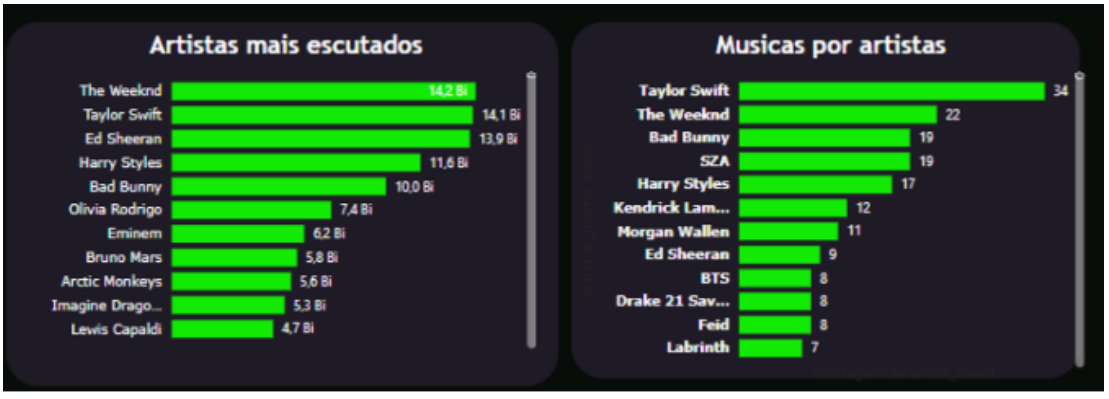
> **!** Não necessariamente, as músicas top 1 no Spotify possuem um ranking bem variável na Apple sendo a maioria abaixo do top 20, já no Deezer vemos que a maioria possui ranking 0 (pode ser um erro ou 0 representa 1).

> *A presença de uma música em um maior número de playlists está correlacionada com um maior número de streams.*



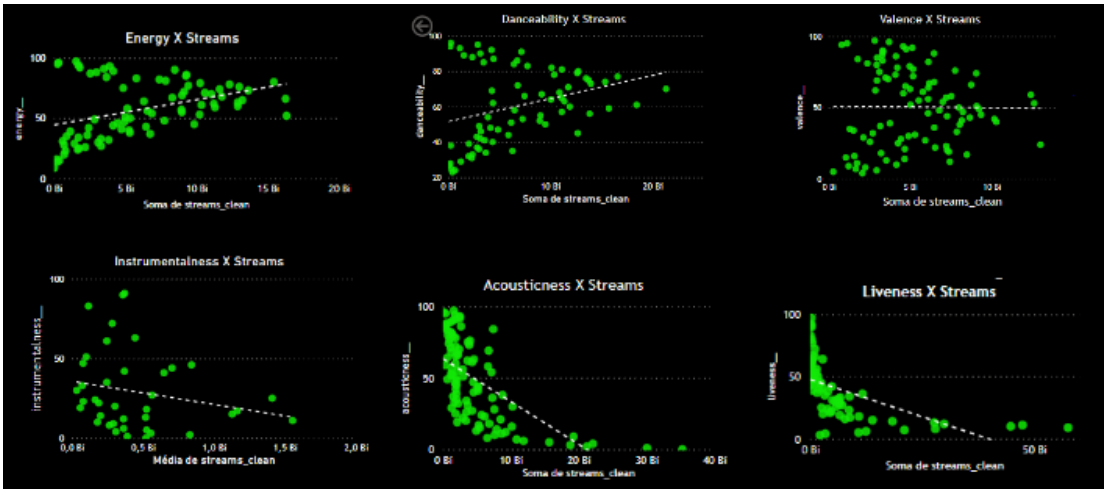
> **!** A hipótese está correta. Conforme o gráfico acima, podemos verificar que quanto maior o número de playlists maior o número de streams. A linha de tendência central indica exatamente esse comportamento.

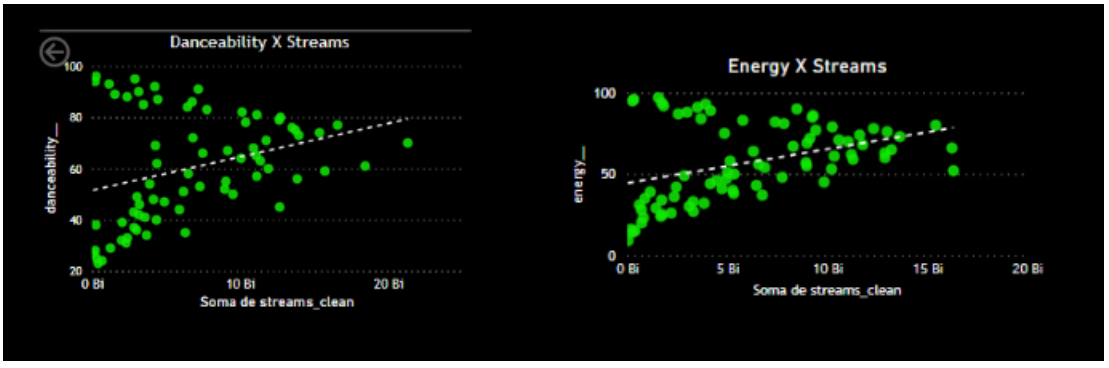
> *Artistas com um maior número de músicas no Spotify têm mais streams.*



> ! Correto. Ao analisar os dados, verificamos que os artistas com maiores número de música possuem um maior número de stream.

> *As características da música influenciam o sucesso em termos de número de streams no Spotify.*





> *As características da música influenciam o sucesso em termos de número de streams no Spotify.*

! As características tem uma correlação, tanto positiva ou para negativa, porém não necessariamente significa que seguindo as características musicais alcançará o sucesso já que depende também de fatores externos.