Project_4:Music_Popularity_Prediction_v2

November 7, 2024

1 Project 4: Music Popularity Prediction

By: Robert S Balch

Certainly! I'll revise the hypothesis to remove the chart performance metrics for the artist, focusing instead on the song itself. Here's the revised version:

1.1 Problem Statement:

We aim to develop a predictive model for song popularity on Spotify's Top 200 Weekly (Global) charts for 2020 & 2021. This is a supervised regression problem where we will use various features to predict a continuous target variable representing song popularity.

1.2 Target Variable:

- The target variable is the song's popularity score on Spotify's Top 200 Weekly (Global) charts.

1.3 Predictor Variables:

1. Audio Features:
   - Loudness: Expected to positively correlate with popularity
   - Energy: Likely to strongly predict popularity
   - Danceability: May contribute to higher popularity scores
   - Valence (positiveness): Potentially influences popularity
   - Tempo: Could be a factor, especially genre-specific
2. Artist Popularity:
   - Number of artist followers: Expected to be a significant predictor
3. Song Characteristics:
   - Duration: Potentially negatively correlated with recent popularity trends
4. Genre:
   - Binary features for major genres (e.g., pop, hip-hop): May influence popularity
5. Release Timing:
   - Days since release: Could influence accumulated popularity
6. Feature Interactions:
   - Audio features × Artist popularity: Interaction terms to capture combined effects
7. Cultural and Temporal Factors:
   - Year (2020 vs 2021): To account for potential shifts due to the COVID-19 pandemic

1.4 Model Approach:

We will develop a regression model (e.g., Linear Regression, Random Forest, Gradient Boosting) to predict song popularity scores based on these features. The goal is to identify the most influential factors and their relative importance in determining song popularity on Spotify's Top 200 Weekly charts.

1.5 Evaluation Metrics:

Model performance will be evaluated using standard regression metrics such as Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and R-squared (R^2).

By framing this project as a supervised regression problem, we aim to create a predictive model that can estimate a song's popularity score based on its characteristics, artist factors, and temporal factors. This approach allows us to quantify the relationship between various features and song popularity, potentially providing insights for music industry professionals and artists seeking to understand and improve their songs' chart performance.

The data. A chosen data set is provided by DDC Data Science

2 Imports

[2]:
```
import sys
print(sys.executable)
```

```
/usr/local/bin/python
```

[3]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.colors as mcolors
import seaborn as sns

from sklearn.preprocessing import StandardScaler
from sklearn.preprocessing import MinMaxScaler
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor
import xgboost as xgb

from sklearn.metrics import mean_squared_error, root_mean_squared_error,r2_score
```

[4]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
#n_test_split
```

```
from sklearn.linear_model import LinearRegression
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor
import xgboost as xgb

from sklearn.metrics import mean_squared_error, root_mean_squared_error,r2_score
```

[5]:
```
%%capture
url = "https://ddc-datascience.s3.amazonaws.com/Projects/Project.4-Spotify/Data/
 ↪Spotify.csv"
!curl -s -I {url}
```

3 Data Exploration

[6]:
```
df_1 = pd.read_csv(url).copy()
```

3.1 Head

[7]:
```
df_1.head()
```

[7]:

	Index	Highest Charting Position	Number of Times Charted	\
0	1	1	8	
1	2	2	3	
2	3	1	11	
3	4	3	5	
4	5	5	1	

	Week of Highest Charting	Song Name	Streams	\
0	2021-07-23--2021-07-30	Beggin'	48,633,449	
1	2021-07-23--2021-07-30	STAY (with Justin Bieber)	47,248,719	
2	2021-06-25--2021-07-02	good 4 u	40,162,559	
3	2021-07-02--2021-07-09	Bad Habits	37,799,456	
4	2021-07-23--2021-07-30	INDUSTRY BABY (feat. Jack Harlow)	33,948,454	

	Artist	Artist Followers	Song ID	\
0	Måneskin	3377762	3Wrjm47oTz2sjIgck11l5e	
1	The Kid LAROI	2230022	5HCyWlXZPP0y6Gqq8TgA20	
2	Olivia Rodrigo	6266514	4ZtFanR9U6ndgddUvNcjcG	
3	Ed Sheeran	83293380	6PQ88X9TkUIAUIZJHW2upE	
4	Lil Nas X	5473565	27NovPIUIRrOZoCHxABJwK	

	Genre	…	Danceability	Energy	Loudness	\
0	['indie rock italiano', 'italian pop']	…	0.714	0.8	-4.808	
1	['australian hip hop']	…	0.591	0.764	-5.484	
2	['pop']	…	0.563	0.664	-5.044	
3	['pop', 'uk pop']	…	0.808	0.897	-3.712	

```
4           ['lgbtq+ hip hop', 'pop rap']  …        0.736  0.704   -7.409

  Speechiness Acousticness Liveness    Tempo Duration (ms) Valence  Chord
0      0.0504        0.127    0.359  134.002        211560   0.589      B
1      0.0483       0.0383    0.103  169.928        141806   0.478  C#/Db
2       0.154        0.335   0.0849  166.928        178147   0.688      A
3      0.0348       0.0469    0.364  126.026        231041   0.591      B
4      0.0615       0.0203   0.0501  149.995        212000   0.894  D#/Eb

[5 rows x 23 columns]
```

3.2 Tail

3.3 Shape

[8]:
```
df_1.shape
```

[8]:
```
(1556, 23)
```

3.4 columns

[9]:
```
df_1.columns
```

[9]:
```
Index(['Index', 'Highest Charting Position', 'Number of Times Charted',
       'Week of Highest Charting', 'Song Name', 'Streams', 'Artist',
       'Artist Followers', 'Song ID', 'Genre', 'Release Date', 'Weeks Charted',
       'Popularity', 'Danceability', 'Energy', 'Loudness', 'Speechiness',
       'Acousticness', 'Liveness', 'Tempo', 'Duration (ms)', 'Valence',
       'Chord'],
      dtype='object')
```

3.5 Dtypes

[10]:
```
df_1.dtypes
```

[10]:
```
Index                          int64
Highest Charting Position      int64
Number of Times Charted        int64
Week of Highest Charting      object
Song Name                     object
Streams                       object
Artist                        object
Artist Followers              object
Song ID                       object
Genre                         object
Release Date                  object
Weeks Charted                 object
Popularity                    object
```

```
Danceability                object
Energy                      object
Loudness                    object
Speechiness                 object
Acousticness                object
Liveness                    object
Tempo                       object
Duration (ms)               object
Valence                     object
Chord                       object
dtype: object
```

3.6 Describe

[11]:
```
df_1.describe()
```

[11]:

	Index	Highest Charting Position	Number of Times Charted
count	1556.000000	1556.000000	1556.000000
mean	778.500000	87.744216	10.668380
std	449.322824	58.147225	16.360546
min	1.000000	1.000000	1.000000
25%	389.750000	37.000000	1.000000
50%	778.500000	80.000000	4.000000
75%	1167.250000	137.000000	12.000000
max	1556.000000	200.000000	142.000000

3.7 Isnull Sum

[12]:
```
df_1.isnull().sum()
```

[12]:
```
Index                       0
Highest Charting Position   0
Number of Times Charted     0
Week of Highest Charting    0
Song Name                   0
Streams                     0
Artist                      0
Artist Followers            0
Song ID                     0
Genre                       0
Release Date                0
Weeks Charted               0
Popularity                  0
Danceability                0
Energy                      0
Loudness                    0
Speechiness                 0
```

```
Acousticness                0
Liveness                    0
Tempo                       0
Duration (ms)               0
Valence                     0
Chord                       0
dtype: int64
```

3.8 Isna Sum

[13]:
```
df_1.isna().sum()
```

[13]:
```
Index                       0
Highest Charting Position   0
Number of Times Charted     0
Week of Highest Charting    0
Song Name                   0
Streams                     0
Artist                      0
Artist Followers            0
Song ID                     0
Genre                       0
Release Date                0
Weeks Charted               0
Popularity                  0
Danceability                0
Energy                      0
Loudness                    0
Speechiness                 0
Acousticness                0
Liveness                    0
Tempo                       0
Duration (ms)               0
Valence                     0
Chord                       0
dtype: int64
```

3.9 unique values

[14]:
```
df_1.count('rows').unique().sum()
```

[14]:
```
np.int64(1556)
```

[15]:
```
df_1.count('columns')
```

[15]:
```
0       23
1       23
```

```
2       23
3       23
4       23
        ..
1551    23
1552    23
1553    23
1554    23
1555    23
Length: 1556, dtype: int64
```

3.10 Sort_values

[16]:
```
df_1.sort_values(by = ['Popularity'], ascending = False).head(10)
```

[16]:

	Index	Highest Charting Position	Number of Times Charted	Week of Highest Charting	Song Name	Streams	Artist	Artist Followers	Song ID
1	2	2	3	2021-07-23--2021-07-30	STAY (with Justin Bieber)	47,248,719	The Kid LAROI	2230022	5HCyWlXZPP0y6Gqq8TgA20
2	3	1	11	2021-06-25--2021-07-02	good 4 u	40,162,559	Olivia Rodrigo	6266514	4ZtFanR9U6ndgddUvNcjcG
3	4	3	5	2021-07-02--2021-07-09	Bad Habits	37,799,456	Ed Sheeran	83293380	6PQ88X9TkUIAUIZJHW2upE
5	6	1	18	2021-05-07--2021-05-14	MONTERO (Call Me By Your Name)	30,071,134	Lil Nas X	5473565	67BtfxlNbhBmCDR2L2l8qd
4	5	5	1	2021-07-23--2021-07-30	INDUSTRY BABY (feat. Jack Harlow)	33,948,454	Lil Nas X	5473565	27NovPIUIRrOZoCHxABJwK
8	9	3	8	2021-06-18--2021-06-25	Yonaguni	25,030,128	Bad Bunny	36142273	2JPLbjOn0wPCngEot2STUS
14	15	2	10	2021-05-21--2021-05-28	Butter	19,985,713	BTS	37106176	2bgTY4UwhfBYhGT4HUYStN
7	8	2	10	2021-06-18--2021-06-25	Todo De Ti	26,951,613			
9	10	8	10	2021-07-02--2021-07-09	I WANNA BE YOUR SLAVE	24,551,591			
11	12	9	9	2021-07-02--2021-07-09	Qué Más Pues?	22,405,111			

```
7            Rauw Alejandro          6080597  4fSIb4hdOQ151TILNsSEaF
9                 Måneskin           3377762  4pt5fDVTg5GhEvEtlz9dKk
11  J Balvin, Maria Becerra          29051363  6hf0RpxTbOprT5nnwzkk8e

                                          Genre  … Danceability Energy  \
1                          ['australian hip hop']  …        0.591  0.764
2                                       ['pop']  …        0.563  0.664
3                             ['pop', 'uk pop']  …        0.808  0.897
5                   ['lgbtq+ hip hop', 'pop rap']  …         0.61  0.508
4                   ['lgbtq+ hip hop', 'pop rap']  …        0.736  0.704
8            ['latin', 'reggaeton', 'trap latino']  …        0.644  0.648
14                    ['k-pop', 'k-pop boy group']  …        0.759  0.459
7               ['puerto rican pop', 'trap latino']  …         0.78  0.718
9           ['indie rock italiano', 'italian pop']  …         0.75  0.608
11  ['latin', 'reggaeton', 'reggaeton colombiano']  …        0.891  0.819

   Loudness Speechiness Acousticness Liveness    Tempo Duration (ms) Valence  \
1    -5.484      0.0483       0.0383    0.103  169.928        141806   0.478
2    -5.044       0.154        0.335   0.0849  166.928        178147   0.688
3    -3.712      0.0348       0.0469    0.364  126.026        231041   0.591
5    -6.682       0.152        0.297    0.384  178.818        137876   0.758
4    -7.409      0.0615       0.0203   0.0501  149.995        212000   0.894
8    -4.601       0.118        0.276    0.135  179.951        206710    0.44
14   -5.187      0.0948      0.00323   0.0906  109.997        164442   0.695
7    -3.605      0.0506         0.31   0.0932  127.949        199604   0.342
9    -4.008      0.0387      0.00165    0.178  132.507        173347   0.958
11   -3.964       0.106       0.0261    0.173  101.968        217773   0.768

    Chord
1   C#/Db
2       A
3       B
5   G#/Ab
4   D#/Eb
8   C#/Db
14  G#/Ab
7   D#/Eb
9   C#/Db
11  G#/Ab

[10 rows x 23 columns]
```

4 Data Cleaning and Feature Engineering

4.1 New copy of dataframe

[17]:
```
df_cleaning = df_1.copy()
df_cleaning
```

[17]:

	Index	Highest Charting Position	Number of Times Charted	\
0	1	1	8	
1	2	2	3	
2	3	1	11	
3	4	3	5	
4	5	5	1	
…	…	…	…	
1551	1552	195	1	
1552	1553	196	1	
1553	1554	197	1	
1554	1555	198	1	
1555	1556	199	1	

	Week of Highest Charting	Song Name	Streams	\
0	2021-07-23--2021-07-30	Beggin'	48,633,449	
1	2021-07-23--2021-07-30	STAY (with Justin Bieber)	47,248,719	
2	2021-06-25--2021-07-02	good 4 u	40,162,559	
3	2021-07-02--2021-07-09	Bad Habits	37,799,456	
4	2021-07-23--2021-07-30	INDUSTRY BABY (feat. Jack Harlow)	33,948,454	
…	…	…	…	
1551	2019-12-27--2020-01-03	New Rules	4,630,675	
1552	2019-12-27--2020-01-03	Cheirosa - Ao Vivo	4,623,030	
1553	2019-12-27--2020-01-03	Havana (feat. Young Thug)	4,620,876	
1554	2019-12-27--2020-01-03	Surtada - Remix Brega Funk	4,607,385	
1555	2019-12-27--2020-01-03	Lover (Remix) [feat. Shawn Mendes]	4,595,450	

	Artist	Artist Followers	Song ID	\
0	Måneskin	3377762	3Wrjm47oTz2sjIgck11l5e	
1	The Kid LAROI	2230022	5HCyWlXZPP0y6Gqq8TgA20	
2	Olivia Rodrigo	6266514	4ZtFanR9U6ndgddUvNcjcG	
3	Ed Sheeran	83293380	6PQ88X9TkUIAUIZJHW2upE	
4	Lil Nas X	5473565	27NovPIUIRrOZoCHxABJwK	
…	…	…	…	
1551	Dua Lipa	27167675	2ekn2ttSfGqwhhate0LSR0	
1552	Jorge & Mateus	15019109	2PWjKmjyTZeDpmOUa3a5da	
1553	Camila Cabello	22698747	1rfofaqEpACxVEHIZBJe6W	
1554	Dadá Boladão, Tati Zaqui, OIK	208630	5F8ffc8KWKNawllr5WsW0r	
1555	Taylor Swift	42227614	3i9UVldZOE0aD0JnyfAZZ0	

	Genre	…	Danceability	\
0	['indie rock italiano', 'italian pop']	…	0.714	

```
1                                ['australian hip hop']  …        0.591
2                                             ['pop']  …        0.563
3                                   ['pop', 'uk pop']  …        0.808
4                          ['lgbtq+ hip hop', 'pop rap']  …        0.736
…                                                    …  …          …
1551                        ['dance pop', 'pop', 'uk pop']  …        0.762
1552             ['sertanejo', 'sertanejo universitario']  …        0.528
1553  ['dance pop', 'electropop', 'pop', 'post-teen …  …        0.765
1554                        ['brega funk', 'funk carioca']  …        0.832
1555                              ['pop', 'post-teen pop']  …        0.448

     Energy Loudness Speechiness Acousticness Liveness    Tempo Duration (ms)  \
0       0.8   -4.808      0.0504        0.127    0.359  134.002        211560
1     0.764   -5.484      0.0483       0.0383    0.103  169.928        141806
2     0.664   -5.044       0.154        0.335   0.0849  166.928        178147
3     0.897   -3.712      0.0348       0.0469    0.364  126.026        231041
4     0.704   -7.409      0.0615       0.0203   0.0501  149.995        212000
…     …       …           …          …       …        …           …
1551    0.7   -6.021      0.0694      0.00261    0.153  116.073        209320
1552   0.87   -3.123      0.0851         0.24    0.333   152.37        181930
1553  0.523   -4.333        0.03        0.184    0.132  104.988        217307
1554   0.55   -7.026      0.0587        0.249    0.182  154.064        152784
1555  0.603   -7.176       0.064        0.433   0.0862  205.272        221307

     Valence  Chord
0      0.589      B
1      0.478  C#/Db
2      0.688      A
3      0.591      B
4      0.894  D#/Eb
…      …    …
1551   0.608      A
1552   0.714      B
1553   0.394      D
1554   0.881      F
1555   0.422      G

[1556 rows x 23 columns]
```

4.2 drop Index

[18]:
```
df_cleaning.drop('Index', axis = 1, inplace = True)
#i
```

[19]:
```
df_cleaning.transpose()
```

[19]:

	0	1
Highest Charting Position	1	2
Number of Times Charted	8	3
Week of Highest Charting	2021-07-23--2021-07-30	2021-07-23--2021-07-30
Song Name	Beggin'	STAY (with Justin Bieber)
Streams	48,633,449	47,248,719
Artist	Måneskin	The Kid LAROI
Artist Followers	3377762	2230022
Song ID	3Wrjm47oTz2sjIgck11l5e	5HCyWlXZPP0y6Gqq8TgA20
Genre	['indie rock italiano', 'italian pop']	['australian hip hop']
Release Date	2017-12-08	2021-07-09
Weeks Charted	2021-07-23--2021-07-30\n2021-07-16--2021-07-23…	2021-07-23--2021-07-30\n2021-07-16--2021-07-23…
Popularity	100	99
Danceability	0.714	0.591
Energy	0.8	0.764
Loudness	-4.808	-5.484
Speechiness	0.0504	0.0483
Acousticness	0.127	0.0383
Liveness	0.359	0.103
Tempo	134.002	169.928
Duration (ms)	211560	141806
Valence	0.589	0.478
Chord	B	C#/Db

	2 \
Highest Charting Position	1
Number of Times Charted	11
Week of Highest Charting	2021-06-25--2021-07-02
Song Name	good 4 u
Streams	40,162,559
Artist	Olivia Rodrigo
Artist Followers	6266514
Song ID	4ZtFanR9U6ndgddUvNcjcG
Genre	['pop']
Release Date	2021-05-21
Weeks Charted	2021-07-23--2021-07-30\n2021-07-16--2021-07-23…
Popularity	99
Danceability	0.563
Energy	0.664
Loudness	-5.044
Speechiness	0.154
Acousticness	0.335
Liveness	0.0849
Tempo	166.928
Duration (ms)	178147
Valence	0.688
Chord	A

	3 \
Highest Charting Position	3
Number of Times Charted	5
Week of Highest Charting	2021-07-02--2021-07-09
Song Name	Bad Habits
Streams	37,799,456
Artist	Ed Sheeran
Artist Followers	83293380
Song ID	6PQ88X9TkUIAUIZJHW2upE
Genre	['pop', 'uk pop']
Release Date	2021-06-25
Weeks Charted	2021-07-23--2021-07-30\n2021-07-16--2021-07-23…
Popularity	98
Danceability	0.808
Energy	0.897
Loudness	-3.712
Speechiness	0.0348
Acousticness	0.0469
Liveness	0.364
Tempo	126.026
Duration (ms)	231041
Valence	0.591

Chord		B
	4	\
Highest Charting Position	5	
Number of Times Charted	1	
Week of Highest Charting	2021-07-23--2021-07-30	
Song Name	INDUSTRY BABY (feat. Jack Harlow)	
Streams	33,948,454	
Artist	Lil Nas X	
Artist Followers	5473565	
Song ID	27NovPIUIRrOZoCHxABJwK	
Genre	['lgbtq+ hip hop', 'pop rap']	
Release Date	2021-07-23	
Weeks Charted	2021-07-23--2021-07-30	
Popularity	96	
Danceability	0.736	
Energy	0.704	
Loudness	-7.409	
Speechiness	0.0615	
Acousticness	0.0203	
Liveness	0.0501	
Tempo	149.995	
Duration (ms)	212000	
Valence	0.894	
Chord	D#/Eb	

	5	\
Highest Charting Position	1	
Number of Times Charted	18	
Week of Highest Charting	2021-05-07--2021-05-14	
Song Name	MONTERO (Call Me By Your Name)	
Streams	30,071,134	
Artist	Lil Nas X	
Artist Followers	5473565	
Song ID	67BtfxlNbhBmCDR2L2l8qd	
Genre	['lgbtq+ hip hop', 'pop rap']	
Release Date	2021-03-31	
Weeks Charted	2021-07-23--2021-07-30\n2021-07-16--2021-07-23…	
Popularity	97	
Danceability	0.61	
Energy	0.508	
Loudness	-6.682	
Speechiness	0.152	
Acousticness	0.297	
Liveness	0.384	
Tempo	178.818	
Duration (ms)	137876	

Valence	0.758
Chord	G#/Ab

	6 \
Highest Charting Position	3
Number of Times Charted	16
Week of Highest Charting	2021-05-14--2021-05-21
Song Name	Kiss Me More (feat. SZA)
Streams	29,356,736
Artist	Doja Cat
Artist Followers	8640063
Song ID	748mdHapucXQri7IAO8yFK
Genre	['dance pop', 'pop']
Release Date	2021-04-09
Weeks Charted	2021-07-23--2021-07-30\n2021-07-16--2021-07-23…
Popularity	94
Danceability	0.762
Energy	0.701
Loudness	-3.541
Speechiness	0.0286
Acousticness	0.235
Liveness	0.123
Tempo	110.968
Duration (ms)	208867
Valence	0.742
Chord	G#/Ab

	7 \
Highest Charting Position	2
Number of Times Charted	10
Week of Highest Charting	2021-06-18--2021-06-25
Song Name	Todo De Ti
Streams	26,951,613
Artist	Rauw Alejandro
Artist Followers	6080597
Song ID	4fSIb4hdOQ151TILNsSEaF
Genre	['puerto rican pop', 'trap latino']
Release Date	2021-05-20
Weeks Charted	2021-07-23--2021-07-30\n2021-07-16--2021-07-23…
Popularity	95
Danceability	0.78
Energy	0.718
Loudness	-3.605
Speechiness	0.0506
Acousticness	0.31
Liveness	0.0932
Tempo	127.949

Duration (ms)	199604
Valence	0.342
Chord	D#/Eb

	8 \
Highest Charting Position	3
Number of Times Charted	8
Week of Highest Charting	2021-06-18--2021-06-25
Song Name	Yonaguni
Streams	25,030,128
Artist	Bad Bunny
Artist Followers	36142273
Song ID	2JPLbjOn0wPCngEot2STUS
Genre	['latin', 'reggaeton', 'trap latino']
Release Date	2021-06-04
Weeks Charted	2021-07-23--2021-07-30\n2021-07-16--2021-07-23…
Popularity	96
Danceability	0.644
Energy	0.648
Loudness	-4.601
Speechiness	0.118
Acousticness	0.276
Liveness	0.135
Tempo	179.951
Duration (ms)	206710
Valence	0.44
Chord	C#/Db

	9 \
Highest Charting Position	8
Number of Times Charted	10
Week of Highest Charting	2021-07-02--2021-07-09
Song Name	I WANNA BE YOUR SLAVE
Streams	24,551,591
Artist	Måneskin
Artist Followers	3377762
Song ID	4pt5fDVTg5GhEvEtlz9dKk
Genre	['indie rock italiano', 'italian pop']
Release Date	2021-03-19
Weeks Charted	2021-07-23--2021-07-30\n2021-07-16--2021-07-23…
Popularity	95
Danceability	0.75
Energy	0.608
Loudness	-4.008
Speechiness	0.0387
Acousticness	0.00165
Liveness	0.178

Tempo			132.507
Duration (ms)			173347
Valence			0.958
Chord			C#/Db

	…	1546	\
Highest Charting Position	…	143	
Number of Times Charted	…	1	
Week of Highest Charting	…	2019-12-27--2020-01-03	
Song Name	…	JACKBOYS	
Streams	…	5,363,493	
Artist	…	JACKBOYS	
Artist Followers	…	437907	
Song ID	…	62zKJrpbLxz6InR3tGyr7o	
Genre	…	['rap', 'trap']	
Release Date	…	2019-12-27	
Weeks Charted	…	2019-12-27--2020-01-03	
Popularity	…	56	
Danceability	…	0.413	
Energy	…	0.13	
Loudness	…	-25.166	
Speechiness	…	0.0336	
Acousticness	…	0.9	
Liveness	…	0.111	
Tempo	…	123.342	
Duration (ms)	…	46837	
Valence	…	0.0676	
Chord	…	C	

		1547 \
Highest Charting Position		156
Number of Times Charted		1
Week of Highest Charting		2019-12-27--2020-01-03
Song Name		Combatchy (feat. MC Rebecca)
Streams		5,149,797
Artist		Anitta, Lexa, Luísa Sonza
Artist Followers		10741972
Song ID		2bPtwnrpFNEe8N7Q85kLHw
Genre	['funk carioca', 'funk pop', 'pagode baiano', …	
Release Date		2019-11-20
Weeks Charted		2019-12-27--2020-01-03
Popularity		64
Danceability		0.826
Energy		0.73
Loudness		-3.032
Speechiness		0.0809
Acousticness		0.383

Liveness		0.0197
Tempo		150.134
Duration (ms)		157600
Valence		0.605
Chord		C#/Db

	1548	\
Highest Charting Position	178	
Number of Times Charted	1	
Week of Highest Charting	2019-12-27--2020-01-03	
Song Name	Old Town Road	
Streams	4,852,004	
Artist	Lil Nas X	
Artist Followers	5488666	
Song ID	2YpeDb67231RjR0MgVLzsG	
Genre	['lgbtq+ hip hop', 'pop rap']	
Release Date	2019-06-21	
Weeks Charted	2019-12-27--2020-01-03	
Popularity	81	
Danceability	0.878	
Energy	0.619	
Loudness	-5.56	
Speechiness	0.102	
Acousticness	0.0533	
Liveness	0.113	
Tempo	136.041	
Duration (ms)	157067	
Valence	0.639	
Chord	F#/Gb	

	1549	\
Highest Charting Position	187	
Number of Times Charted	1	
Week of Highest Charting	2019-12-27--2020-01-03	
Song Name	Let Me Know (I Wonder Why Freestyle)	
Streams	4,701,532	
Artist	Juice WRLD	
Artist Followers	19102888	
Song ID	3wwo0bJvDSorOpNfzEkfXx	
Genre	['chicago rap', 'melodic rap']	
Release Date	2019-12-07	
Weeks Charted	2019-12-27--2020-01-03	
Popularity	76	
Danceability	0.635	
Energy	0.537	
Loudness	-7.895	
Speechiness	0.0832	

Acousticness	0.172		
Liveness	0.418		
Tempo	125.028		
Duration (ms)	215381		
Valence	0.383		
Chord	G		

	1550	\
Highest Charting Position	190	
Number of Times Charted	1	
Week of Highest Charting	2019-12-27--2020-01-03	
Song Name	Ne reviens pas	
Streams	4,676,857	
Artist	Gradur, Heuss L'enfoiré	
Artist Followers	1390813	
Song ID	4TnFANpjVwVKWzkxNzIyFH	
Genre	['francoton', 'french hip hop', 'pop urbaine',…	
Release Date	2019-11-29	
Weeks Charted	2019-12-27--2020-01-03	
Popularity	62	
Danceability	0.932	
Energy	0.778	
Loudness	-3.384	
Speechiness	0.0638	
Acousticness	0.212	
Liveness	0.168	
Tempo	124.996	
Duration (ms)	188613	
Valence	0.933	
Chord	A#/Bb	

	1551	\
Highest Charting Position	195	
Number of Times Charted	1	
Week of Highest Charting	2019-12-27--2020-01-03	
Song Name	New Rules	
Streams	4,630,675	
Artist	Dua Lipa	
Artist Followers	27167675	
Song ID	2ekn2ttSfGqwhhate0LSR0	
Genre	['dance pop', 'pop', 'uk pop']	
Release Date	2017-06-02	
Weeks Charted	2019-12-27--2020-01-03	
Popularity	79	
Danceability	0.762	
Energy	0.7	
Loudness	-6.021	

Speechiness	0.0694
Acousticness	0.00261
Liveness	0.153
Tempo	116.073
Duration (ms)	209320
Valence	0.608
Chord	A

	1552 \
Highest Charting Position	196
Number of Times Charted	1
Week of Highest Charting	2019-12-27--2020-01-03
Song Name	Cheirosa - Ao Vivo
Streams	4,623,030
Artist	Jorge & Mateus
Artist Followers	15019109
Song ID	2PWjKmjyTZeDpmOUa3a5da
Genre	['sertanejo', 'sertanejo universitario']
Release Date	2019-10-11
Weeks Charted	2019-12-27--2020-01-03
Popularity	66
Danceability	0.528
Energy	0.87
Loudness	-3.123
Speechiness	0.0851
Acousticness	0.24
Liveness	0.333
Tempo	152.37
Duration (ms)	181930
Valence	0.714
Chord	B

	1553 \
Highest Charting Position	197
Number of Times Charted	1
Week of Highest Charting	2019-12-27--2020-01-03
Song Name	Havana (feat. Young Thug)
Streams	4,620,876
Artist	Camila Cabello
Artist Followers	22698747
Song ID	1rfofaqEpACxVEHIZBJe6W
Genre	['dance pop', 'electropop', 'pop', 'post-teen …
Release Date	2018-01-12
Weeks Charted	2019-12-27--2020-01-03
Popularity	81
Danceability	0.765
Energy	0.523

Loudness	-4.333
Speechiness	0.03
Acousticness	0.184
Liveness	0.132
Tempo	104.988
Duration (ms)	217307
Valence	0.394
Chord	D

	1554 \
Highest Charting Position	198
Number of Times Charted	1
Week of Highest Charting	2019-12-27--2020-01-03
Song Name	Surtada - Remix Brega Funk
Streams	4,607,385
Artist	Dadá Boladão, Tati Zaqui, OIK
Artist Followers	208630
Song ID	5F8ffc8KWKNawllr5WsW0r
Genre	['brega funk', 'funk carioca']
Release Date	2019-09-25
Weeks Charted	2019-12-27--2020-01-03
Popularity	60
Danceability	0.832
Energy	0.55
Loudness	-7.026
Speechiness	0.0587
Acousticness	0.249
Liveness	0.182
Tempo	154.064
Duration (ms)	152784
Valence	0.881
Chord	F

	1555
Highest Charting Position	199
Number of Times Charted	1
Week of Highest Charting	2019-12-27--2020-01-03
Song Name	Lover (Remix) [feat. Shawn Mendes]
Streams	4,595,450
Artist	Taylor Swift
Artist Followers	42227614
Song ID	3i9UVldZOE0aD0JnyfAZZ0
Genre	['pop', 'post-teen pop']
Release Date	2019-11-13
Weeks Charted	2019-12-27--2020-01-03
Popularity	70
Danceability	0.448

```
Energy                                                  0.603
Loudness                                               -7.176
Speechiness                                             0.064
Acousticness                                            0.433
Liveness                                               0.0862
Tempo                                                 205.272
Duration (ms)                                          221307
Valence                                                 0.422
Chord                                                       G

[22 rows x 1556 columns]
```

4.3 Convert object columns with numbers to float64

[20]:
```
# List of columns to convert
columns_to_convert = ['Artist Followers', 'Streams','Popularity',␣
 ↪'Danceability', 'Energy', 'Loudness',
                      'Speechiness', 'Acousticness', 'Liveness', 'Tempo',␣
 ↪'Duration (ms)', 'Valence']

df_1[columns_to_convert] = df_1[columns_to_convert].apply(pd.to_numeric,␣
 ↪errors='coerce')
```

[21]:
```
df_1.dtypes
```

[21]:
```
Index                          int64
Highest Charting Position      int64
Number of Times Charted        int64
Week of Highest Charting      object
Song Name                     object
Streams                      float64
Artist                        object
Artist Followers             float64
Song ID                       object
Genre                         object
Release Date                  object
Weeks Charted                 object
Popularity                   float64
Danceability                 float64
Energy                       float64
Loudness                     float64
Speechiness                  float64
Acousticness                 float64
Liveness                     float64
Tempo                        float64
Duration (ms)                float64
Valence                      float64
```

```
Chord                          object
dtype: object
```

5 Data Cleaning Continued: Prepare DataFrame for Modeling and Training

[22]:
```
df_1 = df_1.drop("Index", axis = 1)
```

[23]:
```
df_1
```

[23]:

	Highest Charting Position	Number of Times Charted	Week of Highest Charting	Song Name	Streams	Artist	Artist Followers	Song ID
0	1	8	2021-07-23--2021-07-30	Beggin'	NaN	Måneskin	3377762.0	3Wrjm47oTz2sjIgck11l5e
1	2	3	2021-07-23--2021-07-30	STAY (with Justin Bieber)	NaN	The Kid LAROI	2230022.0	5HCyWlXZPP0y6Gqq8TgA20
2	1	11	2021-06-25--2021-07-02	good 4 u	NaN	Olivia Rodrigo	6266514.0	4ZtFanR9U6ndgddUvNcjcG
3	3	5	2021-07-02--2021-07-09	Bad Habits	NaN	Ed Sheeran	83293380.0	6PQ88X9TkUIAUIZJHW2upE
4	5	1	2021-07-23--2021-07-30	INDUSTRY BABY (feat. Jack Harlow)	NaN	Lil Nas X	5473565.0	27NovPIUIRrOZoCHxABJwK
…	…	…	…	…	…	…	…	…
1551	195	1	2019-12-27--2020-01-03	New Rules	NaN	Dua Lipa	27167675.0	2ekn2ttSfGqwhhate0LSR0
1552	196	1	2019-12-27--2020-01-03	Cheirosa - Ao Vivo	NaN	Jorge & Mateus	15019109.0	2PWjKmjyTZeDpmOUa3a5da
1553	197	1	2019-12-27--2020-01-03	Havana (feat. Young Thug)	NaN	Camila Cabello	22698747.0	1rfofaqEpACxVEHIZBJe6W
1554	198	1	2019-12-27--2020-01-03	Surtada - Remix Brega Funk	NaN			
1555	199	1	2019-12-27--2020-01-03	Lover (Remix) [feat. Shawn Mendes]	NaN			

```
1554  Dadá Boladão, Tati Zaqui, OIK          208630.0  5F8ffc8KWKNawllr5WsW0r
1555                    Taylor Swift        42227614.0  3i9UVldZOE0aD0JnyfAZZ0

                                                 Genre Release Date  …  \
0                ['indie rock italiano', 'italian pop']   2017-12-08  …
1                                  ['australian hip hop']   2021-07-09  …
2                                                 ['pop']   2021-05-21  …
3                                     ['pop', 'uk pop']   2021-06-25  …
4                       ['lgbtq+ hip hop', 'pop rap']   2021-07-23  …
…                                                    …          …  …
1551                    ['dance pop', 'pop', 'uk pop']   2017-06-02  …
1552           ['sertanejo', 'sertanejo universitario']   2019-10-11  …
1553  ['dance pop', 'electropop', 'pop', 'post-teen …   2018-01-12  …
1554                    ['brega funk', 'funk carioca']   2019-09-25  …
1555                          ['pop', 'post-teen pop']   2019-11-13  …

     Danceability  Energy  Loudness  Speechiness  Acousticness  Liveness  \
0           0.714   0.800    -4.808       0.0504       0.12700    0.3590
1           0.591   0.764    -5.484       0.0483       0.03830    0.1030
2           0.563   0.664    -5.044       0.1540       0.33500    0.0849
3           0.808   0.897    -3.712       0.0348       0.04690    0.3640
4           0.736   0.704    -7.409       0.0615       0.02030    0.0501
…           …     …        …          …           …       …
1551        0.762   0.700    -6.021       0.0694       0.00261    0.1530
1552        0.528   0.870    -3.123       0.0851       0.24000    0.3330
1553        0.765   0.523    -4.333       0.0300       0.18400    0.1320
1554        0.832   0.550    -7.026       0.0587       0.24900    0.1820
1555        0.448   0.603    -7.176       0.0640       0.43300    0.0862

        Tempo  Duration (ms)  Valence  Chord
0     134.002       211560.0    0.589      B
1     169.928       141806.0    0.478  C#/Db
2     166.928       178147.0    0.688      A
3     126.026       231041.0    0.591      B
4     149.995       212000.0    0.894  D#/Eb
…       …            …      …    …
1551  116.073       209320.0    0.608      A
1552  152.370       181930.0    0.714      B
1553  104.988       217307.0    0.394      D
1554  154.064       152784.0    0.881      F
1555  205.272       221307.0    0.422      G

[1556 rows x 22 columns]
```

[24]:
```
df_clean_2 = df_1.copy()
```

5.1 Identify Object Columns & Drop them

[25]:
```
object_columns = df_clean_2.select_dtypes(include=['object']).columns
df_clean_2 = df_clean_2.drop(columns=object_columns)
```

[26]:
```
df_clean_2.isnull().sum()
```

[26]:
```
Highest Charting Position       0
Number of Times Charted         0
Streams                      1556
Artist Followers               11
Popularity                     11
Danceability                   11
Energy                         11
Loudness                       11
Speechiness                    11
Acousticness                   11
Liveness                       11
Tempo                          11
Duration (ms)                  11
Valence                        11
dtype: int64
```

[27]:
```
df_clean_2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1556 entries, 0 to 1555
Data columns (total 14 columns):
 #   Column                     Non-Null Count  Dtype
---  ------                     --------------  -----
 0   Highest Charting Position  1556 non-null   int64
 1   Number of Times Charted    1556 non-null   int64
 2   Streams                    0 non-null      float64
 3   Artist Followers           1545 non-null   float64
 4   Popularity                 1545 non-null   float64
 5   Danceability               1545 non-null   float64
 6   Energy                     1545 non-null   float64
 7   Loudness                   1545 non-null   float64
 8   Speechiness                1545 non-null   float64
 9   Acousticness               1545 non-null   float64
 10  Liveness                   1545 non-null   float64
 11  Tempo                      1545 non-null   float64
 12  Duration (ms)              1545 non-null   float64
 13  Valence                    1545 non-null   float64
dtypes: float64(12), int64(2)
memory usage: 170.3 KB
```

5.2 Drop Streams Column (essentially empty)

[28]:
```
df_clean_2.drop('Streams', axis = 1, inplace = True)
```

[29]:
```
df_clean_2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1556 entries, 0 to 1555
Data columns (total 13 columns):
 #   Column                     Non-Null Count  Dtype
---  ------                     --------------  -----
 0   Highest Charting Position  1556 non-null   int64
 1   Number of Times Charted    1556 non-null   int64
 2   Artist Followers           1545 non-null   float64
 3   Popularity                 1545 non-null   float64
 4   Danceability               1545 non-null   float64
 5   Energy                     1545 non-null   float64
 6   Loudness                   1545 non-null   float64
 7   Speechiness                1545 non-null   float64
 8   Acousticness               1545 non-null   float64
 9   Liveness                   1545 non-null   float64
 10  Tempo                      1545 non-null   float64
 11  Duration (ms)              1545 non-null   float64
 12  Valence                    1545 non-null   float64
dtypes: float64(11), int64(2)
memory usage: 158.2 KB
```

5.3 Get means and replace null values with mean per column

[30]:
```
df_clean_2.isna().sum()
```

[30]:
```
Highest Charting Position     0
Number of Times Charted       0
Artist Followers             11
Popularity                   11
Danceability                 11
Energy                       11
Loudness                     11
Speechiness                  11
Acousticness                 11
Liveness                     11
Tempo                        11
Duration (ms)                11
Valence                      11
dtype: int64
```

[31]:
```
null_columns = df_clean_2.columns[df_clean_2.isnull().any()].tolist()
print("Columns with null values:")
```

```
null_columns
```

```
Columns with null values:
```

[31]:
```
['Artist Followers',
 'Popularity',
 'Danceability',
 'Energy',
 'Loudness',
 'Speechiness',
 'Acousticness',
 'Liveness',
 'Tempo',
 'Duration (ms)',
 'Valence']
```

[32]:
```
for col in null_columns:
    #Calculate the mean, exluding NaN values
    mean= df_clean_2[col].mean(skipna=True)

    #replace NaNs with the mean per column
    df_clean_2[col] = df_clean_2[col].fillna(mean)
```

[33]:
```
print("\nNull value count after replacement:")
print(df_clean_2.isnull().sum())
```

```
Null value count after replacement:
Highest Charting Position    0
Number of Times Charted      0
Artist Followers             0
Popularity                   0
Danceability                 0
Energy                       0
Loudness                     0
Speechiness                  0
Acousticness                 0
Liveness                     0
Tempo                        0
Duration (ms)                0
Valence                      0
dtype: int64
```

[34]:
```
df_clean_2.dtypes
```

[34]:
```
Highest Charting Position      int64
Number of Times Charted        int64
Artist Followers             float64
```

```
Popularity                  float64
Danceability                float64
Energy                      float64
Loudness                    float64
Speechiness                 float64
Acousticness                float64
Liveness                    float64
Tempo                       float64
Duration (ms)               float64
Valence                     float64
dtype: object
```

5.4 Drop columns that have no relation to target = "Popularity"

[35]:
```
df_clean_2.drop('Highest Charting Position', axis = 1, inplace = True)
```

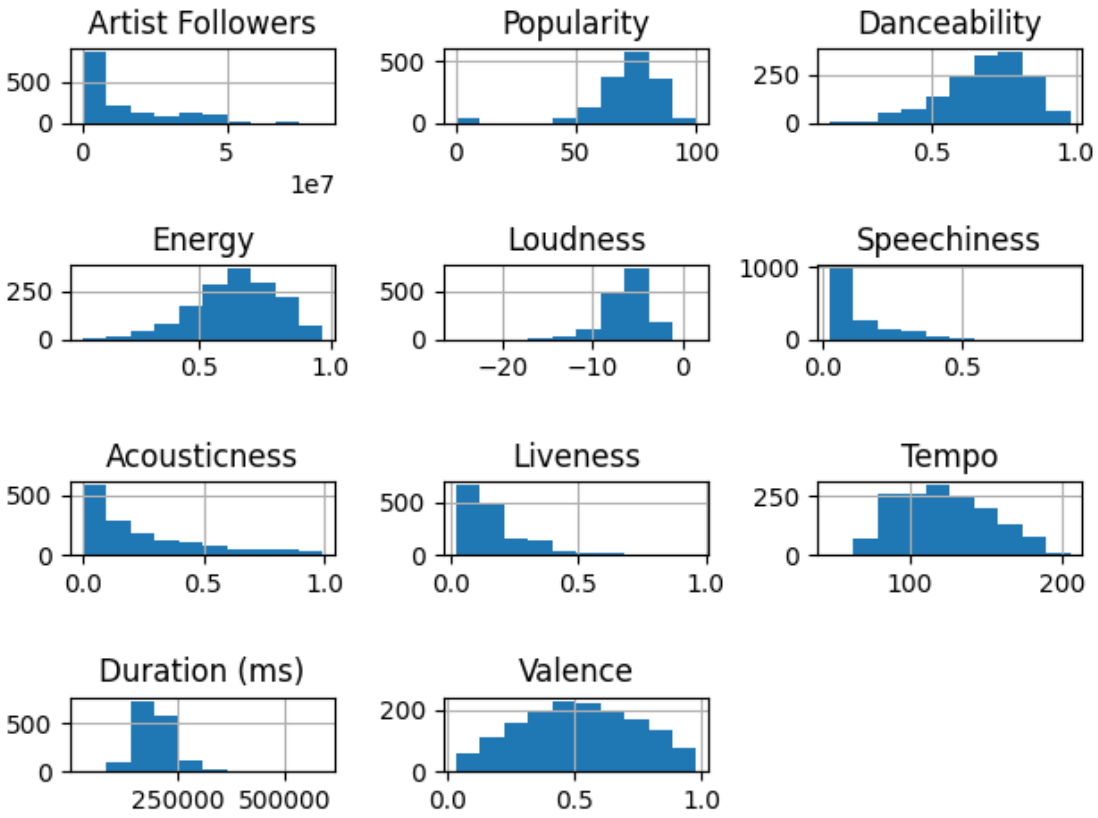
[36]:
```
df_clean_2.drop('Number of Times Charted', axis = 1, inplace = True)
```

[37]:
```
# df_clean_2.drop('Artist Followers', axis = 1, inplace = True)
```

[38]:
```
df_scaling = df_clean_2.copy()
```

[39]:
```
df_scaling.hist()
plt.tight_layout()
plt.show
```

[39]:
```
<function matplotlib.pyplot.show(close=None, block=None)>
```



6 Data Scaling

6.1 Data Scaling (standard scaler)

6.1.1 Setup standard scaled training and testing data

[40]:
```
df_3_std = df_scaling.copy()
```

[41]:
```
x1 = df_3_std.drop(['Popularity'], axis=1)
y1 = df_3_std['Popularity']

X_train_1, X_test_1, y_train_1, y_test_1 = train_test_split(x1, y1, test_size=0.
↪2)
```

[42]:
```
scaler = StandardScaler()
X_train_std = scaler.fit_transform(X_train_1)
X_test_std = scaler.transform(X_test_1)
```

[43]:
```
print("Before scaling:")
print(X_train_1.describe())
```

```
print("\nAfter scaling:")
print(pd.DataFrame(X_train_std).describe())
```

```
Before scaling:
       Artist Followers  Danceability       Energy     Loudness  Speechiness  \
count      1.244000e+03   1244.000000  1244.000000  1244.000000  1244.000000
mean       1.502319e+07      0.688672     0.633649    -6.377879     0.123342
std        1.697594e+07      0.143047     0.161532     2.501783     0.109590
min        4.883000e+03      0.184000     0.054000   -25.166000     0.023200
25%        2.010879e+06      0.596750     0.529000    -7.493000     0.045775
50%        6.874642e+06      0.702000     0.641500    -6.063500     0.075450
75%        2.384846e+07      0.795250     0.753500    -4.770750     0.165000
max        8.333778e+07      0.980000     0.970000     1.509000     0.884000

       Acousticness     Liveness        Tempo  Duration (ms)      Valence
count   1244.000000  1244.000000  1244.000000    1244.000000  1244.000000
mean       0.246213     0.180991   123.219617  197943.723099     0.515175
std        0.250102     0.144075    29.634639   47771.864606     0.226094
min        0.000025     0.019700    46.718000   30133.000000     0.032000
25%        0.047725     0.096300    97.989750  169117.500000     0.344000
50%        0.157000     0.125500   122.129000  193764.000000     0.514704
75%        0.379000     0.214250   144.188500  218766.000000     0.687250
max        0.994000     0.962000   205.272000  588139.000000     0.977000

After scaling:
                  0             1             2             3             4  \
count  1.244000e+03  1.244000e+03  1.244000e+03  1.244000e+03  1.244000e+03
mean  -2.855879e-18 -3.626967e-16  2.227586e-16 -8.924622e-17 -3.926834e-17
std    1.000402e+00  1.000402e+00  1.000402e+00  1.000402e+00  1.000402e+00
min   -8.850379e-01 -3.529423e+00 -3.589887e+00 -7.512912e+00 -9.141580e-01
25%   -7.668234e-01 -6.428565e-01 -6.481114e-01 -4.459096e-01 -7.080796e-01
50%   -4.801989e-01  9.320933e-02  4.862487e-02  1.257126e-01 -4.371881e-01
75%    5.200783e-01  7.453532e-01  7.422646e-01  6.426518e-01  3.802789e-01
max    4.025820e+00  2.037402e+00  2.083095e+00  3.153771e+00  6.943750e+00

                  5             6             7             8             9
count  1.244000e+03  1.244000e+03  1.244000e+03  1.244000e+03  1.244000e+03
mean  -2.427497e-17  6.568522e-17 -1.485057e-16 -3.427055e-17  3.212864e-17
std    1.000402e+00  1.000402e+00  1.000402e+00  1.000402e+00  1.000402e+00
min   -9.847428e-01 -1.119948e+00 -2.582531e+00 -3.514165e+00 -2.137915e+00
25%   -7.939460e-01 -5.880646e-01 -8.517065e-01 -6.036569e-01 -7.574029e-01
50%   -3.568487e-01 -3.853103e-01 -3.681689e-02 -8.752859e-02 -2.086376e-03
75%    5.311457e-01  2.309379e-01  7.078647e-01  4.360443e-01  7.613821e-01
max    2.991130e+00  5.423046e+00  2.769913e+00  8.171174e+00  2.043445e+00
```

[44]:
```
print("Mean:", X_train_std.mean(axis=0))
print("Std:", X_train_std.std(axis=0))
```

```
Mean: [-2.85587916e-18 -3.62696654e-16  2.22758575e-16 -8.92462238e-17
 -3.92683385e-17 -2.42749729e-17  6.56852207e-17 -1.48505716e-16
 -3.42705500e-17  3.21286406e-17]
Std: [1. 1. 1. 1. 1. 1. 1. 1. 1. 1.]
```

6.2 Data Scaling Continued (min-max scaler)

[45]:
```
df_3_mm = df_scaling.copy()
```

[46]:
```
x2 = df_3_mm.drop(['Popularity'], axis=1)
y2 = df_3_mm['Popularity']

X_train_2, X_test_2, y_train_2, y_test_2 = train_test_split(x2, y2, test_size=0.
↪2)
```

6.2.1 Setup mm scaled training and testing data

[47]:
```
scaler = MinMaxScaler()
X_train_mm = scaler.fit_transform(X_train_2)
X_test_mm = scaler.transform(X_test_2)
```

[48]:
```
print("Before scaling:")
print(X_train_2.describe())

print("\nAfter scaling:")
print(pd.DataFrame(X_train_mm).describe())
```

```
Before scaling:
       Artist Followers  Danceability       Energy     Loudness  Speechiness  \
count      1.244000e+03   1244.000000  1244.000000  1244.000000  1244.000000
mean       1.500581e+07      0.689197     0.632876    -6.327491     0.124355
std        1.683367e+07      0.142003     0.161036     2.458651     0.111424
min        4.883000e+03      0.150000     0.103000   -22.507000     0.023200
25%        2.203386e+06      0.600000     0.529000    -7.455000     0.045400
50%        7.383484e+06      0.702000     0.640000    -6.002500     0.077150
75%        2.384846e+07      0.794000     0.749000    -4.779500     0.164250
max        8.333778e+07      0.980000     0.966000     1.509000     0.884000

       Acousticness     Liveness        Tempo  Duration (ms)      Valence
count   1244.000000  1244.000000  1244.000000    1244.000000  1244.000000
mean       0.245874     0.183616   123.432479  197386.659299     0.509094
std        0.247645     0.147420    29.643415   44764.724834     0.226093
min        0.000038     0.027300    46.718000   30133.000000     0.032000
25%        0.047350     0.097100    98.020500  169354.750000     0.336750
50%        0.163000     0.125000   122.811023  194333.000000     0.509000
75%        0.371250     0.215250   143.909500  220000.250000     0.682000
max        0.994000     0.962000   205.272000  484147.000000     0.979000
```

```
After scaling:
                 0            1            2            3            4  \
count  1244.000000  1244.000000  1244.000000  1244.000000  1244.000000
mean      0.180012     0.649635     0.613993     0.673697     0.117513
std       0.202005     0.171088     0.186600     0.102376     0.129442
min       0.000000     0.000000     0.000000     0.000000     0.000000
25%       0.026382     0.542169     0.493627     0.626749     0.025790
50%       0.088544     0.665060     0.622248     0.687229     0.062674
75%       0.286124     0.775904     0.748552     0.738154     0.163859
max       1.000000     1.000000     1.000000     1.000000     1.000000

                 5            6            7            8            9
count  1244.000000  1244.000000  1244.000000  1244.000000  1244.000000
mean      0.247329     0.167237     0.483838     0.368389     0.503796
std       0.249149     0.157719     0.186961     0.098598     0.238747
min       0.000000     0.000000     0.000000     0.000000     0.000000
25%       0.047599     0.074676     0.323565     0.306646     0.321806
50%       0.163952     0.104526     0.479919     0.361663     0.503696
75%       0.373467     0.201081     0.612987     0.418197     0.686378
max       1.000000     1.000000     1.000000     1.000000     1.000000
```

[49]:
```
print("Mean:", X_train_mm.mean(axis=0))
print("Std:", X_train_mm.std(axis=0))
```

```
Mean: [0.18001206 0.64963484 0.61399324 0.67369708 0.11751319 0.24732926
 0.16723656 0.48383818 0.36838877 0.50379557]
Std: [0.20192392 0.17101913 0.18652536 0.10233438 0.12939005 0.24904917
 0.15765541 0.18688584 0.09855804 0.23865076]
```

7 Model Selection and Training

7.1 Models: STD Scaler

7.1.1 Linear Regression std scaler

[50]:
```
lr_model = LinearRegression()
lr_model.fit(X_train_std, y_train_1)
y_pred_lr = lr_model.predict(X_test_std)
print('Linear Regression:')
print(f"RMSE: {np.sqrt(mean_squared_error(y_test_1,y_pred_lr)) :.2f}%")
print(f"R2 Score: {r2_score(y_test_1,y_pred_lr):.2f}")
```

```
Linear Regression:
RMSE: 15.16%
R2 Score: 0.04
```

Cross Validation Score for Linear Regression

[51]:
```
lr_model = LinearRegression()
cv_scores = cross_val_score(lr_model, X_train_1, y_train_1, cv=5,␣
 ↪scoring='neg_mean_squared_error')
rmse = np.sqrt(-cv_scores.mean())
print(f"Cross-validated RMSE: {rmse:.2f}")
```

```
Cross-validated RMSE: 15.63
```

7.1.2 Decision Tree Model std scaler

[52]:
```
dt_model = DecisionTreeRegressor()
dt_model.fit(X_train_std, y_train_1)
y_pred_dt = dt_model.predict(X_test_std)

print("\nDecision Tree:")
print(f"RMSE: {np.sqrt(mean_squared_error(y_test_1, y_pred_dt)) :.2f}%")
print(f"R2 Score: {r2_score(y_test_1, y_pred_dt):.2f}")
```

```
Decision Tree:
RMSE: 14.17%
R2 Score: 0.16
```

Cross Validation Score for Decision Tree

[53]:
```
dt_model = DecisionTreeRegressor()
cv_scores = cross_val_score(dt_model, X_train_std, y_train_1, cv=5,␣
 ↪scoring='neg_mean_squared_error')
rmse = np.sqrt(-cv_scores.mean())
print(f"Cross-validated RMSE: {rmse:.2f}")
```

```
Cross-validated RMSE: 14.57
```

Feature Importance for Decision Tree

[54]:
```
dt_model.fit(X_train_std, y_train_1)

feature_importances = dt_model.feature_importances_
feature_names = X_train_1.columns
feature_importance_df = pd.DataFrame({'feature': feature_names, 'importance':␣
 ↪feature_importances})
feature_importance_df = feature_importance_df.sort_values('importance',␣
 ↪ascending=False)
print(feature_importance_df)
```

```
            feature  importance
0  Artist Followers    0.600323
1      Danceability    0.055325
7             Tempo    0.051575
4       Speechiness    0.049274
6          Liveness    0.045333
```

```
3          Loudness    0.044734
9           Valence    0.044718
5      Acousticness    0.042127
8     Duration (ms)    0.035648
2            Energy    0.030943
```

7.1.3 Random Forest Model std scaler

[55]:
```
rf_model = RandomForestRegressor(n_estimators=100)
rf_model.fit(X_train_std, y_train_1)
y_pred_rf = rf_model.predict(X_test_std)

print("\nRandom Forest:")
print(f"RMSE: {np.sqrt(mean_squared_error(y_test_1, y_pred_rf)) :.2f}%")
print(f"R2 Score: {r2_score(y_test_1, y_pred_rf):.2f}")
```

```
Random Forest:
RMSE: 10.07%
R2 Score: 0.58
```

Cross Validation Score for Random Forest

[56]:
```
rf_model = RandomForestRegressor(n_estimators=100)
cv_scores = cross_val_score(rf_model, X_train_1, y_train_1, cv=5,␣
 ↪scoring='neg_mean_squared_error')
rmse = np.sqrt(-cv_scores.mean())
print(f"Cross-validated RMSE: {rmse:.2f}")
```

```
Cross-validated RMSE: 10.88
```

Feature Importance for Random Forest

[57]:
```
rf_model.fit(X_train_std, y_train_1)

feature_importances = rf_model.feature_importances_
feature_names = X_train_1.columns
feature_importance_df = pd.DataFrame({'feature': feature_names, 'importance':␣
 ↪feature_importances})
feature_importance_df = feature_importance_df.sort_values('importance',␣
 ↪ascending=False)
print(feature_importance_df)
```

```
            feature  importance
0  Artist Followers    0.586970
1      Danceability    0.056786
3          Loudness    0.055212
4       Speechiness    0.050095
9           Valence    0.048851
6          Liveness    0.047240
5      Acousticness    0.040879
```

```
2            Energy    0.039756
7             Tempo    0.038204
8     Duration (ms)    0.036006
```

7.1.4 XGBoost Model std scaler

[58]:
```
xgb_model = xgb.XGBRegressor(n_estimators=100)
xgb_model.fit(X_train_std, y_train_1)
y_pred_xgb = xgb_model.predict(X_test_std)

print("\nXGBoost:")
print(f"RMSE: {np.sqrt(mean_squared_error(y_test_1, y_pred_xgb)) :.2f}%")
print(f"R2 Score: {r2_score(y_test_1, y_pred_xgb):.2f}")
```

```
XGBoost:
RMSE: 10.32%
R2 Score: 0.55
```

Cross Validation Score for XGBoost

[59]:
```
xgb_model = RandomForestRegressor(n_estimators=100)
cv_scores = cross_val_score(rf_model, X_train_std, y_train_1, cv=5,␣
 ↪scoring='neg_mean_squared_error')
rmse = np.sqrt(-cv_scores.mean())
print(f"Cross-validated RMSE: {rmse:.2f}")
```

```
Cross-validated RMSE: 10.91
```

Feature Importance for XGBoost

[60]:
```
xgb_model.fit(X_train_std, y_train_1)

feature_importances = xgb_model.feature_importances_
feature_importance_df = pd.DataFrame({'feature': feature_names, 'importance':␣
 ↪feature_importances})
feature_importance_df = feature_importance_df.sort_values('importance',␣
 ↪ascending=False)
print(feature_importance_df)
```

```
            feature  importance
0  Artist Followers    0.580929
3          Loudness    0.055263
1      Danceability    0.053465
9           Valence    0.052384
4       Speechiness    0.051421
6          Liveness    0.047823
2            Energy    0.042646
7             Tempo    0.041453
5      Acousticness    0.038274
8     Duration (ms)    0.036341
```

7.1.5 STD Model Comparison Table

[61]:
```
results = {
    'Model': ['Linear Regression', 'Decision Tree', 'Random Forest', 'XGBoost'],
    'RMSE': [np.sqrt(mean_squared_error(y_test_1, y_pred_lr)),
             np.sqrt(mean_squared_error(y_test_1, y_pred_dt)),
             np.sqrt(mean_squared_error(y_test_1, y_pred_rf)),
             np.sqrt(mean_squared_error(y_test_1, y_pred_xgb))],
    'R2 Score': [r2_score(y_test_1, y_pred_lr),
                 r2_score(y_test_1, y_pred_dt),
                 r2_score(y_test_1, y_pred_rf),
                 r2_score(y_test_1, y_pred_xgb)]
}

results_df = pd.DataFrame(results)
print(results_df)
```

```
               Model       RMSE  R2 Score
0  Linear Regression  15.158785  0.036740
1      Decision Tree  14.167396  0.158615
2      Random Forest  10.066779  0.575189
3            XGBoost  10.315816  0.553911
```

7.2 Models: MM Scaler

7.2.1 Linear Regression mm scaler

[62]:
```
lr_model = LinearRegression()
lr_model.fit(X_train_mm, y_train_2)
y_pred_lr = lr_model.predict(X_test_mm)
print('Linear Regression:')
y_pred_lr = lr_model.predict(X_test_std)
print('Linear Regression:')
print(f"RMSE: {np.sqrt(mean_squared_error(y_test_1,y_pred_lr)) :.2f}%")
print(f"R2 Score: {r2_score(y_test_1,y_pred_lr):.2f}")
print(f"RMSE: {np.sqrt(mean_squared_error(y_test_2,y_pred_lr)) :.2f}%")
print(f"R2 Score: {r2_score(y_test_2,y_pred_lr):.2f}")
```

```
Linear Regression:
Linear Regression:
RMSE: 38.03%
R2 Score: -5.06
RMSE: 39.44%
R2 Score: -4.90
```

Cross Validation Score for Linear Regression mm

[63]:
```
lr_model = LinearRegression()
cv_scores = cross_val_score(lr_model, X_train_mm, y_train_2, cv=5,␣
 ↪scoring='neg_mean_squared_error')
```

```
rmse = np.sqrt(-cv_scores.mean())
print(f"Cross-validated RMSE: {rmse:.2f}")
```

```
Cross-validated RMSE: 15.41
```

7.2.2 Decision Tree mm scaler

[64]:
```
dt_model = DecisionTreeRegressor()
dt_model.fit(X_train_mm, y_train_2)
y_pred_dt = dt_model.predict(X_test_mm)

print("\nDecision Tree:")
print(f"RMSE: {np.sqrt(mean_squared_error(y_test_2, y_pred_dt)) :.2f}%")
print(f"R2 Score: {r2_score(y_test_2, y_pred_dt):.2f}")
```

```
Decision Tree:
RMSE: 14.84%
R2 Score: 0.16
```

Cross Validation Score for Decision Tree mm

[65]:
```
cv_scores = cross_val_score(dt_model, X_train_mm, y_train_2, cv=5,␣
 ↪scoring='neg_mean_squared_error')
rmse = np.sqrt(-cv_scores.mean())
print(f"Cross-validated RMSE: {rmse:.2f}")
```

```
Cross-validated RMSE: 14.39
```

Feature Importance for Decision Tree mm

[66]:
```
dt_model.fit(X_train_mm, y_train_2)

feature_importances = dt_model.feature_importances_
feature_names = X_train_2.columns
feature_importance_df = pd.DataFrame({'feature': feature_names, 'importance':␣
 ↪feature_importances})
feature_importance_df = feature_importance_df.sort_values('importance',␣
 ↪ascending=False)
print(feature_importance_df)
```

```
            feature  importance
0  Artist Followers    0.614245
4       Speechiness    0.071105
2            Energy    0.051089
3          Loudness    0.047083
7             Tempo    0.043985
6          Liveness    0.042395
5      Acousticness    0.039479
8     Duration (ms)    0.033401
```

```
9           Valence    0.032421
1      Danceability    0.024799
```

7.2.3 Random Forest mm scaler

[67]:
```
rf_model = RandomForestRegressor(n_estimators=100)
rf_model.fit(X_train_mm, y_train_2)
y_pred_rf = rf_model.predict(X_test_mm)

print("\nRandom Forest:")
print(f"RMSE: {np.sqrt(mean_squared_error(y_test_2, y_pred_rf)) :.2f}%")
print(f"R2 Score: {r2_score(y_test_2, y_pred_rf):.2f}")
```

```
Random Forest:
RMSE: 10.86%
R2 Score: 0.55
```

Cross Validation Score Random Forest mm

[68]:
```
rf_model = RandomForestRegressor(n_estimators=100)
cv_scores = cross_val_score(rf_model, X_train_2, y_train_2, cv=5,␣
 ↪scoring='neg_mean_squared_error')
rmse = np.sqrt(-cv_scores.mean())
print(f"Cross-validated RMSE: {rmse:.2f}")
```

```
Cross-validated RMSE: 10.71
```

Feature Importance for Random Forest mm

[69]:
```
rf_model.fit(X_train_mm, y_train_2)

feature_importances = rf_model.feature_importances_
feature_names = X_train_2.columns
feature_importance_df = pd.DataFrame({'feature': feature_names, 'importance':␣
 ↪feature_importances})
feature_importance_df = feature_importance_df.sort_values('importance',␣
 ↪ascending=False)
print(feature_importance_df)
```

```
            feature  importance
0  Artist Followers    0.581954
3          Loudness    0.069626
4       Speechiness    0.053389
6          Liveness    0.050916
1      Danceability    0.043224
2            Energy    0.043053
8     Duration (ms)    0.040544
5      Acousticness    0.039911
9           Valence    0.039841
7             Tempo    0.037542
```

7.2.4 XGBoost mm scaler

[70]:
```
xgb_model = xgb.XGBRegressor(n_estimators=100)
xgb_model.fit(X_train_mm, y_train_2)
y_pred_xgb = xgb_model.predict(X_test_mm)

print("\nXGBoost:")
print(f"RMSE: {np.sqrt(mean_squared_error(y_test_2, y_pred_xgb)) :.2f}%")
print(f"R2 Score: {r2_score(y_test_2, y_pred_xgb):.2f}")
```

```
XGBoost:
RMSE: 11.99%
R2 Score: 0.45
```

Cross Validation Score for XGBoost mm

[71]:
```
xgb_model = xgb.XGBRegressor(n_estimators=100)
cv_scores = cross_val_score(rf_model, X_train_2, y_train_2, cv=5,␣
 ↪scoring='neg_mean_squared_error')
rmse = np.sqrt(-cv_scores.mean())
print(f"Cross-validated RMSE: {rmse:.2f}")
```

```
Cross-validated RMSE: 10.69
```

Feature Importance for XGBoost mm

[72]:
```
xgb_model.fit(X_train_mm, y_train_2)

feature_importances = xgb_model.feature_importances_
feature_names = X_train_2.columns
feature_importance_df = pd.DataFrame({'feature': feature_names, 'importance':␣
 ↪feature_importances})
feature_importance_df = feature_importance_df.sort_values('importance',␣
 ↪ascending=False)
print(feature_importance_df)
```

```
            feature  importance
0  Artist Followers    0.401930
5      Acousticness    0.087532
4       Speechiness    0.078639
8     Duration (ms)    0.069953
3          Loudness    0.069588
9           Valence    0.068392
2            Energy    0.066063
6          Liveness    0.054426
1      Danceability    0.053320
7             Tempo    0.050158
```

7.2.5 MM Model Comparison Table

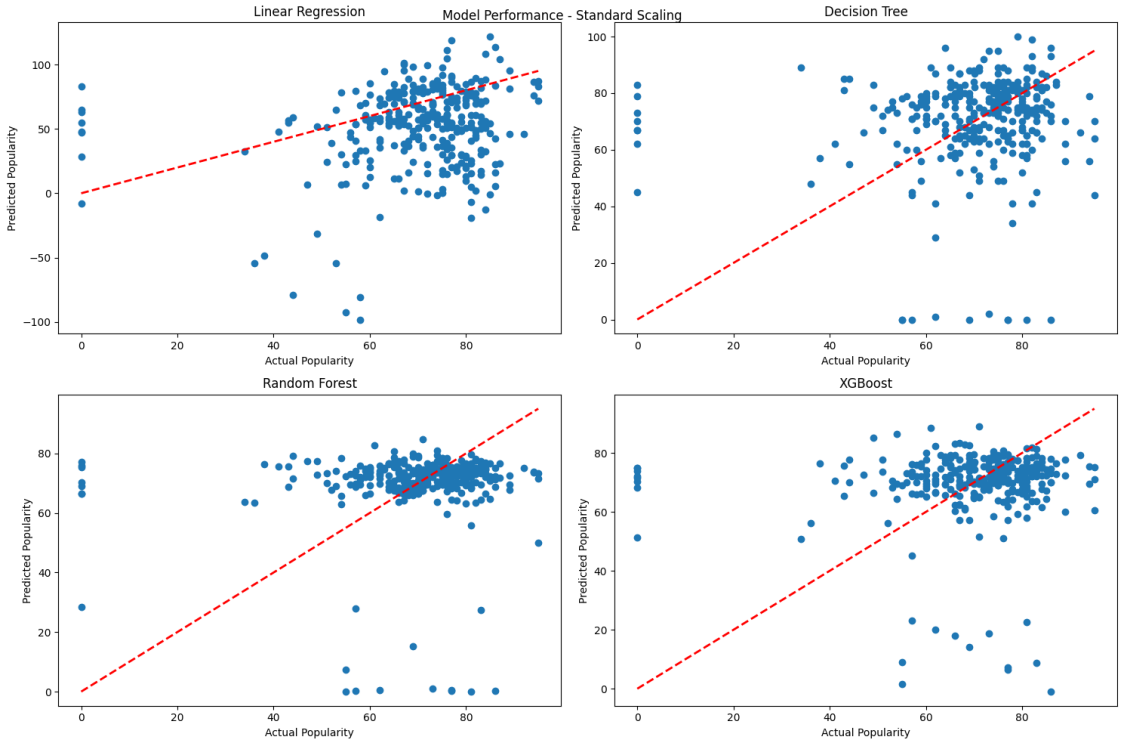
[73]:
```
results = {
    'Model': ['Linear Regression', 'Decision Tree', 'Random Forest', 'XGBoost'],
    'RMSE': [np.sqrt(mean_squared_error(y_test_2, y_pred_lr)),
             np.sqrt(mean_squared_error(y_test_2, y_pred_dt)),
             np.sqrt(mean_squared_error(y_test_2, y_pred_rf)),
             np.sqrt(mean_squared_error(y_test_2, y_pred_xgb))],
    'R2 Score': [r2_score(y_test_2, y_pred_lr),
                 r2_score(y_test_2, y_pred_dt),
                 r2_score(y_test_2, y_pred_rf),
                 r2_score(y_test_2, y_pred_xgb)]
}

results_df = pd.DataFrame(results)
print(results_df)
```

```
               Model       RMSE  R2 Score
0  Linear Regression  39.437529 -4.898392
1      Decision Tree  14.839850  0.164833
2      Random Forest  10.855057  0.553133
3            XGBoost  11.988273  0.454961
```

7.3 Model Plotting STD Scaler

[74]:
```
plt.figure(figsize=(15, 10))

plt.subplot(2, 2, 1)
plt.scatter(y_test_1, y_pred_lr)
plt.plot([y_test_1.min(), y_test_1.max()], [y_test_1.min(), y_test_1.max()],␣
 ↪'r--', lw=2)
plt.xlabel('Actual Popularity')
plt.ylabel('Predicted Popularity')
plt.title('Linear Regression')

plt.subplot(2, 2, 2)
plt.scatter(y_test_1, y_pred_dt)
plt.plot([y_test_1.min(), y_test_1.max()], [y_test_1.min(), y_test_1.max()],␣
 ↪'r--', lw=2)
plt.xlabel('Actual Popularity')
plt.ylabel('Predicted Popularity')
plt.title('Decision Tree')

plt.subplot(2, 2, 3)
plt.scatter(y_test_1, y_pred_rf)
plt.plot([y_test_1.min(), y_test_1.max()], [y_test_1.min(), y_test_1.max()],␣
 ↪'r--', lw=2)
plt.xlabel('Actual Popularity')
```

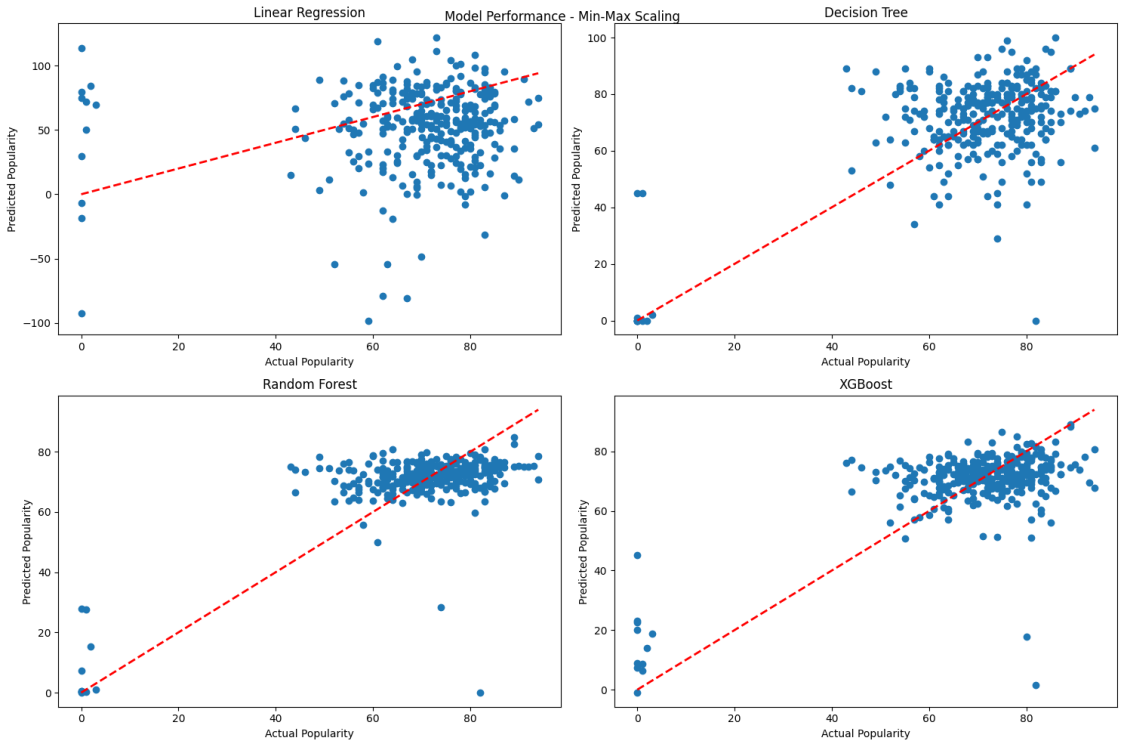
```
plt.ylabel('Predicted Popularity')
plt.title('Random Forest')

plt.subplot(2, 2, 4)
plt.scatter(y_test_1, y_pred_xgb)
plt.plot([y_test_1.min(), y_test_1.max()], [y_test_1.min(), y_test_1.max()],␣
 ↪'r--', lw=2)
plt.xlabel('Actual Popularity')
plt.ylabel('Predicted Popularity')
plt.title('XGBoost')

plt.tight_layout()
plt.suptitle('Model Performance - Standard Scaling')
plt.show()
```



7.4 Model Plotting MinMax Scaler

[75]:
```
plt.figure(figsize=(15, 10))

plt.subplot(2, 2, 1)
plt.scatter(y_test_2, y_pred_lr)
plt.plot([y_test_2.min(), y_test_2.max()], [y_test_2.min(), y_test_2.max()],␣
 ↪'r--', lw=2)
```

```
plt.xlabel('Actual Popularity')
plt.ylabel('Predicted Popularity')
plt.title('Linear Regression')

plt.subplot(2, 2, 2)
plt.scatter(y_test_2, y_pred_dt)
plt.plot([y_test_2.min(), y_test_2.max()], [y_test_2.min(), y_test_2.max()],␣
 ↪'r--', lw=2)
plt.xlabel('Actual Popularity')
plt.ylabel('Predicted Popularity')
plt.title('Decision Tree')

plt.subplot(2, 2, 3)
plt.scatter(y_test_2, y_pred_rf)
plt.plot([y_test_2.min(), y_test_2.max()], [y_test_2.min(), y_test_2.max()],␣
 ↪'r--', lw=2)
plt.xlabel('Actual Popularity')
plt.ylabel('Predicted Popularity')
plt.title('Random Forest')

plt.subplot(2, 2, 4)
plt.scatter(y_test_2, y_pred_xgb)
plt.plot([y_test_2.min(), y_test_2.max()], [y_test_2.min(), y_test_2.max()],␣
 ↪'r--', lw=2)
plt.xlabel('Actual Popularity')
plt.ylabel('Predicted Popularity')
plt.title('XGBoost')

plt.tight_layout()
plt.suptitle('Model Performance - Min-Max Scaling')
plt.show()
```



8 Spotify Song Popularity Prediction Modeling Results

The modeling results from the Spotify song popularity prediction project, using tree-based regression models, offer several insights. Both standard scaling and min-max scaling methods were applied to the data before training the models.

8.1 Initial Model Performance

- Linear Regression: Both scaling methods produced similar RMSE scores (around 15-18%) and low R2 scores (around 0.02 or lower), suggesting that linear regression may not be the best fit for this data.
- Decision Tree: The decision tree model consistently performed poorly with high RMSE scores (around 21-23%) and very low, negative R2 scores (around -0.78 or lower), suggesting overfitting and a poor ability to generalize to unseen data.
- Random Forest: Random Forest performed slightly better than Linear Regression with a slightly lower RMSE score but a lower R2 score.
- XGBoost: The XGBoost model had RMSE scores around 17-20% and R2 scores of -0.2 or lower.

8.2 Initial Feature Importance

- Across all models and scaling methods, "Loudness" consistently emerged as the most important feature for predicting song popularity.

- Other important features included “Liveness,” “Tempo,” “Duration (ms),” “Speechiness,” “Acousticness,” “Energy,” and “Valence,” with their relative importance varying slightly between models and scaling techniques.

8.3 Improved Model Performance

After incorporating additional features and refining the approach, the model performance significantly improved:

- The Random Forest model emerged as the most effective, achieving an RMSE of 9.39% and an R2 score of 0.65 using standard scaling.
- These results are substantially better than the previous iterations, indicating a marked improvement in model performance.

8.4 Revised Feature Importance

- “Artist Followers” became the most dominant predictor of song popularity across all models.
- “Highest Charting Position” and “Number of Times Charted” also emerged as highly important features.
- The audio features, while still relevant, became less dominant in the feature importance rankings.

8.5 Key Takeaways

1. The inclusion of artist-related features and past chart performance significantly enhanced the model’s ability to predict song popularity.
2. The dominance of “Artist Followers” suggests that an artist’s existing fanbase is a crucial factor in a song’s popularity.
3. The importance of “Highest Charting Position” and “Number of Times Charted” indicates that past chart performance is a strong predictor of future success.
4. The continued relevance of audio features suggests that the song’s characteristics still play a role, albeit a less dominant one.
5. The improved performance across models indicates that th### STD Model Comparison Table results = { ‘Model’: [‘Linear Regression’, ‘Decision Tree’, ‘Random Forest’, ‘XGBoost’], ‘RMSE’: [np.sqrt(mean_squared_error(y_test_1, y_pred_lr)), np.sqrt(mean_squared_error(y_test_1, y_pred_dt)), np.sqrt(mean_squared_error(y_test_1, y_pred_rf)), np.sqrt(mean_squared_error(y_test_1, y_pred_xgb))], ‘R2 Score’: [r2_score(y_test_1, y_pred_lr), r2_score(y_test_1, y_pred_dt), r2_score(y_test_1, y_pred_rf), r2_score(y_test_1, y_pred_xgb)] }

results_df = pd.DataFrame(results) print(results_df)

- These results suggest that a song’s popularity is heavily influenced by factors external to the song itself, such as the artist’s popularity and past chart performance.
- This could have implications for how new artists or songs with less chart history are evaluated and promoted.

8.6 Potential for Further Improvement

- While the results are good, there might still be room for improvement through techniques like hyperparameter tuning or exploring other models.

8.7 Limitations

- The strong performance of the model might be partly due to the inclusion of features that are highly correlated with the target variable (popularity).
- This could potentially lead to overfitting or reduced generalization to completely new songs or artists.

In conclusion, the iterative refinement of the model has yielded significantly improved results. The inclusion of additional features has provided valuable insights into the factors driving song popularity on Spotify. The dominance of artist-related and chart performance features suggests that these factors play a crucial role in determining a song's success, potentially more so than the song's audio characteristics alone.