Spotify Regression Project

Shai Finger
August 2021

Overview

- Dataset and Objective
- EDA
- Preprocessing
- Feature Engineering
- Regression Models
- Summary and Future Directions

Data and Objective

The dataset chosen for this project has 170,653 rows, each representing one music track on Spotify originally released between 1921 to Nov. 2020.
The 19 columns of the dataset contain various features of the tracks:

1. Valence (Float ranging 0 to 1)
2. Year (Release year)
3. Acousticness (Float ranging 0 to 1)
4. Artists (List of artists)
5. Danceability (Float ranging 0 to 1)
6. Duration_ms (Integer)
7. Energy (Float ranging 0 to 1)
8. Explicit (0 /1)
9. id (Unique string track generated by Spotify)
10. Instrumentalness (Float ranging 0 to 1)
11. Key (Key encoding 0 to 11 - C = 0, C# = 1, etc.)
12. Liveness (Float ranging 0 to 100)
13. Loudness (Float typically ranging from -60 to 0)
14. Mode (0: Minor / 1: Major)
15. Name (Name of the song)
16. **Popularity (Float ranging 0 to 100)**
17. Release_date
18. Speechiness (Float ranging 0 to 100)
19. Tempo (Float typically ranging from 50 to 150)

Objective
Attempt to predict the popularity of a contemporary track (song) on Spotify based on the track's features.

Data is originally from Spotify (available through API) and was taken from Kaggle.
Source: https://www.kaggle.com/yamaerenay/spotify-dataset-19212020-160k-tracks

EDA (1/5)

- Original data set has 170,653 rows, each representing one track originally released between 1921 to Nov. 2020.
- Since we are trying to predict the popularity of a contemporary or new song, tracks released before 2011 were dropped – leaving us with **19,788 tracks** released in the last decade.
- Initial analysis:
  1. No missing values.
  2. No duplicates found.
  3. However, after dropping the unique “id” feature, 39 duplicate rows were found and removed, resulting in 19,749 rows.
- Preliminary transformations:
  1. Converted duration from milliseconds to minutes.
  2. Added textual values of modality (minor, major) for graphs and easier analysis.
  3. Added textual values of keys (C,D,E,etc.) for graphs and easier analysis.
  4. Added textual explanation of explicitness (“Explicit”, “Not explicit or unknown”).

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 19749 entries, 0 to 19748
Data columns (total 18 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   valence           19749 non-null  float64
 1   year              19749 non-null  int64
 2   acousticness      19749 non-null  float64
 3   artists           19749 non-null  object
 4   danceability      19749 non-null  float64
 5   duration_ms       19749 non-null  int64
 6   energy            19749 non-null  float64
 7   explicit          19749 non-null  int64
 8   instrumentalness  19749 non-null  float64
 9   key               19749 non-null  int64
 10  liveness          19749 non-null  float64
 11  loudness          19749 non-null  float64
 12  mode              19749 non-null  int64
 13  name              19749 non-null  object
 14  popularity        19749 non-null  int64
 15  release_date      19749 non-null  object
 16  speechiness       19749 non-null  float64
 17  tempo             19749 non-null  float64
dtypes: float64(9), int64(6), object(3)
memory usage: 2.7+ MB
```

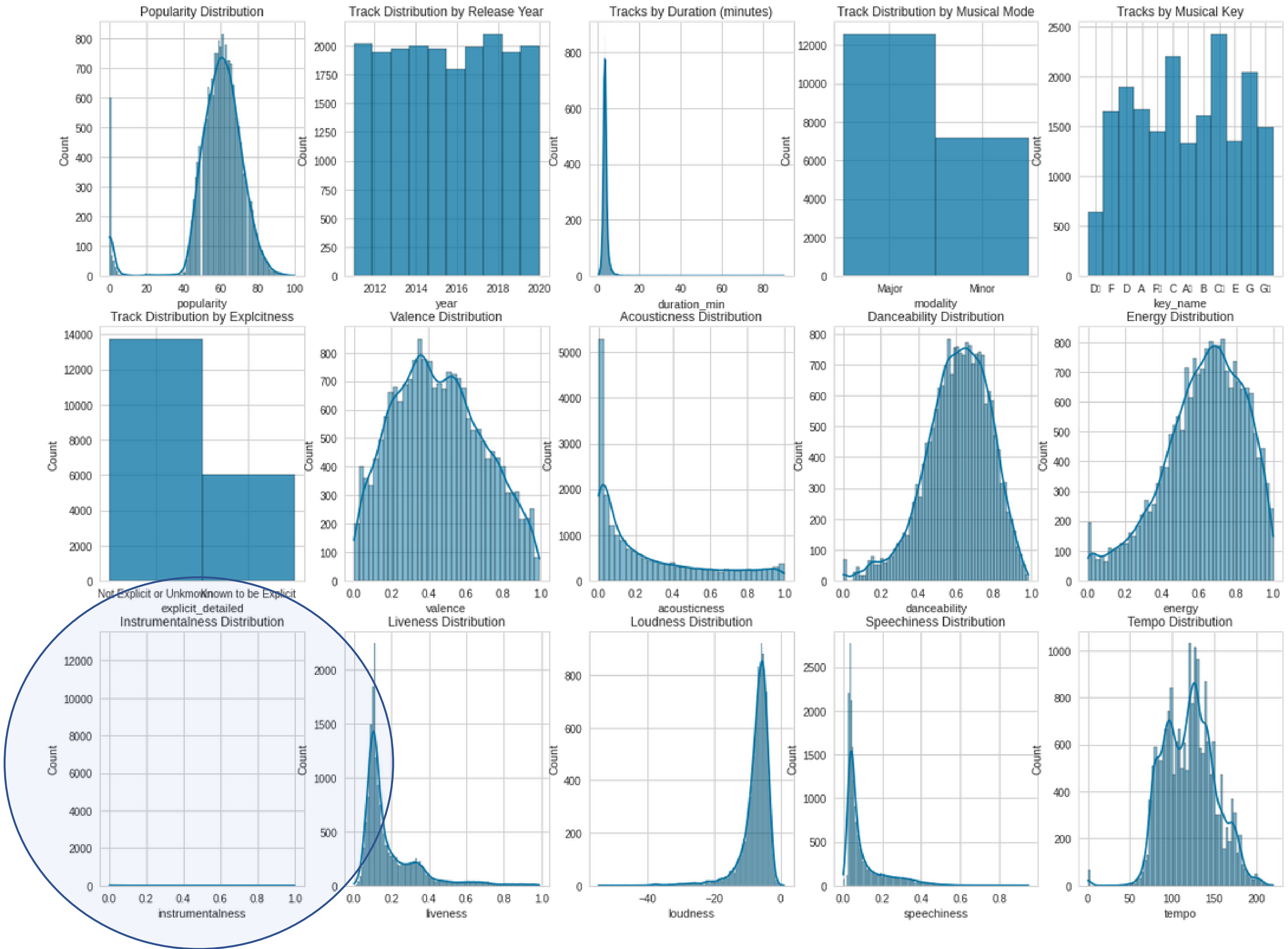
EDA (2/5)

Most numerical features are on a 0-1 scale

Potential outliers:
- Duration
- Tempo

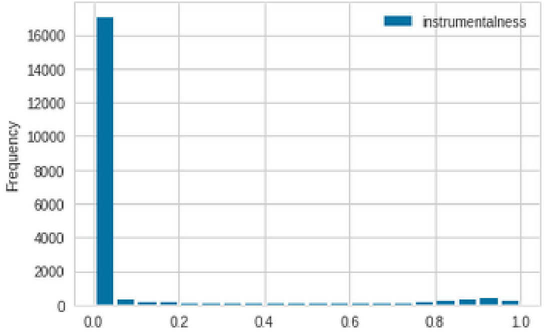
Rescale to 0 to 1?
- Duration
- Tempo
- Loudness
- Popularity

What the f%$#?
- Instrumentalness



EDA (3/5)

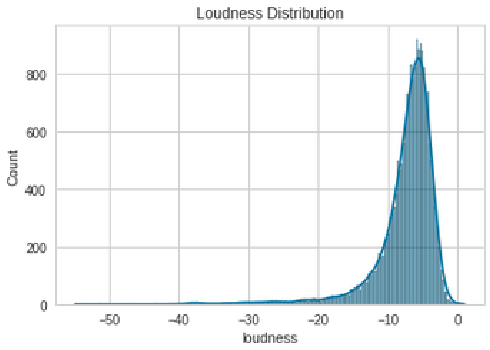
Instrumentalness




```
count    19749.000000
mean         0.080618
std          0.235063
min          0.000000
25%          0.000000
50%          0.000002
75%          0.000717
max          1.000000
```

Decision:
Split to 3 categories:
1. under 0.2
2. between 0.2 and 0.8
3. over 0.8

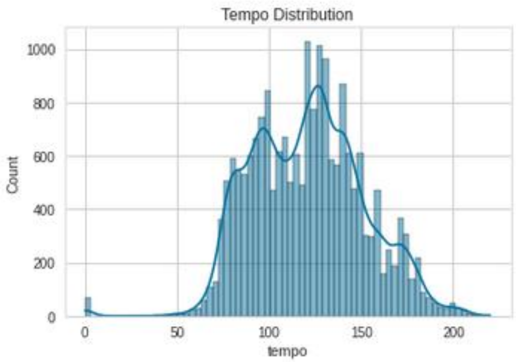
Loudness




```
count    19749.000000
mean        -7.479523
std          4.624972
min        -54.837000
25%         -8.530000
50%         -6.421000
75%         -4.896000
max          1.023000
```

Decision:
1. Rescale to 0-1 using MinMax

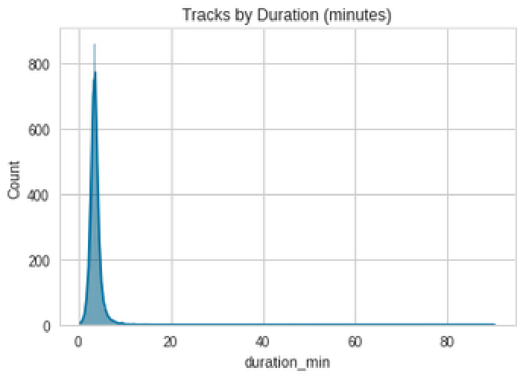
Tempo




```
count    19749.000000
mean       120.888211
std         30.285489
min          0.000000
25%         97.031000
50%        120.931000
75%        140.081000
max        220.099000
```

Decision:
1. Drop tracks with tempo=0 (68 tracks)
2. Rescale to 0-1 using MinMax

Duration




```
count    19749.000000
mean         3.701432
std          1.357751
min          0.505017
25%          3.082667
50%          3.561783
75%          4.099467
max         90.058333
```

Decision:
1. Drop outliers using 3 stdev (265 tracks)
2. Rescale to 0-1 using MinMax

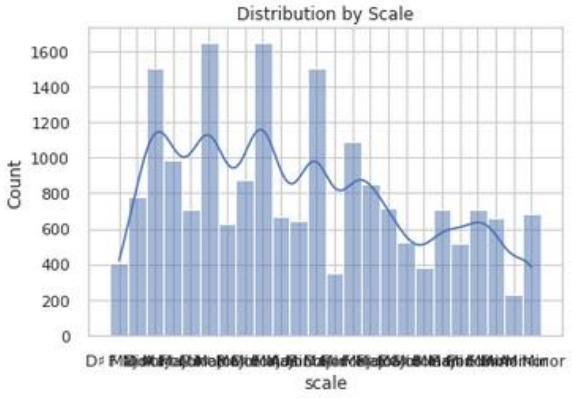
Popularity

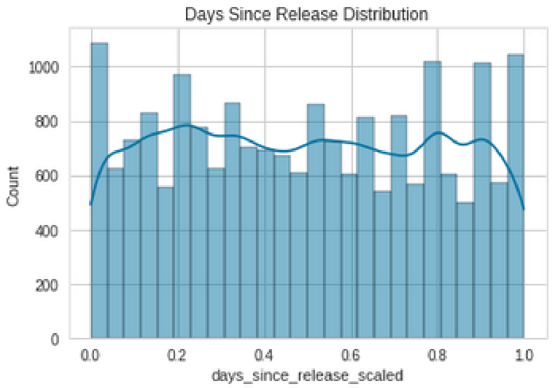
Decision:
Divide by 100 to rescale to 0-1

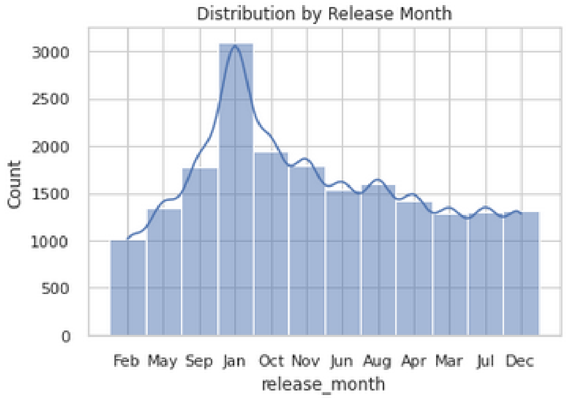
Result: loss of 333 rows (1.7%)

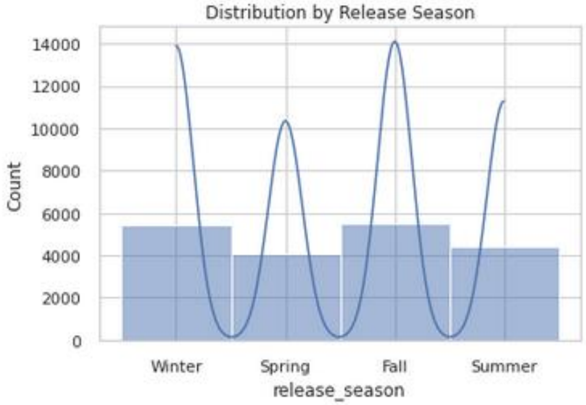
Data Enrichment

1. Added new feature ”scale” – combination of “key” (C) and “mode” (major) - i.e. C# major, Bb minor, etc.
2. Added ”days since release” – computed # of days between release date and database date (scaled to 0-1)
3. Added ”release month” – extracted from “release date” - i.e. Jan, Feb, Mar, etc.
4. Added new feature ”release season” – combination of months - i.e. Dec, Jan, Feb -> “Winter”, etc.



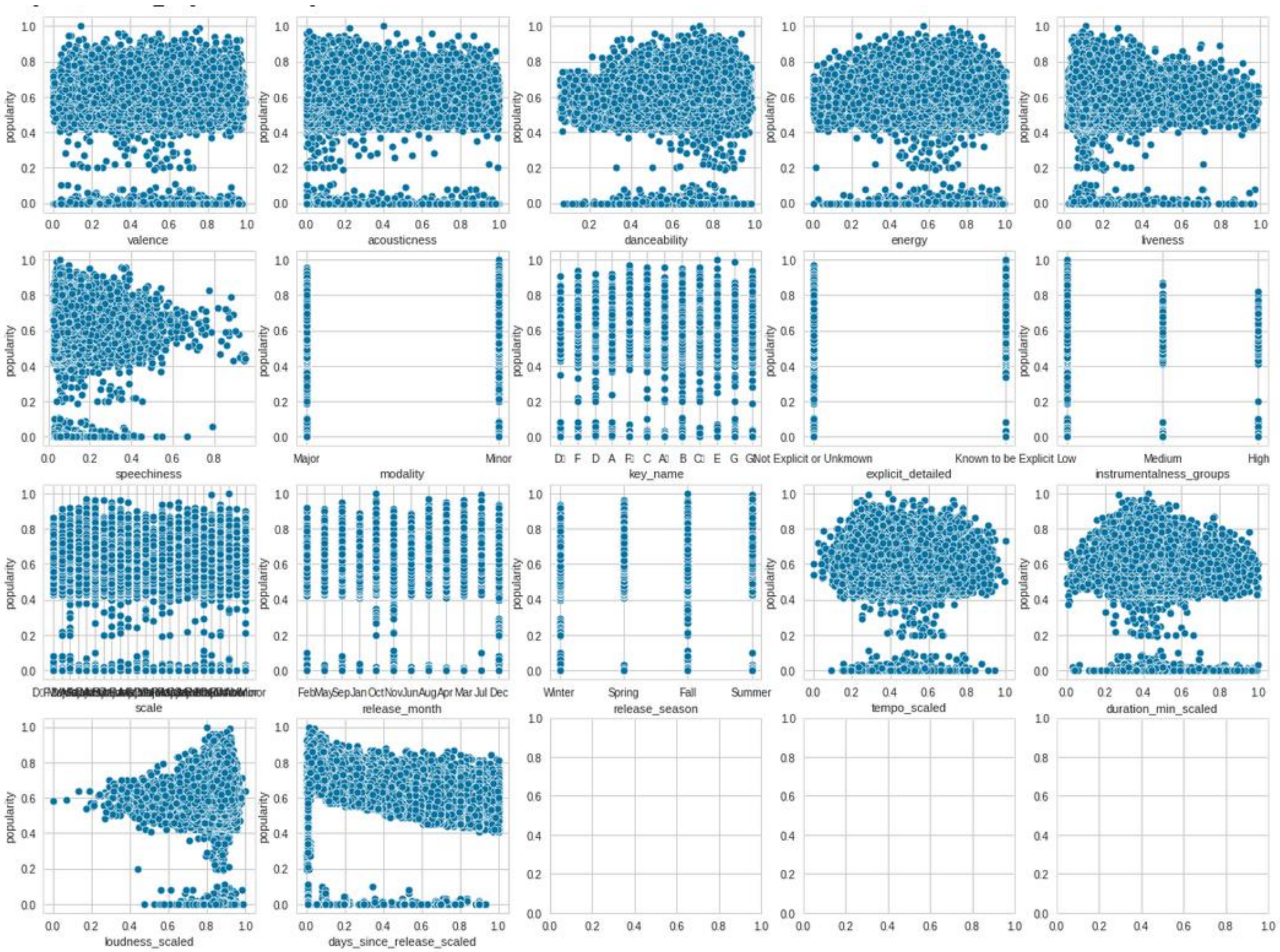






Feature Selection

With plotting it is difficult to see correlations with popularity



Correlation Matrix

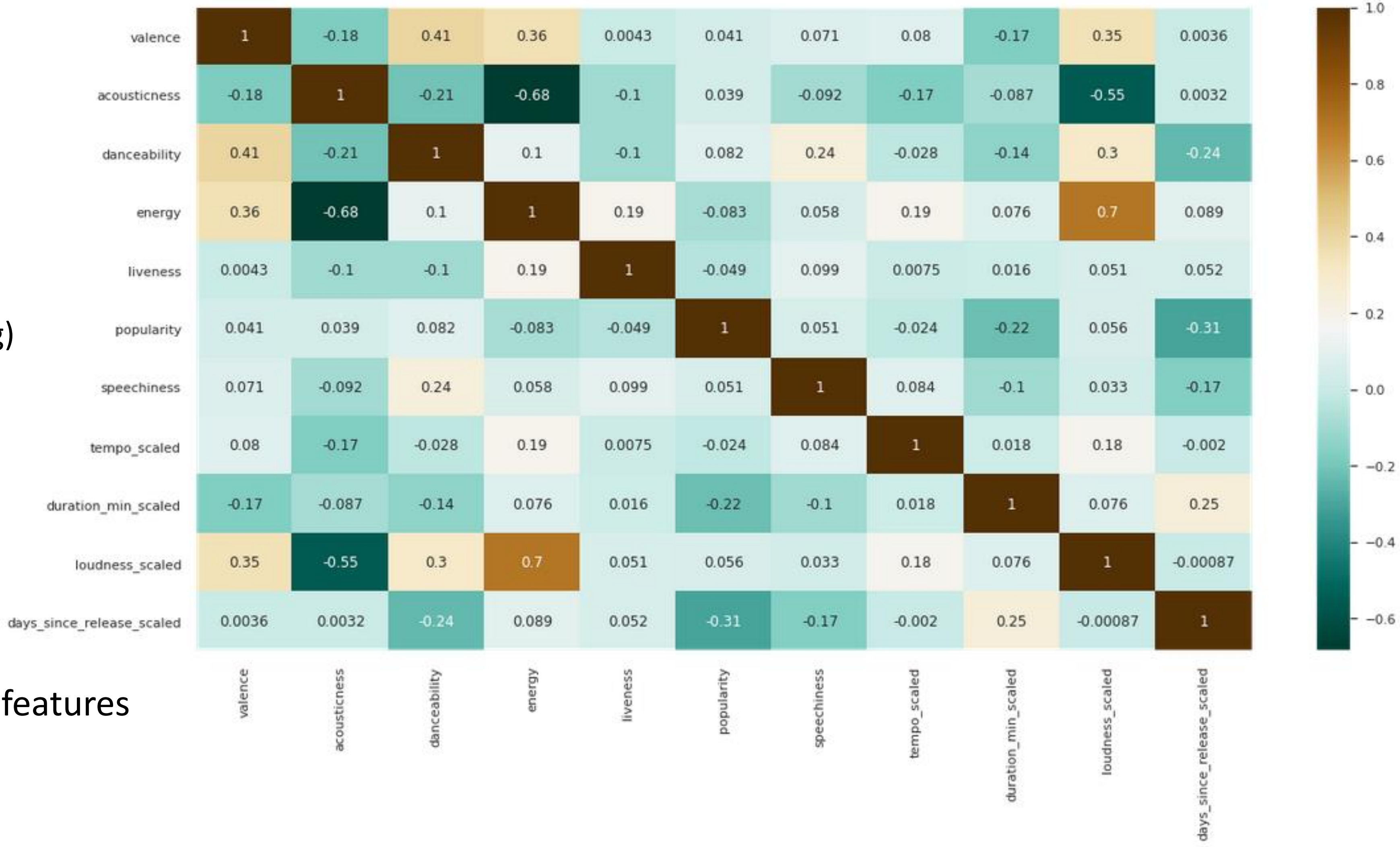
High correlations between features:

- Energy & loudness
- Valence & danceability
- Valence & energy
- Valence & loudness
- Energy & acousticness (neg)
- Loudness & acousticness (neg)

High correlations with popularity:

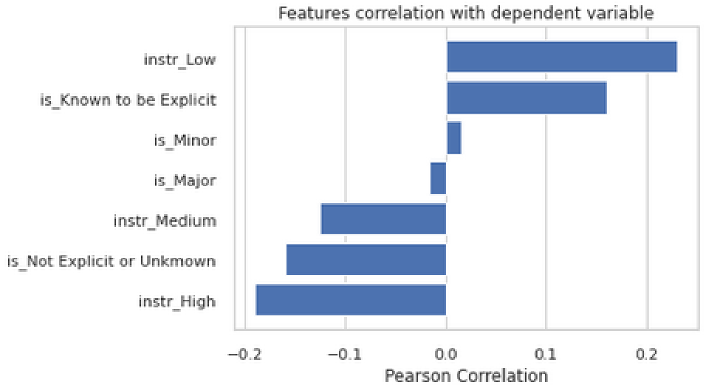
- Duration
- Days since release (neg)

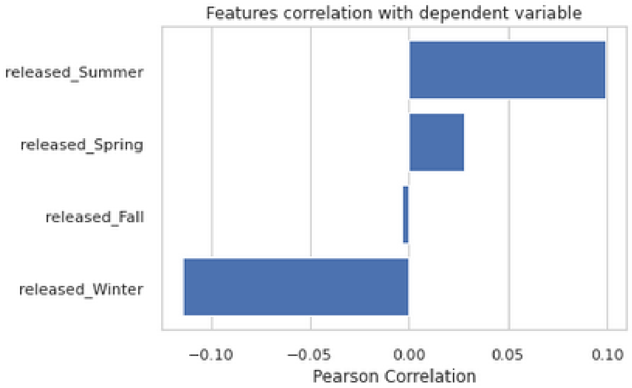
Matrix only shows numerical features

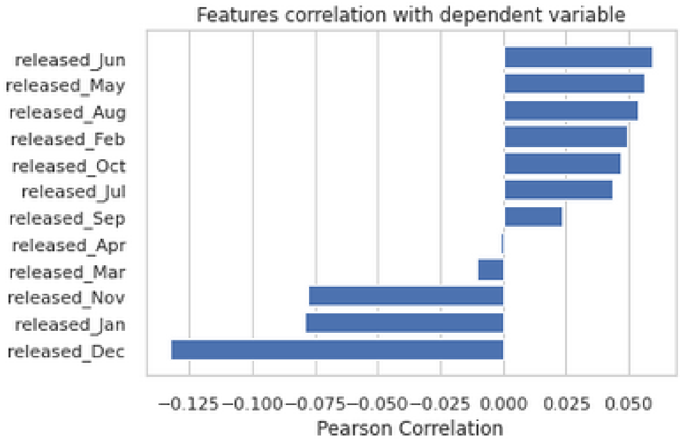


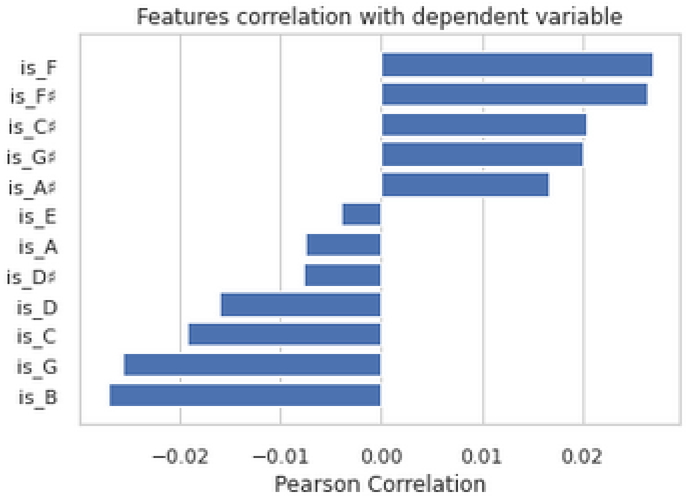
Categorical Feature Correlation

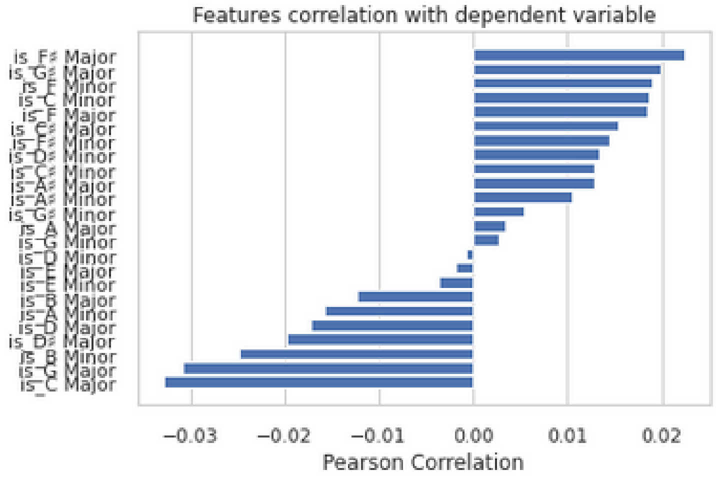
Used one hot encoding (added 58 dummy variables) to examine correlation of categorical features with popularity using Pearson correlation visualizer. Dummies were then dropped and not used for regression.











Pre-processing (1/2)

Applied **target encoding** to transform 8 categorical features using means from the training set data.

1. Season means (popularity) will replace season categories.
2. Month means (popularity) will replace month categories.
3. Modality means (popularity) will replace mode categories.
4. Explicitness means (popularity) will explicitness categories.
5. Key means (popularity) will replace key categories.
6. Modality means (popularity) will replace mode categories.
7. Scale means (popularity) will replace scale categories.
8. Instrumental groups (popularity) will replace instrumentalness.

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 19416 entries, 0 to 19748
Data columns (total 8 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   artists                 19416 non-null  object
 1   modality                19416 non-null  object
 2   key_name                19416 non-null  object
 3   explicit_detailed       19416 non-null  object
 4   instrumentalness_groups 19416 non-null  object
 5   scale                   19416 non-null  object
 6   release_month           19416 non-null  object
 7   release_season          19416 non-null  object
dtypes: object(8)
memory usage: 1.3+ MB
```

```
release_season
Fall        0.590752
Spring      0.599563
Summer      0.617804
Winter      0.563598
```

```
release_month
Apr    0.591183
Aug    0.618209
Dec    0.511941
Feb    0.623445
Jan    0.565928
Jul    0.613778
Jun    0.620769
Mar    0.584581
May    0.623063
Nov    0.557107
Oct    0.611626
Sep    0.601819
```

```
explicit_detailed
Known to be Explicit        0.626210
Not Explicit or Unkmown     0.575851
```

```
key_name
A     0.588178
A#    0.602644
B     0.577196
C     0.580888
C#    0.599136
D     0.581571
D#    0.584408
E     0.593166
F     0.604042
F#    0.603627
G     0.581205
G#    0.603152
```

```
modality
Major    0.589274
Minor    0.594483
```

```
scale
A Major     0.594639
A Minor     0.578377
A# Major    0.607895
A# Minor    0.597927
B Major     0.580414
B Minor     0.574596
C Major     0.572452
C Minor     0.607133
C# Major    0.598167
C# Minor    0.601439
D Major     0.579720
```

```
instrumentalness_groups
High      0.471395
Low       0.602602
Medium    0.504062
```

Pre-processing (2/2)

8. Artists *
   - Initially thought to drop this along with the name of the song.
   - Artist means (popularity) will replace artist name when there are 3 or more rows from the same artist in the training set.
   - Otherwise, will replace the artist's name with the overall popularity mean of the training set.
   - In the testing set, an unknown artist will be replaced with the overall popularity mean of the training set .

** This solution has 2 unsolved problems :*

a. Does not handle multiple artists on the same track (i.e. song by Bruno Mars and Beyonce). A combination is treated as a separate artist.

b. Does not take time into account (the popularity of a track from a specific artist should only be influenced by the popularity of the artist's previous tracks).

```
artists                                                                       
"Adolescents Orquesta"                                                        0.660
"Aulii Cravalho"                                                              0.620
"Aulii Cravalho", Vai Mahina, "Olivia Foai", "Opetaia Foai", Matthew Ineleo   0.620
"Bears Den"                                                                   0.515
"Childrens Music"                                                             0.580
                                                                              ...
```

Linear Regression

```
shape of original dataset : (19416, 19)
shape of input - training set (15532, 18)
shape of output - training set (15532,)
shape of input - testing set (3884, 18)
shape of output - testing set (3884,)
```

Metric	Train	Test
Mean Absolute Error (MAE)	0.076439	0.077052
Mean Squared Error (MSE)	0.013383	0.013103
Root Mean Square Error (**RMSE)**	0.115688	0.114470
R2	0.384753	0.341499

For popularity (unscaled) multiply RMSE X 100
(i.e. 11.5688 out of 100)

```
The model intercept is: -1.029784392251027
The model coefficients are: [ 0.03676127  0.006649   -0.08218222 -0.06348338 -0.01094235 -0.05537881
 -0.02032929 -0.12359748  0.04384836 -0.09498588 -0.40543486  0.79260753
  0.03119926 -0.31567021  0.71928881  0.52637467  0.79034135  0.85798357]
```

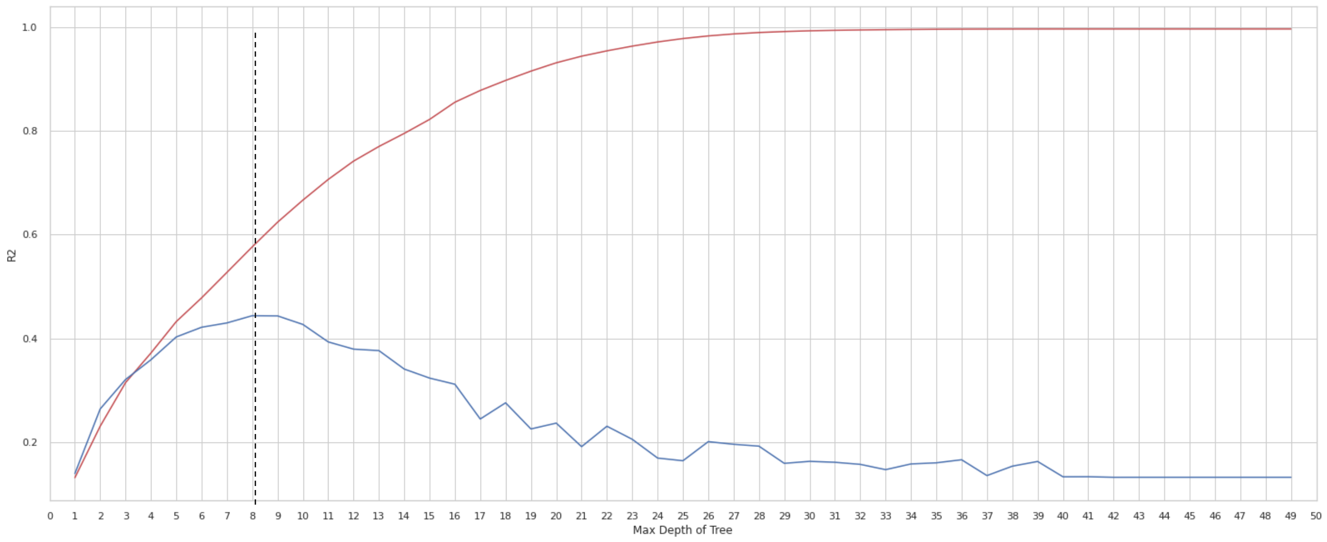
Regression Tree

Results for Regression Tree (max depth = 8)

Metric	Train	Test
Root Mean Square Error (RMSE)	0.096005	0.104603
R2	0.576377	0.450477

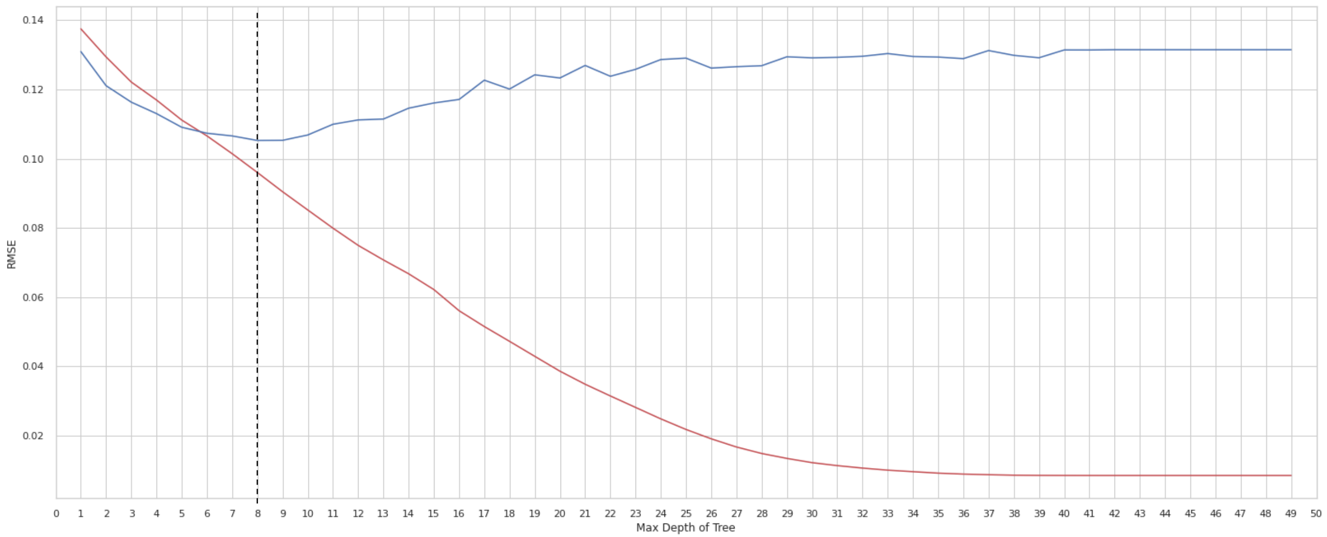
# of Leaves = 140

R^2 vs. Max Depth of Tree



Optimal R2 score is 0.44363983549684305 when max depth = 8

RMSE vs. Max Depth of Tree



Optimal RMSE score is 0.10525253506234485 when max depth of tree = 8

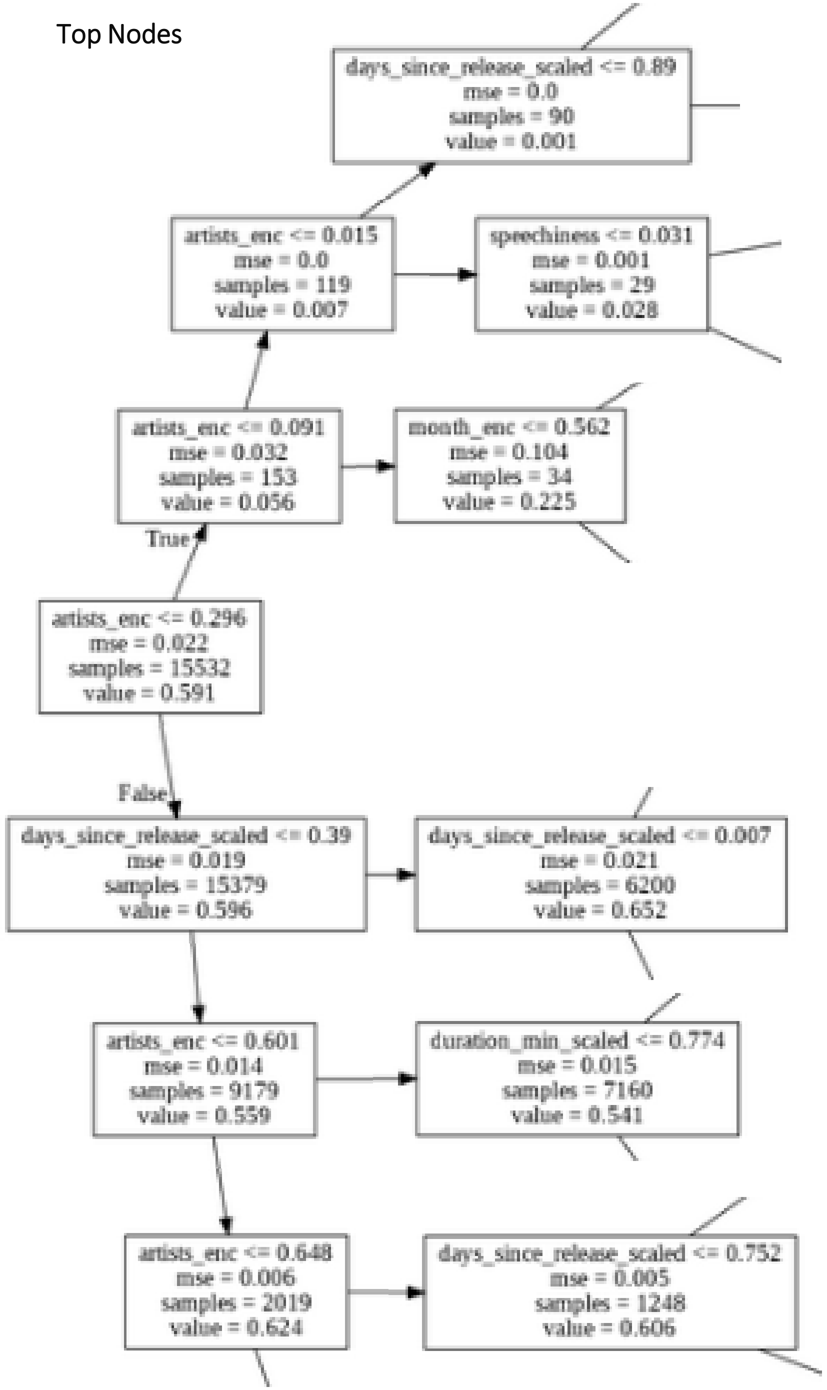
Regression Tree

Results for Regression Tree (max depth = 8)

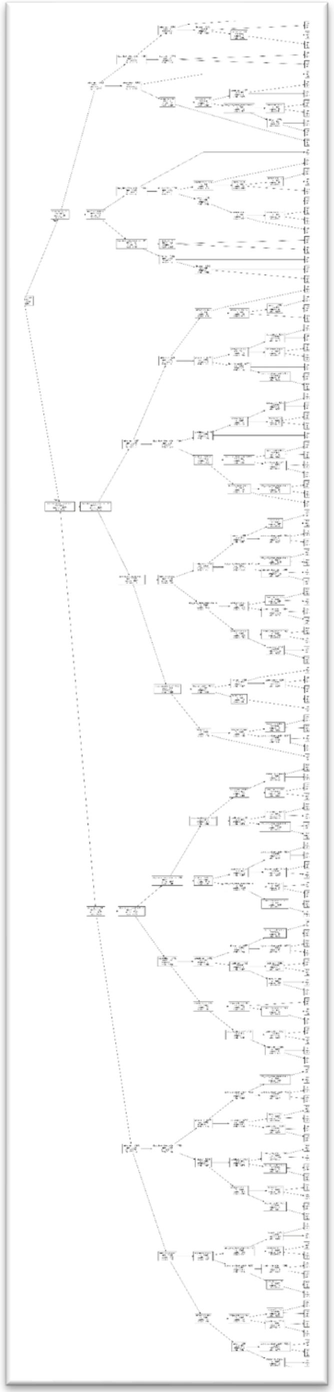
Metric	Train	Test
Root Mean Square Error (RMSE)	0.096005	0.104603
R2	0.576377	0.450477

# of Leaves = 140

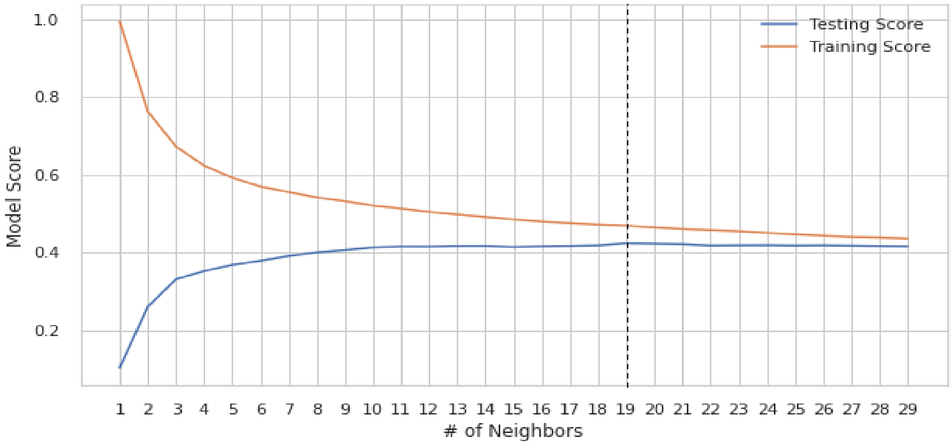
Top Nodes



Full Tree



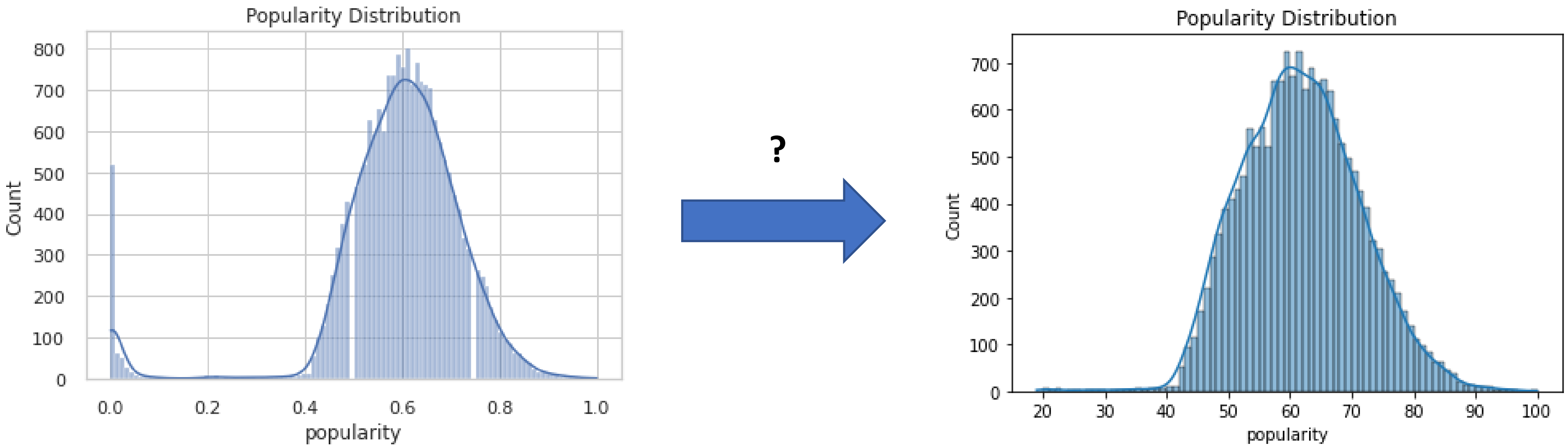
K Nearest Neighbors



K (# of nearest neighbors)	Root Mean Square Error (RMSE)		Accuracy Score		
Metric	Train	Test	Train	Train (CV=5)	Test
5 (default)	0.094228	0.112206	0.591915	0.376604	0.367696
11 (GridSearch)	0.103885	0.107961	0.512561	0.402489	0.414629
19 (Loop)	0.107525	**0.107156**	0.468614	0.396945	0.423332

Removing Outliers from Target

Remove outliers (3 stdev) from target?



Not sure we can allow a track to have a popularity of zero:

- Could be an error
- Track could be new
- Track may not have enough clicks for a score
- Other reasons

Linear Regression

With Popularity Outliers

Metric	Train	Test
Root Mean Square Error (**RMSE)**	0.115688	0.114470
R2	0.384753	0.341499

Without Popularity Outliers

Metric	Train	Test
Root Mean Square Error (**RMSE)**	0.071141	0.073387
R2	0.433742	0.381877

Regression Tree

With Popularity Outliers

Metric	Train	Test
Root Mean Square Error (RMSE)	0.096005	0.104603
R2	0.576377	0.450477

Depth=8

Without Popularity Outliers

Metric	Train	Test
Root Mean Square Error (RMSE)	0.062652	0.073387
R2	0.560811	0.401364

Depth=4

Thoughts and Future Directions

- Does the musical genre influence popularity? Are heavy metal songs more popular than hip-hop songs? (Spoiler: NO!)
- Different musical genres may have different regressions on the various features (i.e. “speechiness” will have a greater correlation to popularity of songs from the genres “rap” and “hip-hop”).
- Many features are highly correlated (loudness and energy, etc.) – is it better practice to reduce the number of features (using factor analysis).
- “Popularity” in Spotify is a score from 0 to 100 – but what makes a track a “hit”? Is there a popularity score threshold (i.e. score over 75?). This can be a classification exercise.
- Will be interesting to predict song success on an external contest (not Spotify) – for example predict a song’s popularity on the “Billboard Hot 100” based on the Spotify song features.

Thank You