Song Popularity Prediction

Can the popularity of the song predicted by using its features?

Elif Sena Özefe | November 2022

elif.ozefe@sabancidx.com

Introduction

If I was working in Spotify's marketing department, I would like to know that which songs deserves more marketing spending. Thus, answer of this question can be the base of budget allocation problem and can be helpful about showing the spending distributions.

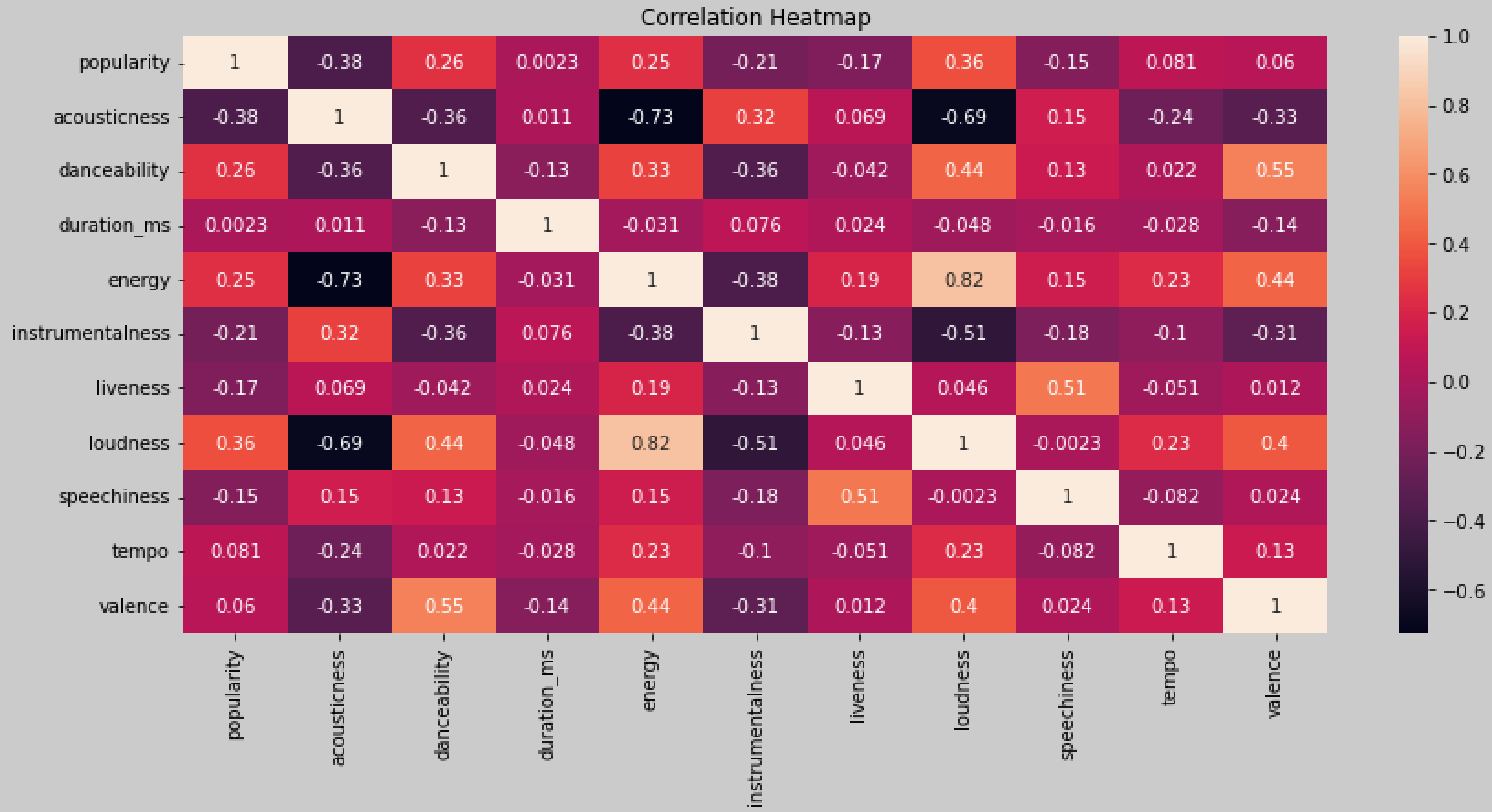
This situatoin rises up the question *"Can the popularity of the song predicted by using its features such as genre, acousticness, danceability, tempo etc.?"*

Data

- To answer the question, I reached the data from kaggle on 2022, 26th October.
- Raw data has 232.725 rows and 18 columns.
- Track_id column is representing tracks but it is not unique.
- Target column is named as popularity.

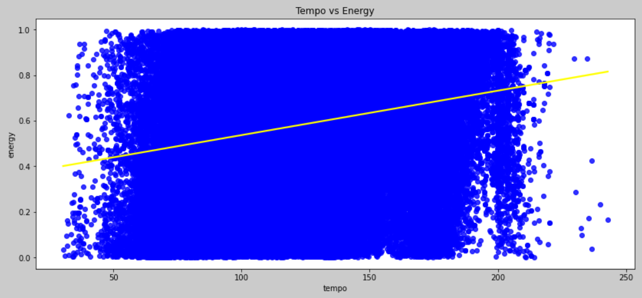
Correlation Map

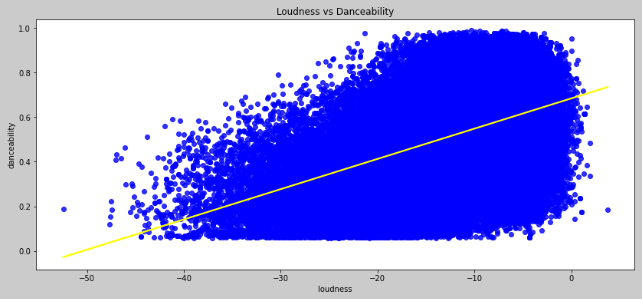
Correlations between the features can be seen from the following chart.

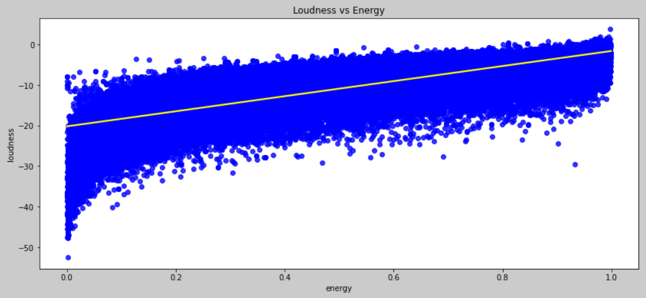


	popularity	acousticness	danceability	duration_ms	energy	instrumentalness	liveness	loudness	speechiness	tempo	valence
popularity	1	-0.38	0.26	0.0023	0.25	-0.21	-0.17	0.36	-0.15	0.081	0.06
acousticness	-0.38	1	-0.36	0.011	-0.73	0.32	0.069	-0.69	0.15	-0.24	-0.33
danceability	0.26	-0.36	1	-0.13	0.33	-0.36	-0.042	0.44	0.13	0.022	0.55
duration_ms	0.0023	0.011	-0.13	1	-0.031	0.076	0.024	-0.048	-0.016	-0.028	-0.14
energy	0.25	-0.73	0.33	-0.031	1	-0.38	0.19	0.82	0.15	0.23	0.44
instrumentalness	-0.21	0.32	-0.36	0.076	-0.38	1	-0.13	-0.51	-0.18	-0.1	-0.31
liveness	-0.17	0.069	-0.042	0.024	0.19	-0.13	1	0.046	0.51	-0.051	0.012
loudness	0.36	-0.69	0.44	-0.048	0.82	-0.51	0.046	1	-0.0023	0.23	0.4
speechiness	-0.15	0.15	0.13	-0.016	0.15	-0.18	0.51	-0.0023	1	-0.082	0.024
tempo	0.081	-0.24	0.022	-0.028	0.23	-0.1	-0.051	0.23	-0.082	1	0.13
valence	0.06	-0.33	0.55	-0.14	0.44	-0.31	0.012	0.4	0.024	0.13	1

Insights



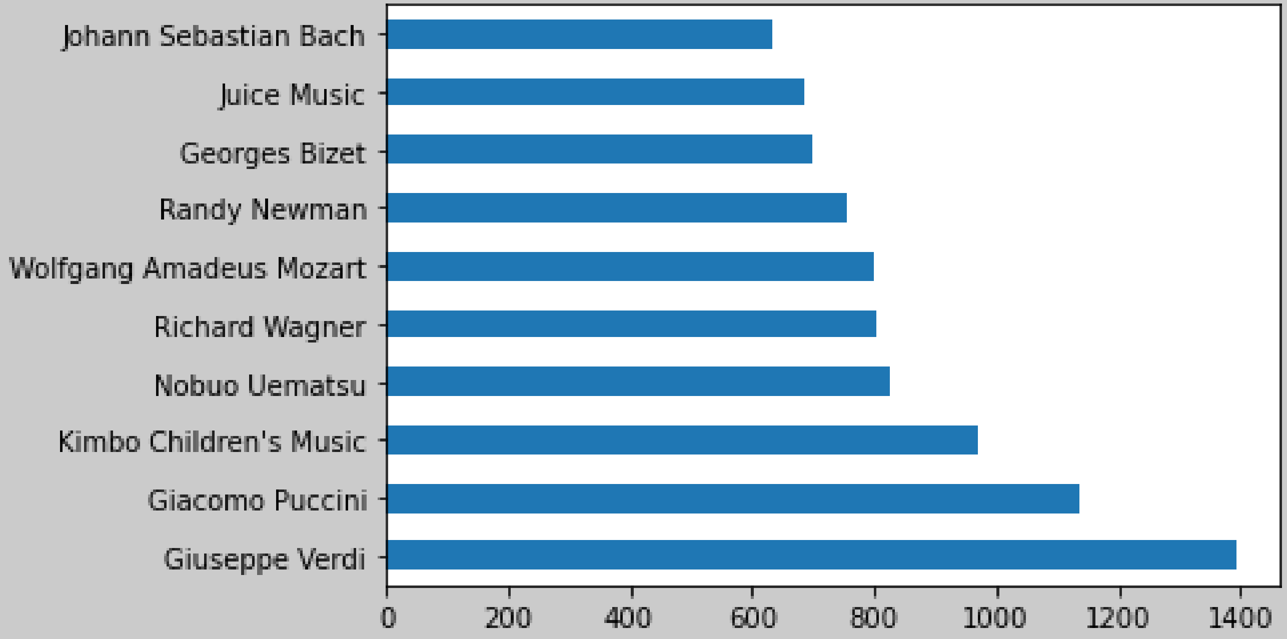




- I checked some variables with high correlation to see if I can find something to feed the model.
- Positive correlations of between the following variables can be seen from the graphs on left;
  - Tempo vs Energy
  - Loudness vs Danceability
  - Loudness vs Energy

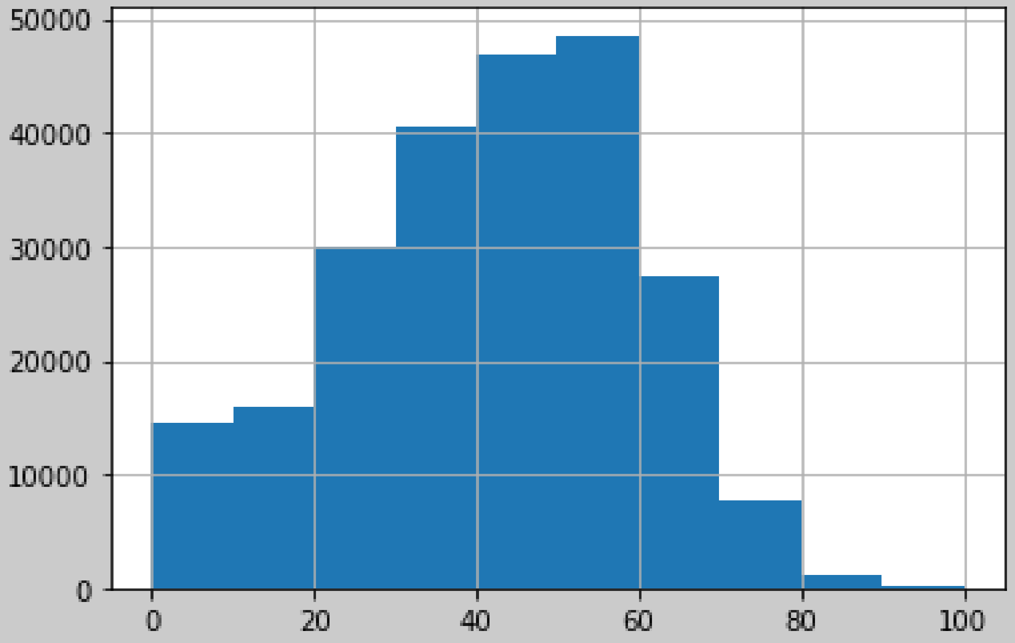
Insights

- Top 5 genres are comedy, soundtrack, indie, jazz and pop, respectively, based on their repeatings in the data.
- Top 10 listening artists, based on their repeating in the data, can be seen in the following chart.



Insights

Distribution of the popularity values can be seen in the following histogram.



Since track_id is not unique, this graph needs to be repeated later.

Preprocess

- Since data does not have any missing values, filling missing step have not needed.
- Dummies generated for genre, key, mode and time_signature columns.
- Track_id is not unique because of genre column. After generating dummies, I have made the data unique based on track_id column. Thus, data has 164.060 unique rows ready for modelling.
- With dummy flags, the data had 54 columns but after applying feature elimination via RFECV, the data has 43 columns ready for modelling.

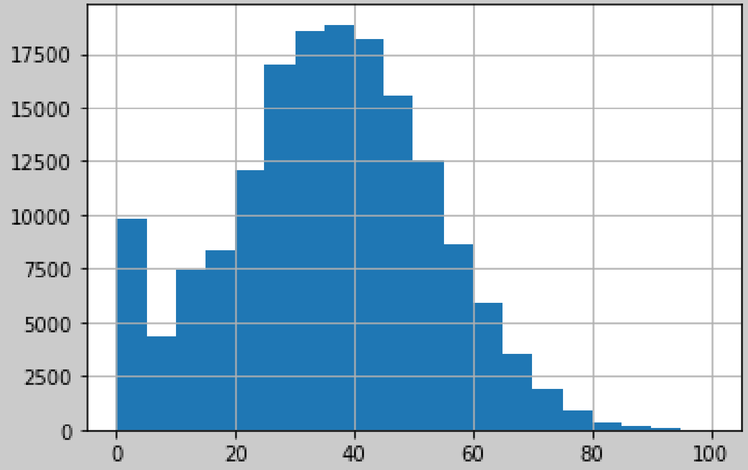
Modelling

Since the popularity column has a large range, I have tried two approaches to answer the question:

- To treat the question as a regression problem
- To treat the question as a classification problem by using bins

Approach I: Regression Problem

- After cleaning the data, distribution of it can be seen from the following histogram.



- Logistic Regression with default parameters ended up with mean absolute error as 35.
- Since this error ratio is too high, I have decided to continue with the second approach.

Approach II: Classification Problem

- I have labeled the songs based on the following rules by using pandas' cut() function;
  - if popularity < 25, «low»,
  - if 25 < popularity < 50, «mid»,
  - if 50 < popularity, «high».
- Ranges are not equal to avoid the imbalance data problem.
- After applying rules, 45.119 low popularity, 87.834 mid popularity and 31.107 high popularity tracks are ready to modelling phase.
- Before training the models, I have splitted 70% of the data as trainset and 30% of it as testset (49.218 rows). I have used stratify parameter to ensure that labels distribution is equal on both on trainset and testset.

Approach II: Classification Problem

Model*	Accuracy
Decision Tree	0.7
Random Forest	0.79
XGBoost	0.79
LightGBM	0.79
SVM	My PC is not enough to run the algorithm.
Naive Bayes	0.5

Even though 3 models have the same accuracy, their other success metrics are not equal and **LightGBM** is the champion model with its metrics**.

* All models run with default parameters.

** Detailed metrics can be found on Jupyter notebook.

Results

- As final step, I have run GridSearchCV with LightGBM and with cv as 5 and scoring as ‘f1_weighted’. GridSearchCV have run to find the best combination among;
  - learning_rate = [0.1, 0.5, 0.9]
  - tree_learner = [serial, future, data, voting]
  - max_depth = [None, 5]
- After 120 fits for 24 candidates, the one with learning_rate = 0.1 , max_depth = None and tree_learner = serial have stood up as the best model with accuracy equals to 0.79.

Next Steps

- Deep learning methods could be tried for the first approach.
- To feed the model, more features of the songs could search online, such as their lyrics.