STATISTICAL AND PREDICTIVE ANALYSIS OF SONG POPULARITY

Advaith Rao	asr209
Ayush Oturkar	ao586
Falgun Malhotra	fm466
Hsiao-Chun Hung (Jeanie)	hh617
Vanshita Gupta	vg422

TABLE OF CONTENTS

ABSTRACT

In this project, we aimed to build a robust model to predict the popularity of a song based on various characteristics of a song, such as energy, acoustics, tempo, liveness, and danceability. In the data pre-processing stage of the project, we identified and removed duplicates and intuitive outliers in the entries. We conducted Exploratory Data Analysis by analysing correlation of various numeric variables. We created new variables using Feature engineering techniques like binning, normalization and aggregation.

We used a variety of regression approaches, including Vanilla OLS model, OLS model with log transformations and OLS model with Box-Cox transformation. We checked Q-Q plots for residual normality and analysed the performance of the model on the basis of R2, adjusted R2 and Root Mean Squared Error (RMSE). We further analysed Random Forest Regressor and CatBoost Regressor with Grid Search CV for optimal hyperparameters. We experimented to see the best model fit on our data based on the criterion of RMSE.

INTRODUCTION

We utilised Ordinary Least Squares which is a type of linear regression model to predict the popularity of a song based on various features of a song. In an OLS model, the goal is to find the line of best fit that minimises the sum of the squared differences between the observed data and the predicted values. The OLS model assumes that there is a linear relationship between the dependent variable *popularity* and the independent variables (*energy, acoustics, tempo, liveness* etc).

In order to improve the performance of our OLS model, we performed log transformation on the y variable. To apply a log transformation to the dependent variable, you would take the natural logarithm of the dependent variable, which is commonly denoted as ln(y). This transformation can help to normalise the distribution of the dependent variable and stabilise the variance, making it more suitable for use in an OLS regression model. The log-transformed dependent variable can then be used as the response variable in the OLS regression model, and the results can be interpreted in the same way as for a model with a normally distributed dependent variable.

Further we implemented the OLS model with Box-Cox transformation. By using the Box-Cox transformation, the OLS model can still be applied to non-normal data, and the results can be interpreted in the same way as for a model with normally distributed data. A Box-Cox transformation is a type of mathematical transformation that is commonly used to stabilise the variance of a dataset and to make the data more normal, or Gaussian-like, in distribution. It is a useful tool for data preprocessing in regression analysis and other statistical modelling techniques. By transforming the data to be more normal in distribution, the assumptions of the statistical model are more likely to be satisfied, which can lead to more accurate and reliable results. The transformation is defined by the following equation:

$$g_\lambda(y) = \begin{cases} (y^\lambda - 1)/\lambda & \lambda \neq 0 \\ log(y) & \lambda = 0 \end{cases}$$

The parameter λ is chosen to maximise the log-likelihood function, which measures the goodness of fit of the data to a normal distribution.

DATA

Dataset Description

We use the Spotify Audio Features dataset[8] which uses the open-source native Spotify API to collect certain audio features for song popularity analysis. The description for the different variables used in our dataset can be found on Spotify developer API[2] page

Data Overview

	count	mean	std	min	10%	25%	50%	75%	90%	99%	max
acousticness	130663.0	0.342500	0.345641	0.000000	0.002440	0.031600	0.203000	0.636000	0.93000	0.994000	0.996000
danceability	130663.0	0.581468	0.190077	0.000000	0.310000	0.459000	0.605000	0.727000	0.80900	0.919000	0.996000
duration_mins	130663.0	3.543885	2.052584	0.053383	1.912167	2.732042	3.365017	4.017458	4.98717	9.539393	93.500333
energy	130663.0	0.569196	0.260312	0.000000	0.163000	0.396000	0.603000	0.775000	0.89400	0.987000	1.000000
instrumentalness	130663.0	0.224018	0.360328	0.000000	0.000000	0.000000	0.000149	0.440000	0.89300	0.969000	1.000000
key	130663.0	5.231894	3.602701	0.000000	0.000000	2.000000	5.000000	8.000000	10.00000	11.000000	11.000000
liveness	130663.0	0.194886	0.167733	0.000000	0.076200	0.097500	0.124000	0.236000	0.38300	0.895000	0.999000
loudness	130663.0	-9.974006	6.544379	-60.000000	-19.276000	-11.898000	-7.979000	-5.684000	-4.20900	-2.082000	1.806000
mode	130663.0	0.607739	0.488256	0.000000	0.000000	0.000000	1.000000	1.000000	1.00000	1.000000	1.000000
speechiness	130663.0	0.112015	0.124327	0.000000	0.032000	0.038900	0.055900	0.129000	0.29400	0.553000	0.966000
tempo	130663.0	119.473353	30.159636	0.000000	80.426000	96.014000	120.027000	139.642000	160.01800	190.196900	249.983000
time_signature	130663.0	3.878986	0.514403	0.000000	3.000000	4.000000	4.000000	4.000000	4.00000	5.000000	5.000000
valence	130663.0	0.439630	0.259079	0.000000	0.097600	0.224000	0.420000	0.638000	0.81500	0.965000	1.000000
popularity	130663.0	24.208988	19.713191	0.000000	0.000000	7.000000	22.000000	38.000000	53.00000	73.000000	100.000000

1. Majority of songs have a duration of 3.36 mins however some of the songs have a duration greater than 10 mins which needs to be treated. We can see the 99th percentile value to be close to 9.54 minutes. Let's remove all the songs greater than 10 mins of duration.
2. In some songs, *tempo*, time_signature is 0 which is not possible hence a potential anomaly in the data, hence such value will add noise to our model. We take all songs with *tempo* & *time_signature* > 0.
3. In some songs, duration is < 1 min. As per our research songs have a minimum duration of 1 mins else it's not a song it is an abstract of the same. In addition some songs are greater than 10 mins in duration. As per our research we tried to search and found that some songs in the data are pure noise or podcasts which is out of scope for this analysis. Hence we will remove the songs with < 1 min of duration and greater than 10 mins of duration. 10 mins taken from the 99th percentile of data split.
4. We also observe some songs to have 0 popularity. The count is roughly around 13k songs. Hence to reduce biases we can remove these songs from our data.
5. Songs which have liveliness over 0.8 are mostly live performances hence it should be removed.
6. We see that there is no key with a value of -1. This means that all the tracks that we have, contain a well-detected key value.

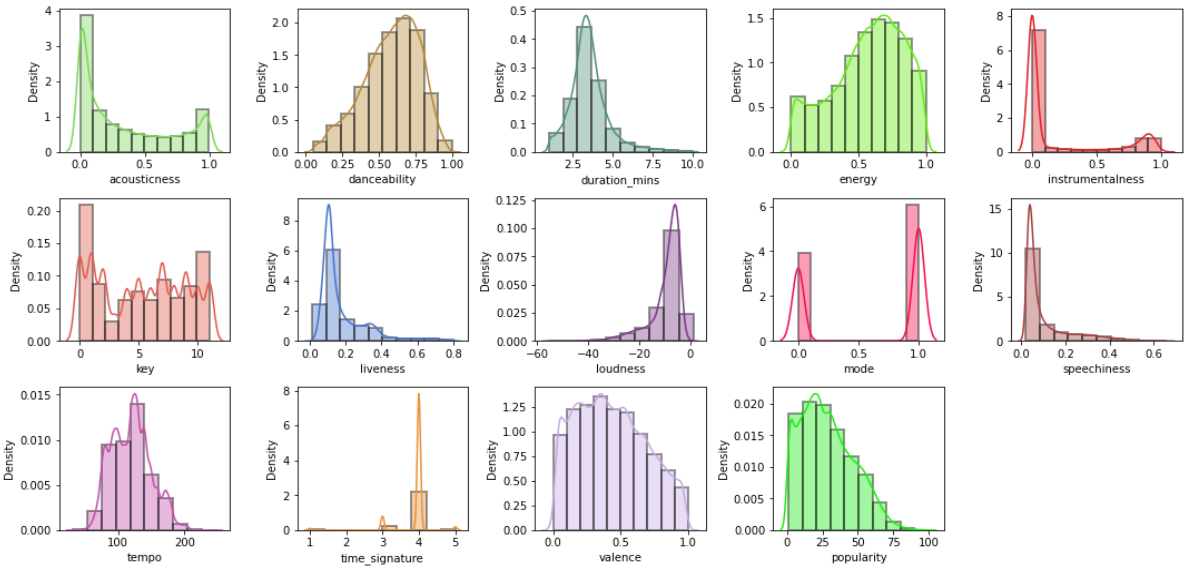
Data Preprocessing

The specific steps involved in data preprocessing in the project include,

1. Removed 337 duplicates on *track_id* level
2. Removed outliers in songs with duration is < 1 min and greater than 10 mins
3. Removed songs with 0 tempo, 0 time signature and songs with 0 popularity
4. Removed songs with speechiness over 0.66
5. Removed songs with liveness less than 0.8

Exploratory Data Analysis

Distribution Of Numeric Features

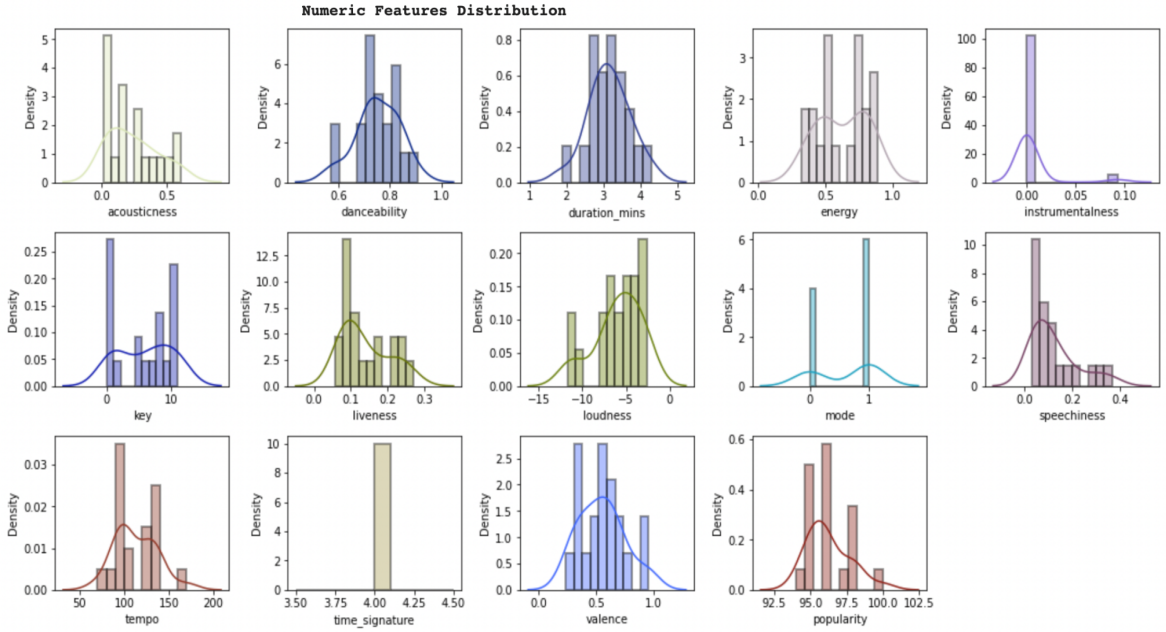


We can observe from the above plots that,

1. **Acousticness** has values that range from 0.0 to 1.0, but most values lie near 0. The songs with high acousticness consist of classical or folk music.
2. **Danceability** has values from 0.0 to 1.0, and we see that most values lie around 0.7, with a small percent of values towards the extreme ends(0 and 1)
3. **Duration_mins** takes values from 1.0 to 10.0, and we can see that most songs range from 3 to 4 minutes.
4. **Energy** consists of values from 0.0 to 1.0, and we see that the density of songs with energy less than 0.5 is very less.
5. **Instrumentalness** has values from 0.0 to 1.0. This variable predicts if a song has vocals(Lower the value, higher the chance of vocals) and we see that most songs have a value near 0, which means most songs have a presence of vocal component in them.
6. **Key** has values from 0 to 12, with values 0 for *C* and 12 for *B*.We can see that most songs are in the keys of *C#* and *C*
7. **Liveness** contains values from 0.0 to 1.0, with higher values meaning more chances of presence of audience in the recording. We see that most songs have a liveness near 0.0, and we will clean our data and remove songs, which can be live performances.

8. **Loudness** has values from 0.0 to 1.0, and we can see most songs have a loudness of around -10dB. Here 0dB is the normal standard for human hearing.
9. **Mode** has 2 values 0 and 1. We see that most songs have a happy tone in them, as they are in the major mode(mode=1), which is usually happy sounding(Except for some)
10. **Speechiness** has values from 0.0 to 1.0, and we can see that most songs have a speechiness of less than 0.25, also we clean out songs with speechiness of more than 0.66, as usually they would be podcasts, intro/outro tracks, and live performances.
11. **Tempo** mostly contains values between 100 and 250, and we can see that the statistical mode for *tempo* is around 125, as people can enjoy the track rhythmically the most with the tempo around 120-130.
12. **Time Signature** contains values 1.0 to 5.0 and we see that most songs have a value of 4.0, which is vital for a song to gain much fan following as it is a universal rhythmic pattern.
13. **Valence** has values from 0.0 to 1.0, and we can see that most songs have a valence of less than 0.5, which induces a lesser tone of happiness in most songs. We can make sense out of this pattern as the most popular key-mode pair from above plots, *C# major* or *Db major*, usually represent a tone of grief or sadness.
14. **Popularity** takes values from 0.0 to 100.0, and we can see that there are only a few songs which are very popular, and there are a lot of tracks which go unheard.

Distribution Of Numeric Features based on Top 20 songs



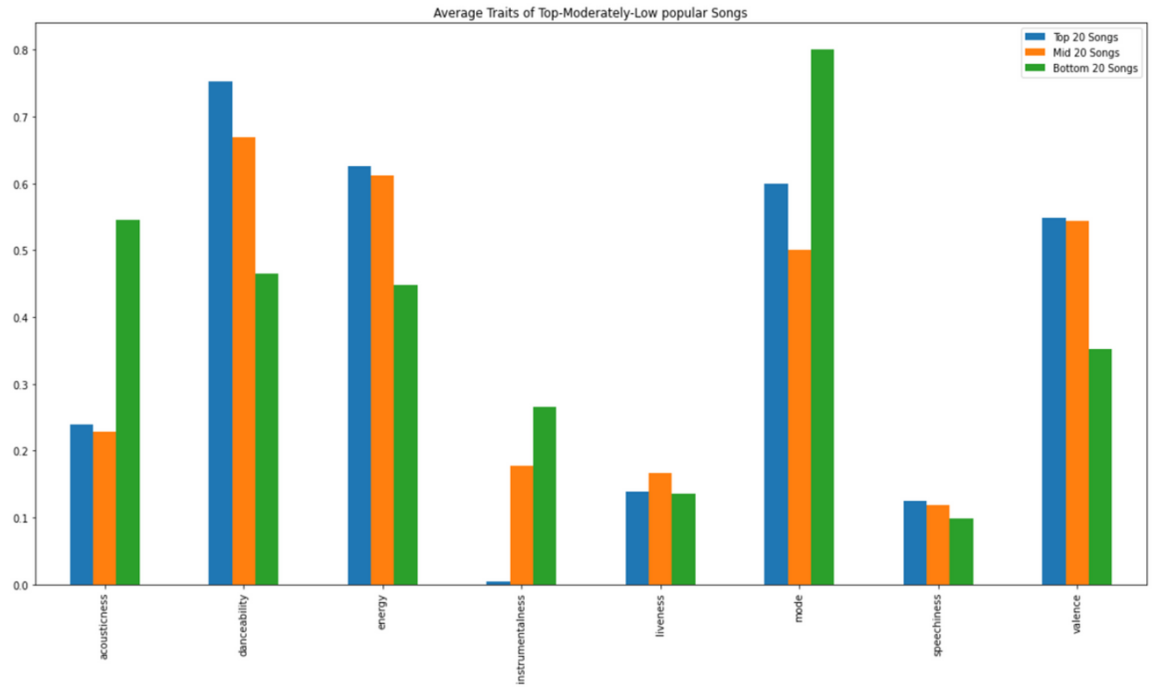
We can see from the above data that the top popular songs have,

1. **Time Signature:** The same time signature of 4, people enjoy rhythmic patterns that they are used to, as 4 is the most universal time signature.
2. **Instrumentalness:** People tend to listen to songs with instrumentalness of value around 0 i.e. people enjoy vocals/lyrical presence than instrumental presence.

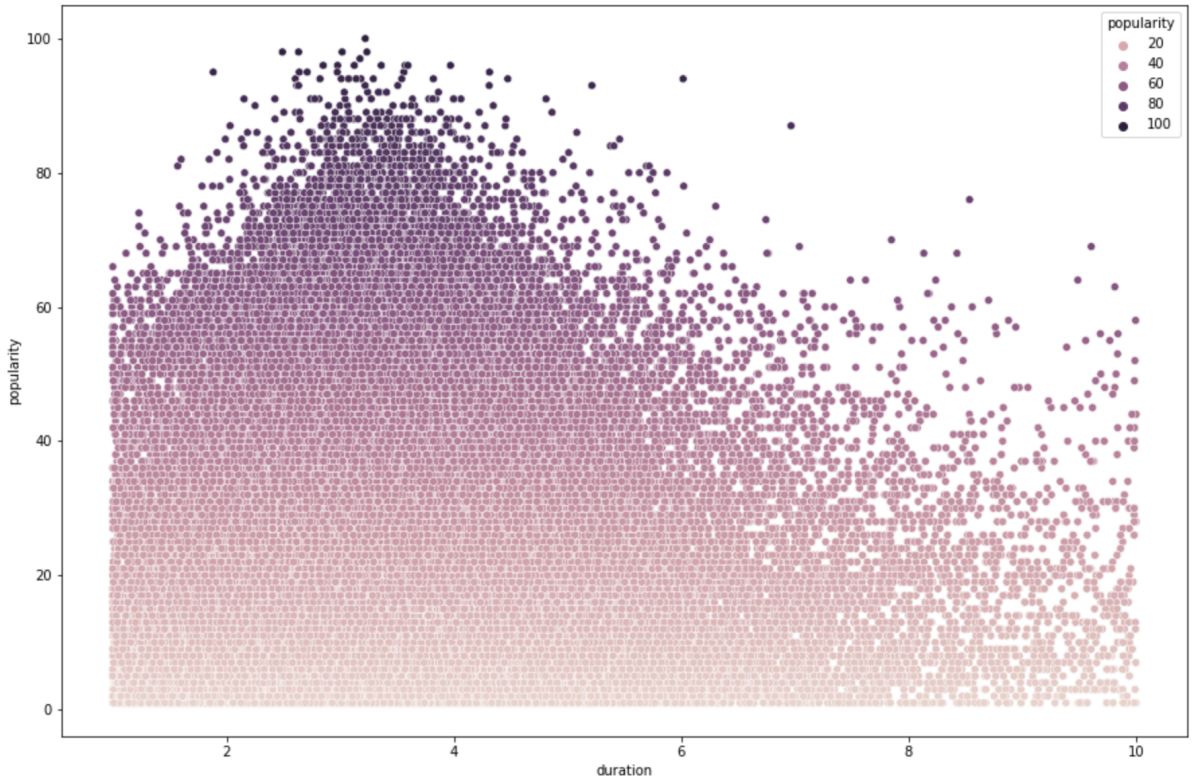
3. **Key:** The most popular songs are seen for key are C,C#, G#, B. Intuitive this make sense as B major has a relative minor of G# minor and both modes sound alike.
4. **Danceability:** Most popular songs have a danceability score of around 0.75-0.85. People listen to more hype music.
5. **Loudness (in scale of log2):** Popular songs tend to have loudness in the range of -7 to -2 while a normal song on average has loudness in the range of -10 to -5. Popular songs tend to have more compression than an average song.
6. **Duration:** Most popular songs have a duration of close to 2.5 mins, this tends to mean that people have less attention span while listening to music or doing anything.
7. **Valence:** As opposed to an average song, the majority of popular songs tend to be more happy sounding, i.e. having a valence of over 0.35, so we see that the most popular songs are usually happy-toned.
8. **Acousticness:** People generally tend to listen to highly electrical music as opposed to clean acoustic tones like classical pieces.
9. **Energy:** High Energy genres like metal and EDM do not usually show up in popular music.
10. **Liveness:** People tend to listen more to clean and compressed songs in comparison to distorted audio.
11. **Tempo:** Most popular songs tend to have a *tempo* less than 150 and more concentrated towards 100.

Comparison Of Audio Features Averages For Top, Mid, Bottom 20 Songs

We created some plots to study the data. We analysed the mean values of the Top, Middle and Bottom 20 songs of all the artists in our dataset for all the significant features of the songs.

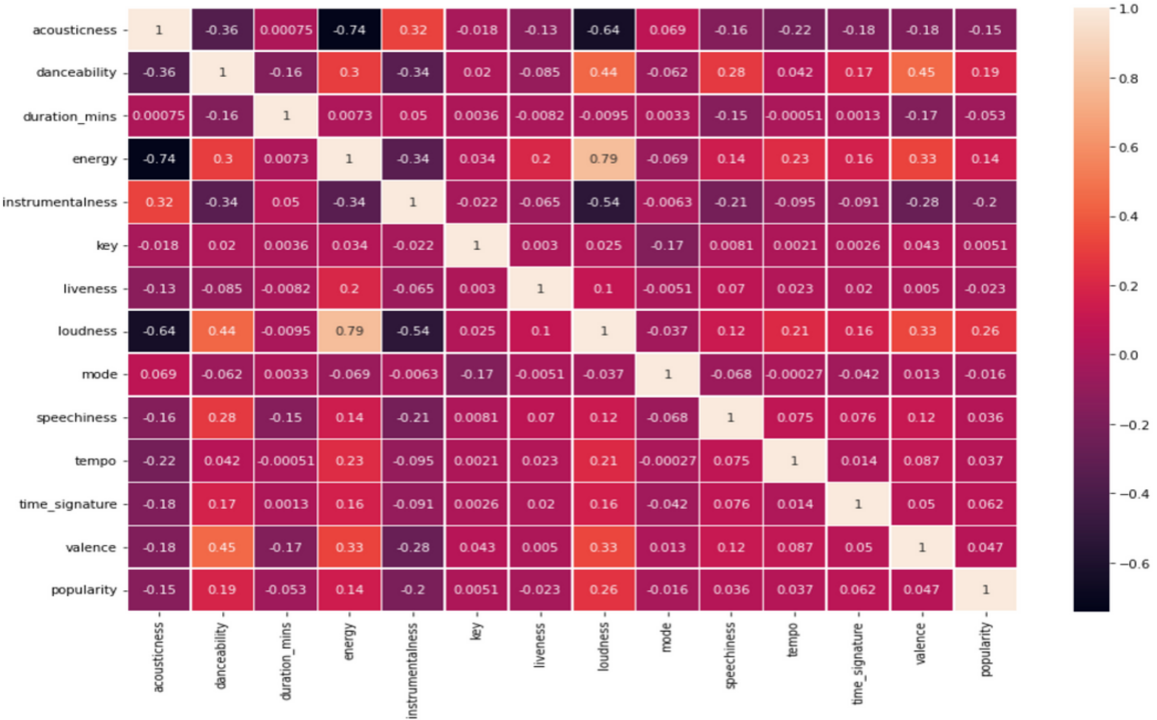


Comparison of song popularity w.r.t duration of the song:



Correlation Heat Matrix Between All Audio Features

We looked at the correlation between all the features of the song. The dark colour indicates the variable is negatively correlated with the respective variable.



Feature Engineering

While we explore the data, we see that,

1. Multicollinearity is significant in the data.
2. *Danceability, energy , tempo* and *time signature* have high VIF values which need to be treated.

Create new variables

We create the following variables based on our intuition and add them to our dataset.

Variable Name	**Description**
Total songs by artists	Number of songs by a given artist
Is new artist	Binary 0 or 1, depending on whether the artist is a new artist
Popular key	Binary 0 or 1, depending on if the song is in a popular key(C, C#, G# or B) or not
Noise level	We will get the difference between energy of a song and its acousticness and will scale it by 0.5, we will also drop energy and acousticness to prevent multicollinearity

Data Cleaning

The following steps were taken to clean the data :

1. Dropped features with high correlation(VIF score above 10 helps us get multicollinearity between variables) including *loudness*, *danceability*, *energy*, *acousticness*, *key*, and *time_signature*.
2. Tracks were removed intuitively. Dropped track_names which have certain keywords pertaining to non-music tracks. Track names including one of the following keywords were removed:
    - ‘mix, rain, podcast, intro, outro, DJ, sleep’
3. The *Tempo* feature was binned based on thresholds. Converted *tempo* variable into a categorical value where
    - *Tempo* less than 76 maps to -1 (Slow-tempo)
    - *Tempo* between 76 and 120 maps to 0 (Medium-tempo)
    - *Tempo* less than 76 maps to 1 (Up-tempo)

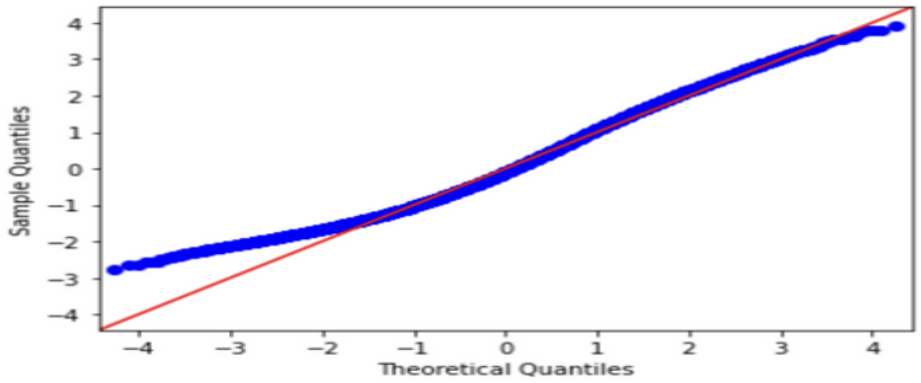
METHODOLOGY

Modelling

Vanilla OLS Model

According to the R^2 value and Q-Q plot below we observe that the R^2 value of 0.694 is a positive indicator that our model fits the data well. The residuals, however, follow a light-tailed distribution, which we must correct with transformations, as seen by the Q-Q plot.

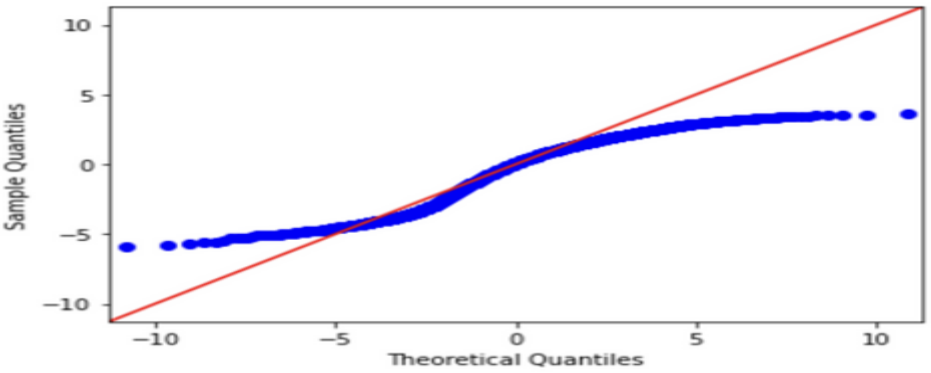
```
R-squared (uncentered):                   0.694
Adj. R-squared (uncentered):              0.694
```



OLS Model with Log Transformation

After the log transformation, the R2 value increased, but the residual distribution deviates even more from the normal distribution. Hence we need to improve our model further. We need to try other transformations like box cox transformation so that residuals follow a normal distribution.

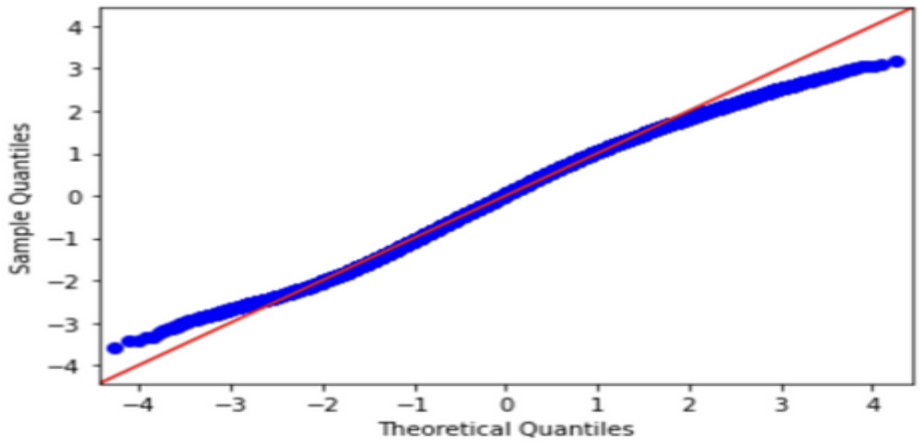
```
R-squared (uncentered):                   0.879
Adj. R-squared (uncentered):              0.879
```



OLS Model with Box-Cox Transformation

As per the results below we can observe that the R square value is 0.789 which indicates a better fit in comparison to the vanilla OLS model. Also the residuals seem to be approximately following normal distribution hence we can use this model as our final OLS model for hypothesis testing and further analysis.

```
R-squared (uncentered):                   0.789
Adj. R-squared (uncentered):              0.789
```



Beta Coefficients And P-Values For The Best Model (OLS Model with Box-Cox Transformation)

	coef	std err	t	P>|t|	[0.025	0.975]
duration_mins	4.5276	0.029	156.571	0.000	4.471	4.584
instrumentalness	0.2364	0.049	4.864	0.000	0.141	0.332
liveness	2.5234	0.115	21.883	0.000	2.297	2.749
mode	0.8211	0.032	26.008	0.000	0.759	0.883
speechiness	4.3791	0.143	30.658	0.000	4.099	4.659
tempo	0.0155	0.002	7.109	0.000	0.011	0.020
valence	3.6915	0.059	62.983	0.000	3.577	3.806
total_songs_by_artist	-0.0016	3.37e-05	-47.983	0.000	-0.002	-0.002
is_new_artist	-0.9996	0.043	-23.149	0.000	-1.084	-0.915
popular_key	0.7069	0.032	21.905	0.000	0.644	0.770
noise_level	0.5742	0.067	8.569	0.000	0.443	0.706

Hypothesis Testing for OLS Model with Box-Cox Transformation

Songs in Major mode are more popular than ones in minor

$$\mathbf{H_0}: \beta_{\text{mode}} <= 0 \text{ Vs } \mathbf{H_1}: \beta_{\text{mode}} > 0$$

We see from our model that, our p-value is close to 1, hence we FAIL to reject our null hypothesis, and conclude that based on our data, songs in minor mode are more popular

For a song to be popular, we need it to be in a popular key

$$\mathbf{H_0}: \beta_{\text{popular\_key}} <= 0 \text{ Vs } \mathbf{H_1}: \beta_{\text{popular\_key}} > 0$$

We hypothesise that, for a song to be popular, it has to be in a key of C, C#, G#, B(with key values of 0,1,8,11 respectively). We see from our model that, our p-value is close to 1, hence we FAIL to reject our null hypothesis, and conclude that based on our data, songs do not have to be in the popular key to be popular.

Songs with high noise are most popular

$$\mathbf{H_0}: \beta_{\text{noise\_level}} <= 0 \text{ Vs } \mathbf{H_1}: \beta_{\text{noise\_level}} > 0$$

We see from our model that our p-value is close to 1, hence we FAIL to reject our null hypothesis, and conclude that based on our data, songs with high noise are not the most popular ones.

Most people like listening to songs of short duration.

$$\mathbf{H_0}: \beta_{\text{duration\_mins}} >= 0 \text{ Vs } \mathbf{H_1}: \beta_{\text{duration\_mins}} < 0$$

We see from our model that, our p-value is close to 0, hence we can reject our null hypothesis, and conclude that based on our data, the longer the duration of a song, the lesser its likelihood of being popular.

An artist with a lot of songs is more popular than one with lesser number of songs

$$\mathbf{H_0}: \beta_{\text{total\_songs\_by\_aritst}} <= 0 \text{ Vs } \mathbf{H_1}: \beta_{\text{total\_songs\_by\_aritst}} > 0$$

We see from our model that, our p-value is close to 1, hence we FAIL to reject our null hypothesis, and conclude that based on our data, an artist with more songs, is not necessarily always more popular

Model Improvements

We also tried other complex ML techniques to get a better fit over the data. These techniques, even though they provide a better RMSE score, have an issue with model explainability as they represent our target as a black box function of our input variable.

These were the results we observed for the different models we fit on our training data to check for RMSE scores on the test set.

Model	**RMSE on test set**
Best OLS Regressor	4.44
Random Forest Regressor	4.03
CatBoost with GridSearchCV	3.81

CONCLUSION

We observe non-parametric models such as random forest and catboost to capture the complex dataset and fit the model more precisely to predict the popularity of the songs. For further improvements in prediction accuracy, we can also try to fit deep neural network models. However, since these models are black models and lack explainability, it is always a better idea to use OLS models, which allow us to test our hypothesis.

The following are the main takeaways from the data analysis:

- Significant features according to the OLS model include song duration, instrumentalness, liveliness, mode, speechiness, tempo, valence, total songs by the artist, whether the artist is a new artist or not & popular key binary flag.
- Songs in minor mode tend to be more popular
- Songs with higher noise are not the most popular ones.
- The longer the duration of a song, the lesser its likelihood of being too popular, which indicates that people prefer listening to shorter songs.
- An artist with more songs, is not necessarily always more popular

Future scope of improvement in our analysis:

- We need more samples of data so that it may help improve the models.
- We require more variables which help explain the data better such as Genre, playlist of the track, lyricist, song director.
- Time stamps are missing from our data. An important predictor of future success is the artist's previous song chart success.
- We can leverage Natural Language Processing to extract useful information out of track names and lyrics.
- Audio processing to get more features out of the audio data
- We could make use of popularity of featured artists

REFERENCES

[1] https://github.com/tgel0/spotify-data - Github repository containing track information data fetch from Spotify using Spotify API

[2] https://developer.spotify.com/documentation/web-api/reference/#/operations/get-audio-features - Description of the different variables we use in our dataset.

[3] https://www.frontiersin.org/articles/10.3389/fpsyg.2018.02118/full - Effects of tempo as an emotional feature on music listeners. We used this as a reference to binning tempo into categorical variables of slowtempo(-1), mediumtempo(0) and uptempo(1)

[4] https://github.com/scikit-learn/scikit-learn - Scikit-Learn package for data cleaning and modeling in python.

[5] https://github.com/statsmodels/statsmodels - Statsmodels package for Regression.

[6] https://github.com/catboost/catboost - Catboost package for fast, scalable Boosting-based Regression in Python.

[7] https://desktop.arcgis.com/en/arcmap/latest/extensions/geostatistical-analyst/normal-qq-plot-and-general-qq-plot.htm - Reference for using Q-Q plot to test for residual normality.

[8] https://www.kaggle.com/datasets/tomigelo/spotify-audio-features - The dataset we used for this project can be found here.