Do Popular Songs Endure?

Final Report by Michael Alvin, Shreya Majumder, Srishti Bhalla

Introduction

Predicting whether a song will be popular is a common problem, but predicting whether popular songs endure is a different, more fascinating problem. Our project focuses on the latter problem, that is we are exploring popular songs throughout the years back to the 1960s to observe whether those songs are still popular today and answer the question of why some popular songs stay popular and why some songs decline in popularity.

After scraping data of popular songs through Billboard Weekly Hot 100 Data, we built a baseline model that predicts a song's current popularity based on two features: years released and peak rank. In our project, we used Spotify popularity data as a measure of a song's current popularity. The baseline model is kept simple to gain a better understanding of the problem statement and to see whether there is a decay based on years released and whether a song's peak rank has an effect on its current popularity. For the most part, we predicted song's popularity decline over the years and a song who reached peak popularity is more likely to be popular today than a song who did not. After building the baseline model, we performed a sniff test to check why a few songs overperform (a song is more popular today than predicted) and underperform (a song is less popular today than predicted), and we explored the features of such outliers to see what makes them different.

In this report, as suggested by Professor Skiena and TAs, we kept our model simple and used the least number of features that will maximize our accuracy. We have avoided audio features such as tempo, beats per minute, valence, and focused on features we believe will play a more significant role. After our sniff test and observation of outliers, we decided to include **artist popularity**, **genre**. Our report describes more in detail our sniff test and our reasoning to include those features in our model.

1 Get Data

Scraping Data

For our project, we have gathered data from the following sources:

- **Billboard.com:** Billboard.py is a Python API for accessing music charts on Billboard.com. We gathered songs on Billboard Weekly Hot 100 Data from 1960 to 2017. From here, we retrieved the song's rank, name, artist, peak position, weeks on chart, and the date.
- **Spotify:** Spotipy is a lightweight Python client library for the Spotify Web API. We retrieved Spotify popularity scores, artist popularity and genres for the songs gathered from Billboard.

Measure of Contemporary Popularity

Initially, we wanted to combined Spotify popularity and Youtube View Count to create a single popularity index. However, we decided to discard Youtube View Count for two reasons: (1) we wanted to keep our model simple, (2) Youtube View Count may not be indicative of current popularity because songs may obtain most of its views near its release and not at recent time. A few things we had to keep in mind for Spotify scores:

- **Normalization:** As per Spotify API documentation, Spotify popularity score is calculated by algorithm and is based on the total number of plays the track has had and how recent those plays are. However, because the values range between 0 and 100, these scores are most likely normalized.
- **Zero:** While gathering Spotify popularity scores for songs, we observed that many songs have 0 scores. As Professor Skiena suggested, these scores suggest several possibilities: these songs are old and unpopular, these songs do not have an agreement with Spotify, etc. We do not know the nature of these songs, so we removed songs with Spotify popularity score of 0.

2 Data Pre-Processing

Data Cleaning

- **Duplicates:** Many songs appear more than once on Billboard Weekly Hot 100. Therefore, we removed rows with the same ‘Song [Name]’ and ‘Artist’, because they represent the same songs.
- **Merging:** While removing duplicates using df.drop_duplicates, we performed the process using keep=’first’ and keep=’last’. The keep=’first’ is to retrieve its first appearance on Billboard, as we assumed its year of release on the data of first appearance on the list. The keep=’last’ is to retrieve the peak position and weeks on chart, because the correct values for those metrics are capture in its last appearance on the list. Then, we simply merged those two results based on ‘Song [Name]’ and ‘Artist’.
- **Zeroes:** Some songs do not have Spotify popularity scores, and we removed these songs from our data as we do not know the nature of these songs, whether they are outliers or not.

Features

As mentioned, we have limited our project to the least number of features to obtain the best model to predict a song’s current popularity. Keeping that in mind, here are our features:

Billboard Features

1. **Song:** Name of the song.
2. **Artist:** Artist of the song.
3. **Date:** Date of the song appeared on Billboard Weekly Hot 100.
4. **Peak Position:** Highest position of the song on Billboard Weekly Hot 100 in its lifetime.
5. **Weeks on Chart:** Number of weeks of the song appearance on Billboard Weekly Hot 100.

Spotify Features

6. **Spotify Popularity:** Contemporary popularity of the song.
7. **Artist Popularity:** Popularity of the song’s artist.

Derived Features

8. **Artist Count:** Count of the song’s artist’s songs appearing on Billboard Weekly Hot 100.

3 Evaluation and Analysis

Data Analysis

Before building our baseline model, we wanted to see in our dataset if there is a correlation between year of release versus current popularity and between peak ranks versus current popularity. In these two plots, we plotted the median current popularity per year of release (figure 1) and per peak rank (figure 2). As shown in figure 1, the general trend is that older songs are less popular now than newer songs. As shown in figure 2, the general trend is that songs with higher peaks (1 - 100, where 1 is the peak position) are more popular today than songs with lower peaks. This aligns with our predictions and we can safely proceed to modeling.

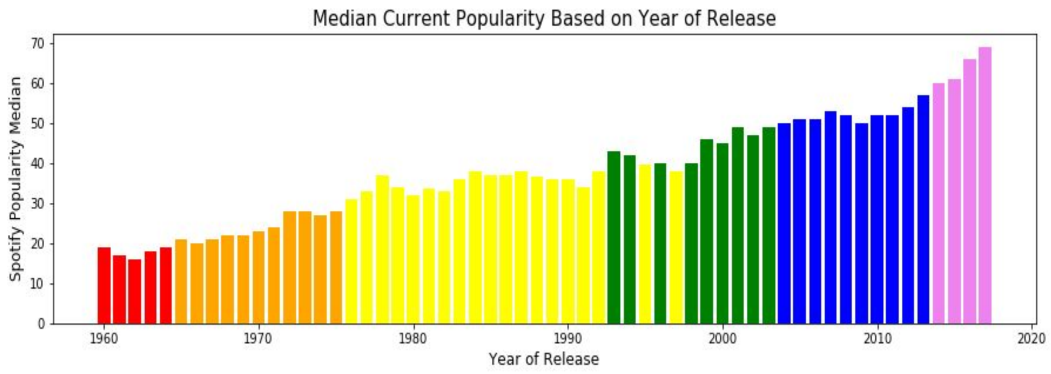


Figure 1

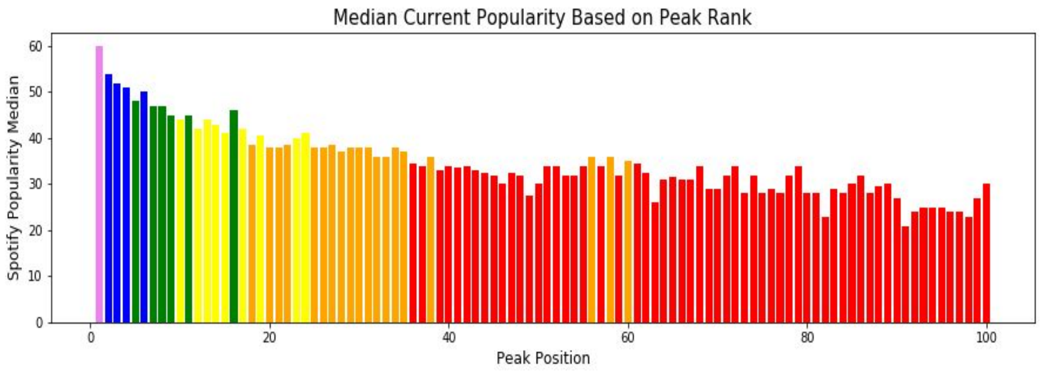
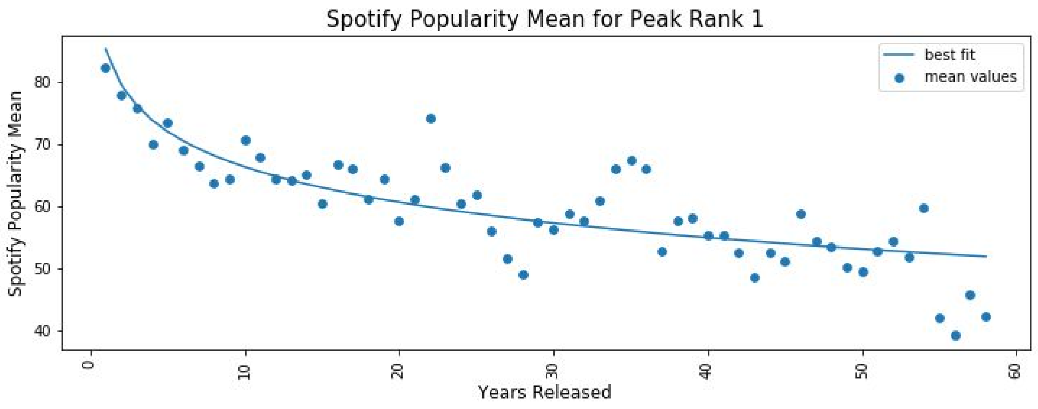
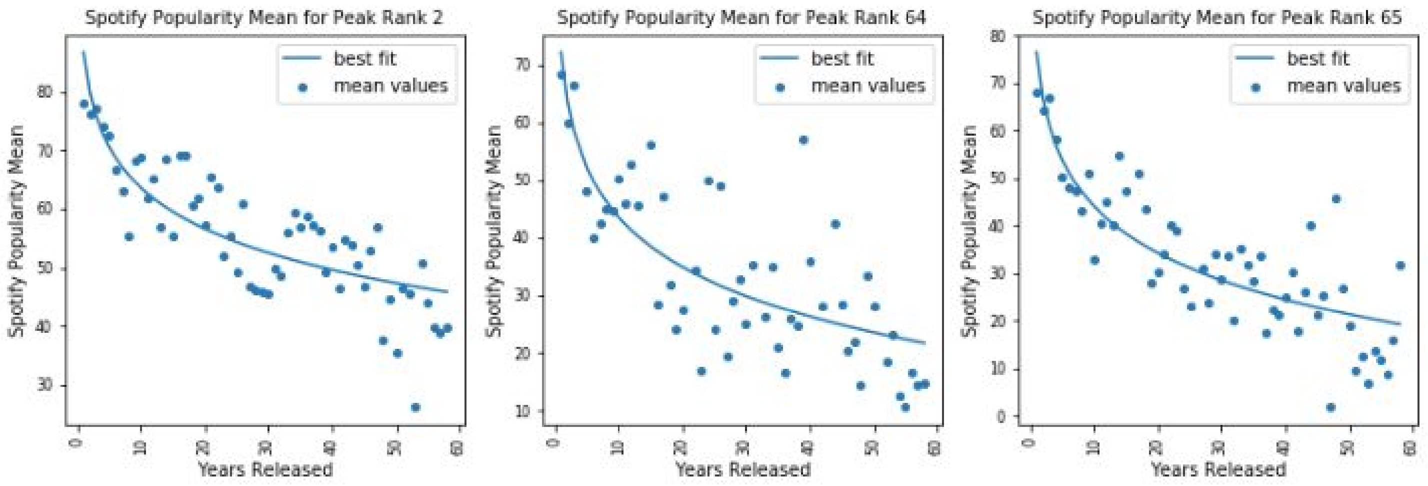


Figure 2

Years Released

Our first approach is to define the decay function illustrated by figure 1. As suggested by Professor Skiena and TA Sahil, we will consider several peak ranks, one at a time, and plot the median current popularity per year for that peak rank and getting the line of best fit which may be linear, exponential, etc. By doing so, we are keeping the peak rank as a constant, and years released will be our only independent variable. Please note that years released is the inverse of year of release.





Base Model, 1

With the insight we have found from the previous section, we were able to detect a logarithmic decay function in terms of years released and Spotify popularity mean. Our base model has the form y = a + b log x, where y is predicted Spotify popularity, x is years released, and a, b are coefficients.

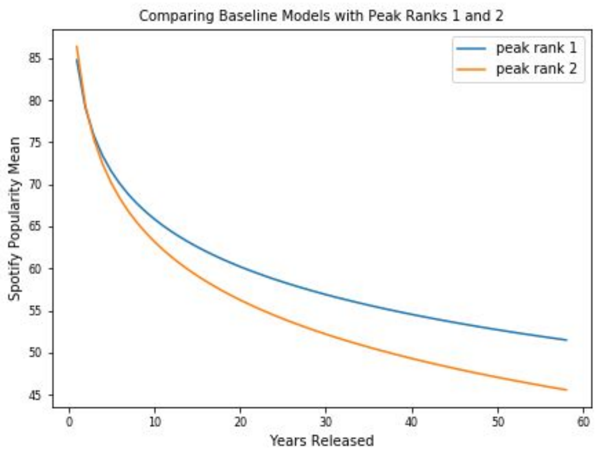
Base Model Evaluation, 1

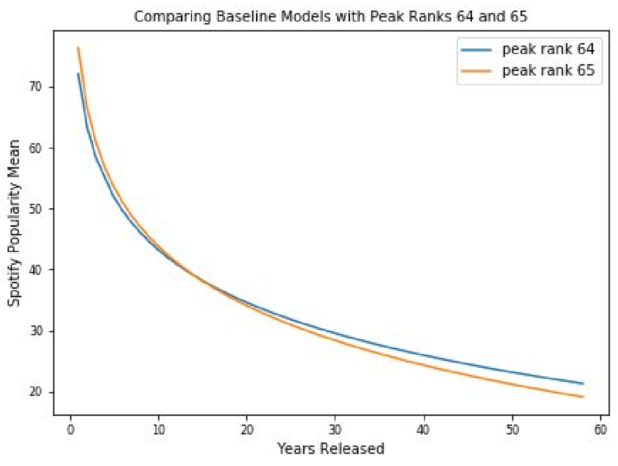
For base model, we trained the model based on the mean values of years released. To evaluate its performance, we ran the base model on in mean values and the individual values. Mean values refers to the plots above or the mean value of songs grouped by years released (one per years released), and individual values refers individual songs (many per years released). From our preliminary results, the errors on mean value are not substantial. However, error on individual values are relatively large, and this makes sense, because our base model trains on mean values and therefore does not consider outliers in our dataset. Therefore, our next step is to perform the sniff test to figure out why some songs over-perform/underperform.

y = a + b log x	a	b	Error on Mean Value	Error on Individual Values
Peak Rank 1	85.15	-8.20	5.03	12.63
Peak Rank 2	86.77	-10.07	6.40	13.37
Peak Rank 64	72.21	-12.44	9.37	14.04
Peak Rank 65	76.58	-14.10	8.17	14.81

Peak Rank

Naturally, we believe that the peak ranks make a difference in the current popularity in such a way that songs that reach higher peaks, say number 1 on Billboard Weekly Hot 100, will be more popular today than songs with lower peaks, say number 5 on the list. However, as discussed in lecture, the difference are not linear in that the popularity of numbers 1 and 2 have a greater difference than the popularity of numbers 64 and 65. Below we have shown the graph, which supports our claim, because peak rank 1 songs is substantially more popular than peak rank 2 songs, but peak rank 64 songs are slightly more popular than peak 65 songs.





Base Model, 2

With our observations of difference of current popularity between ranks, we hope to include rank in our initial base model $y = a + b \log x$. Because the difference between ranks decreases as ranks become higher, we added another logged parameter to the base model, $y = a + b \log x_1 + c \log x_2$, where x_2 is the peak rank.

Base Model Evaluation, 2

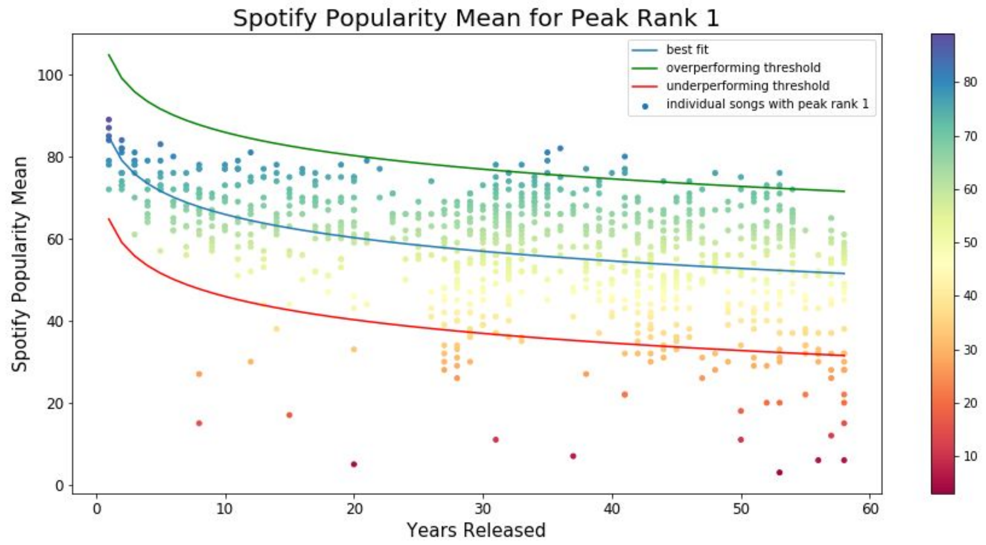
Because now we are training on individual songs, we evaluated our model using cross validation (80% train, 20% test). We compared the performance of $y = a + b \log x_1 + c \log x_2$ and $y = a + b \log x$ below. As seen below, including the peak rank indeed improves the performance of our model, by about 3 Spotify popularity points. Training error may be larger than testing error, because it's likely that there are more outliers in the training set, which contains more songs overall. Now that we are done with our base model with years released and peak rank, the next step is to perform the sniff test, studying why some songs over-perform/underperform and incorporating new features to our model.

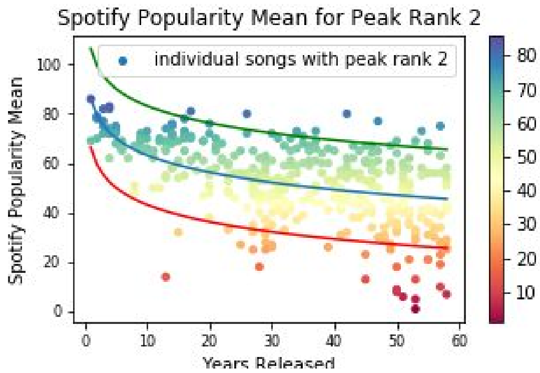
	Training Error	Testing Error
$y = a + b \log x$	16.32	16.27
$y = a + b \log x_1 + c \log x_2$	13.71	13.66

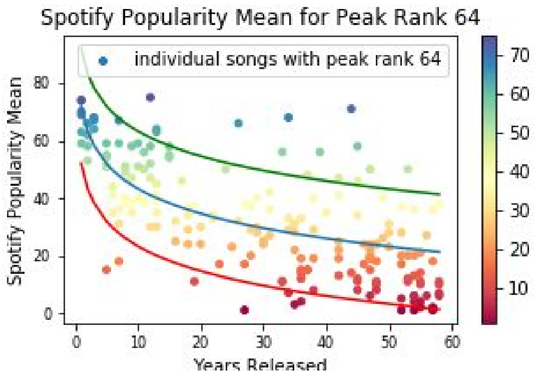
4 Evaluate

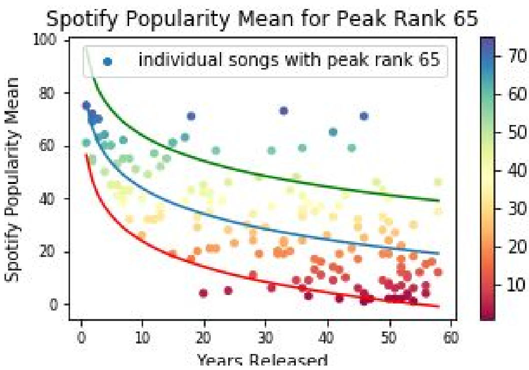
Sniff Test, 1

The most important part of data science is understanding the field being studied. Here we explore and study the outliers in our dataset, and why those songs are either performing better or worse than expected, whether that is because of its artist popularity, genre, or other features. First, we gathered the list of outlier songs by setting a threshold + and - 20 on our best fit log curve, where any point above the curve are over-performing and any point below below are under-performing.









Sniff Test, 2

After gathering all outlier points that are above our overperforming thresholds and below our underperforming thresholds for all peak ranks, we have merged the results and sorted these songs by the ratio of actual popularity/predicted popularity, because Professor Skiena suggested that this is the approach to finding overperforming and underperforming songs. After that, we sorted the outliers by dates, because songs within the same time period are likely to share many similarities.

Top 25 Overperforming Songs of All Time

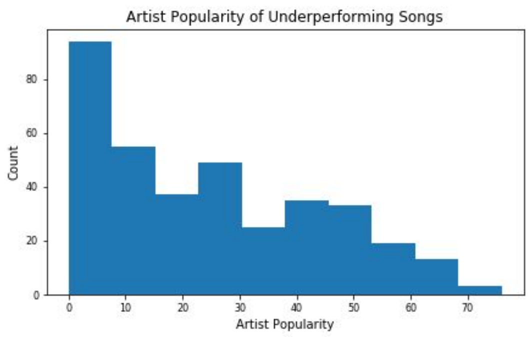
Song	Artist	Date	Peak Positior	Weeks on Ch	Spotify Popul	Spotify Genres	Artist Popula	Years Releas	Genre Score	Prediction	Difference	Ratio
The Christma	Nat King Cole	12/12/60	38	9	77	b'["christmas", "vocal jazz"]'	78	58	5	20.628502	56.371498	3.73269954
Jingle Bell Ro	Bobby Helms	12/12/60	29	19	83	b'["christmas"]'	74	58	5	25.4210244	57.5789756	3.26501398
White Christ	The Drifters	12/19/60	88	2	61	b'["pop", "folk rock", "motown", "r	64	58	4	16.2429628	44.7570372	3.75547249
At Last	Etta James	1/16/61	47	8	71	b'["classic soul", "jazz blues", "sou	68	57	4	20.8077172	50.1922828	3.41219554
Walkin Back	Helen Shapir	12/4/61	100	0	48	b'["brill building pop", "bubblegum	47	57	4	14.0527285	33.9472715	3.41570677
Strange	Patsy Cline	2/10/62	97	2	47	b'["adult standards", "christmas",	59	56	2	13.7965087	33.2034913	3.40665897
Surfin	The Beach Bo	2/17/62	75	6	67	b'["brill building pop", "folk rock",	75	56	4	16.6562189	50.3437811	4.02252159
Cant Nobody	Solomon Bur	7/20/63	66	6	58	b'["classic soul", "funk", "motown"	59	55	2	17.51937	40.48063	3.31062134
Viva Las Veg	Elvis Presley	7/4/64	92	0	56	b'["christmas", "rock-and-roll", "ro	81	54	5	16.2509969	39.7490032	3.44594246
Youre No Go	The Swinging	8/8/64	97	2	48	b'["brill building pop", "folk rock",	48	54	3	14.3016118	33.6983882	3.35626506
LOVE	Nat King Cole	9/26/64	81	4	52	b'["christmas", "vocal jazz"]'	78	54	4	15.1483794	36.8516206	3.43271043
The Race Is (	George Jones	1/23/65	96	0	53	b'["country", "country gospel", "co	63	53	2	15.7487548	37.2512452	3.36534543
I Cant Explai	The Who	3/27/65	93	2	53	b'["album rock", "art rock", "blues-	72	53	5	15.5164526	37.4835474	3.41572918
Play With Fir	The Rolling S	5/22/65	96	0	53	b'["album rock", "british invasion",	81	53	4	15.7487548	37.2512452	3.36534543
Try To Reme	The Brothers	11/6/65	91	3	49	b'["brill building pop", "folk rock",	49	53	6	14.5333609	34.4666391	3.37155324
The Weight	The Band	9/7/68	63	7	67	b'["album rock", "blues-rock", "cla	65	50	5	20.4184622	46.5815378	3.28134408
A Little Less	Elvis Presley	10/12/68	69	4	59	b'["christmas", "rock-and-roll", "ro	81	50	5	18.48937	40.51063	3.19102273
Aint Got No	Nina Simone	1/4/69	94	4	54	b'["christmas", "jazz blues", "neo s	71	49	4	16.0706934	37.9293067	3.36015372
Good Times	Led Zeppelin	3/29/69	80	4	68	b'["album rock", "classic rock", "ha	81	49	5	20.441053	47.5589471	3.32663881
Bell Bottom	Derek The D	2/27/71	91	2	52	b'["album rock", "art rock", "blues"	62	47	6	16.3576885	35.6423115	3.17893326
Can You Get	Funkadelic	9/11/71	93	3	57	b'["funk", "jazz funk", "p funk", "ps	55	47	2	17.2249925	39.7750075	3.30914512
Tiny Dancer	Elton John	3/4/72	41	7	75	b'["glam rock", "mellow gold", "pi	82	46	4.5	23.4999072	51.5000928	3.19150197
America	Simon Garfu	11/25/72	97	2	59	b'["folk", "folk rock", "folk-pop", "r	75	46	6	16.5285796	42.4714204	3.56957472
Psycho Killer	Talking Head	2/18/78	92	5	70	b'["alternative rock", "art rock", "d	71	40	7	20.3023297	49.6976703	3.44788017
Dont Stop Mi	Queen	2/17/79	86	4	77	b'["glam rock", "rock"]'	97	39	5	21.2606263	55.7393737	3.62171833

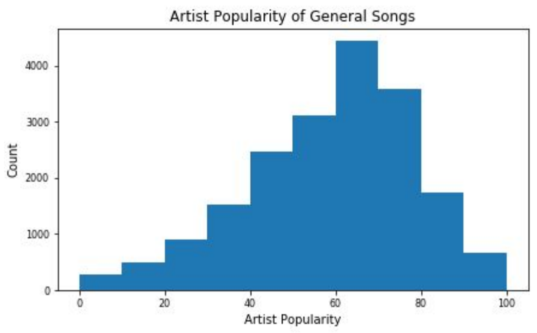
Top 25 Underperforming Songs of All Time

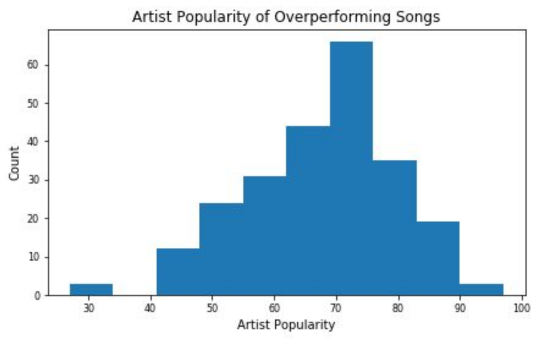
Song	Artist	Date	Peak Positior	Weeks on Ch	Spotify Popul	Spotify Genr	Artist Popula	Years Releas	Genre Score	Prediction	Difference	Ratio
Across The U	Various Artis	3/5/05	22	0	1	b'[]'	0	13	2	47.5725724	46.5725724	0.02102052
Wooly Bully	Sam The Sha	4/3/65	2	17	1	b'["rock_and	0	53	5	46.4968433	45.4968433	0.02150684
King Of Wish	Go West	5/19/90	8	24	1	b'["dance roc	55	28	8.5	46.0555762	45.0555762	0.0217129
Everybody Lo	Gary Lewis A	9/25/65	4	11	1	b'["pop_rock	4	53	4.5	42.652615	41.652615	0.02344522
Apples Peach	Jay And The '	7/15/67	6	17	1	b'["soul_mus	24	51	2	41.7878774	40.7878774	0.02393039
Country Boy '	Glen Campbe	11/8/75	11	14	1	b'["adult sta	62	43	3	39.7507881	38.7507881	0.02515673
Green Grass	Gary Lewis A	5/14/66	8	8	1	b'["country_	4	52	2	39.335321	38.335321	0.02542244
Sure Gonna I	Gary Lewis A	3/5/66	9	8	1	b'["pop_rock	4	52	4.5	37.6073364	36.6073364	0.02659056
One Has My	Barry Young	11/6/65	13	11	1	b'["pop_rock	0	53	4.5	33.8506829	32.8506829	0.0295415
Take A Look	JSon	2/24/96	74	12	1	b'["christian	35	22	8	33.0401677	32.0401677	0.03026619
I Wish That \	Ronnie and T	4/7/62	16	12	1	b'["pop_rock	28	56	4.5	33.0334999	32.0334999	0.0302723
Youre The O	The Brat Pac	2/10/90	36	12	1	b'[]'	1	28	4	32.9270905	31.9270905	0.03037013
God Bless Ar	Daniel Rodrig	12/29/01	99	2	1	b'[]'	11	17	2	32.8538287	31.8538287	0.03043785
Dont Go Nea	Rex Allen	9/15/62	17	8	1	b'["cowboy v	28	56	2	30.8886358	29.8886358	0.03237437
Kisses In The	Brandon	5/11/91	64	10	1	b'[]'	16	27	4	30.8676452	29.8676452	0.03239638
Slowly	Stacy Earl	6/20/92	52	9	1	b'[]'	2	26	4	30.7181957	29.7181957	0.032554
So Good	BoB	3/10/12	11	18	2	b'[]'	3	6	3	61.2685024	59.2685024	0.0326432
There Was A	Dorsey Burne	2/8/60	23	15	1	b'["traditiona	13	58	2	30.2888524	29.2888524	0.03301545
Misty	Lloyd Price	10/5/63	21	8	1	b'["adult sta	48	55	4	30.2473534	29.2473534	0.03306074
Deeper Deep	Freda Payne	9/12/70	24	12	1	b'["classic so	39	48	2	29.7994471	28.7994471	0.03355767
Ole Butterm	Bill Blacks Co	6/5/61	25	8	1	b'[jazz]'	22	57	2	29.5881787	28.5881787	0.03379728
Its Not A Lov	Geoffrey Wil	3/21/92	70	9	1	b'[]'	20	26	4	29.0777171	28.0777171	0.0343906
Letter Full O	Gladys Knigh	12/11/61	19	12	1	b'["motown"	55	57	2	28.6037758	27.6037758	0.03496042
Id Rather Go	Sydney Youn	8/25/90	46	11	1	b'[]'	31	28	4	28.5921182	27.5921182	0.03497467
The Nickel So	Melanie	1/22/72	35	10	1	b'["folk", "fo	47	46	4.5	28.2410356	27.2410356	0.03540947

Features, Artist Popularity

Artist Popularity: While observing the overperforming songs, we recognize many artists on the list, including Elton John, Elvis Presley, Nat King Cole, Queen. Furthermore, artist popularity for these artists are high (78, 82, 81, 97 respectively), suggesting songs with higher artist popularity are more likely to stay popular. The reason behind this is, the followers of these artists, number of awards won by them, have a high number, which is captured by Artist Popularity score.
On the other hand, we do not recognize a single artist on the underperforming list, and artist popularity for these artists are low. Therefore, songs with lower artist popularity are less likely to stay popular. As captured by the bar plots below, artist popularity for underperforming songs are very low and lower than average whereas artist popularity for overperforming songs are very high and slightly higher than average. From this insight, we conclude that artist popularity is part of predicting whether a song will be popular in contemporary times.







Modeling, Artist Popularity

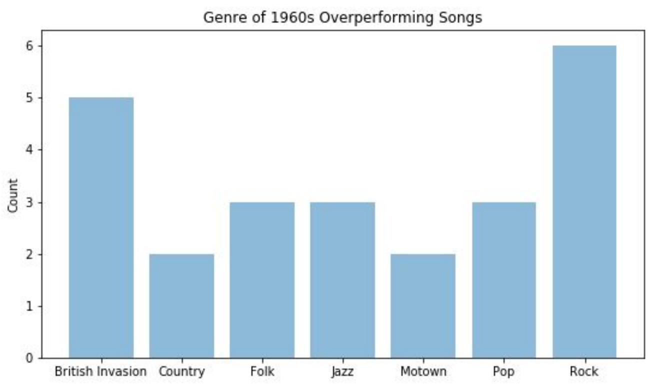
In the previous section, we observed how artist popularity affect song popularity. Because artist popularity is normalized by Spotify, the effect of artist popularity on the song popularity is linearly correlated. Therefore, our model becomes $y = a + b \log x_1 + c \log x_2 + d x_3$, where x_3 is the artist popularity.

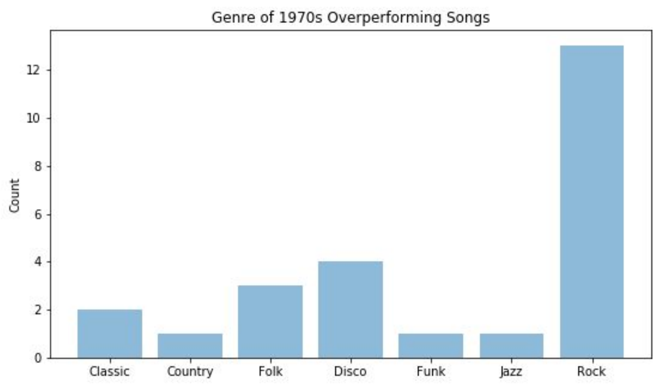
	Training Error	Testing Error
$y = a + b \log x_1 + c \log x_2$	13.71	13.66
$y = a + b \log x_1 + c \log x_2 + d x_3$	11.90	11.92

Features, Genre

Popular Genres: We scraped genres of over-performing and underperforming songs from Wikipedia to analyze the relation between the genre of a song and its popularity endurance. We split the outliers by decade and found the most popular genres in that decade. A numerical score was given to the genres to denote its genre popularity in its decade. For example in 1960s, we found out that folk was a popular genre at that time and country music was not that popular, so we gave minimum weightage (2) to country music.

Observations: While observing the over-performing songs, we noticed many songs with 'Holiday/Christmas' as genre has higher score. This can be attributed to the fact that the songs are played every year during Christmas. For example "The Christmas Song Merry Christmas to you" , "White Christmas" have endured over the years. This is also somewhat intuitive as these songs are played during festivals, year after year, therefore contributing towards its popularity.We also noticed several songs with genres and sub -genres of 'Rock' and 'Pop' music, suggesting that 'Rock' and 'Pop' genres have endure over time. While, on the other hand, for under- performing songs, we did not observe any such trend. Most of the songs are songs of not so popular genres in today's time and hence getting low scores.





Modeling, Genre

In the previous section, we observed how genre affect song popularity endurance. We have encoded each genre with an appropriate number value, these scores range from 1-10 where higher marks mean more popular genres at the time. Because these scores are normalized, our model becomes $y = a + b \log x_1 + c \log x_2 + d x_3 + e x_4$, where x_4 is the encoded genre score.

	Training Error	Testing Error
$y = a + b \log x_1 + c \log x_2 + d x_3$	11.90	11.92
$y = a + b \log x_1 + c \log x_2 + d x_3 + e x_4$	10.79	10.74

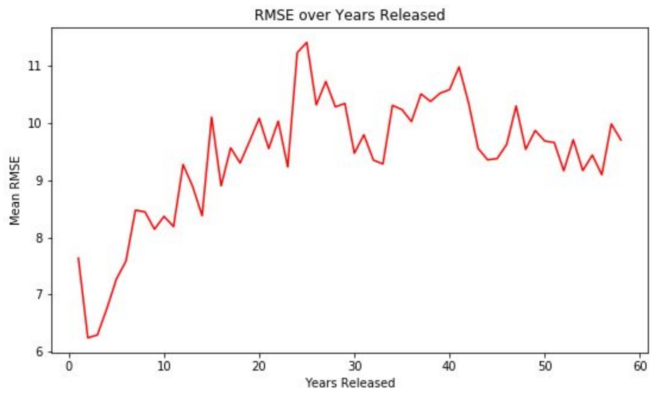
Other Features

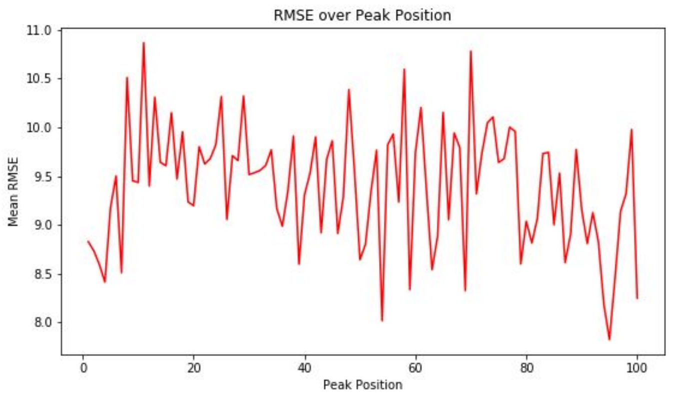
Song Features: We scraped song features like danceability, loudness,tempo,valence and other song features from spotify's API, which we thought might be an important feature for the endurance of song. This dataset was for 9000 songs only. We tried to include Loudness and Tempo in our model along with artist popularity, years ,peak rank for this amount of dataset and observed that loudness and tempo doesn't contribute significantly towards the popularity of a song.

Weeks On Chart: We included weeks on chart as an additional feature along with Artist popularity, Years, peak rank, in our model and observed that this feature doesn't contribute a lot in deciding whether a song will endure or not.

Error Analysis

In this section, we will analyze the errors and possible errors in our base model. Below is the mean error metrics for all the years released. The model seems to average out around 10.0 Spotify points. With our base model of years released, we were able to predict the songs within 16.0 Spotify points, but with our additional features, we were able to trim the error value to around 10.0. This is great, however, we can do better. One possible way is to analyze where our model is making mistakes. The error seems to be spread out in regards to peak positions, but in certain years 1978, 1994, the model has its maximum errors.





Conclusion

In this project, we tried to find the endurance of popular songs based on various features. Our assumption was that the popularity of a song would decrease as the number of years would increase. To do this, we built a non-linear regression model which helped predict the popularity of a song based on the number of years, artist popularity, genre and other song features like weeks on chart, loudness and tempo. The model performed as per our assumption, as evident from the plots above. However we noticed that there are few songs which have still maintained their popularity even today. From our model, we observed that features like Artist Popularity, Genre plays an important role in deciding the endurance of a song. For example: 'Jingle Bell Rock by Bobby Helms' has a high value for both the features (artist popularity: 74, genre score: 5). In conclusion, we would like to say that finding the endurance of a song is a difficult problem to solve as it incorporates many uncertainties like listener's change in music taste, death of an artist, which may have an impact in the endurance.