Rickord, Jake JJ
Applied Data Science
Final Report

Predicting Playlists: Logistic Regression for Spotify Recommendations

Business Problem

Since I was a young child my mother used to spin me around the living room and sing Nat King Cole to me. Every night at bedtime I would turn on my Sony boombox and fall asleep to a CD of Frank Sinatra's or Garth Brooks' greatest hits. Into my younger years and my teens, my dad and I would listen to 70's and 80's rock music and quiz each other on the first 10 seconds to see who would name the artist first. Nowadays, folk and christian rappers, and metal for working out dominate my playlists with some latino and pop blends along the way to be more in-tune with the party scene and my fiancée's favorite songs. Sufficed to say, my whole life has been dominated by listening to music constantly. Even as I sit writing this up, Creed, the Band CAMINO, and NF sit in the queue. With all this being said, I'm always hungry for new music (more evidence sits with my playlist at 531 songs). Inspired by the discovery of Spotify's API containing details surrounding the quality and characteristics of songs being callable, the aim of this project is simple: using the various data points available from my large playlist of evidentiary information, can I accurately predict new songs that I'm likely to enjoy? (Hopefully creating a methodology for others to use in a similar fashion).

Background/History

Spotify, Pandora, Apple Music, each of these and many more have various methodologies for identifying songs a user enjoys (from manual likes to more advanced methods such as time listened or nearest neighbor algorithms). Being one of the most common methods of spending one's free time (see Figure 1), a lot of research has been done and a lot of money rides on a platform's ability to not only attain music to stream, but also create an application that continues to provide users with fresh music that they might be interested in, and keep them immersed in deeper dives in genres they're familiar with, as well as auxiliary ones that might be of interest based on prior listening in some fashion. Back in 2016 for example, Spotify in particular worked on advanced analysis in such a way that '...they construct algorithms that can dissect the sound structures of songs and analyze how songs are related by scanning the billions of user-generated playlists already on the platform. And, they approximate a given user's music taste by analyzing their historical and real-time listening patterns…". With the advancement of their Spotify API platform, they've essentially passed the keys to accessing some of this information they've gathered into the hands of outside developers to give them further insight as well as to allow them to try their gambit at creating a better method for user analysis and recommendations.

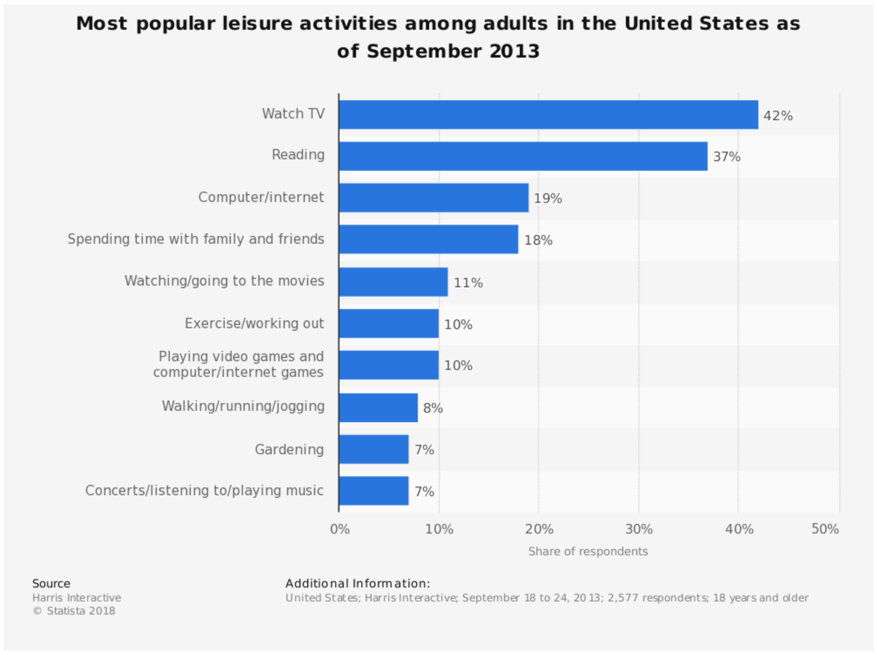

Figure 1: Most Common Leisure Activities

Data Explanation

The primary dataset we will use will be pulled from the Spotify API generated against my primary Spotify playlist entitled “Folk, Country, and Everything In-between” which is made up of over 24 hours and 500 of my favorite songs: https://open.spotify.com/playlist/3gfQ9CScIyMxMZwphYM4cZ

Additionally, since these were all listed as “liked” songs, we also needed the negative category. To do so, we selected 50 songs we had a strong preference against, and combined these into a playlist found here: https://open.spotify.com/playlist/6kT6nn9Vd1e52XRG7OS2IE

One element to consider during our later analysis is that both playlists contain songs from many different genres, i.e. no list is made up solely of country, rap, pop, etc but rather a mix of all in the affirmative or negative fashion. The features for each track in the playlists include a number of unique details including: danceability, energy, key, loudness, mode, speechiness, acousticness, instrumentalness, liveness, valence, and tempo.

Our first step in accessing the details listed above was establishing a Spotify API account. Once we were able to set this up, we established our API connection via the Spotify package in Python, and we were ready to pull the data down from Spotify! The first step in that process was to grab all the tracks from the respective playlists, which were saved into a list element. Once compiled, we went through each list, grabbed each track within them, and pulled down the interior song features discussed earlier such as energy, danceability, key, etc. For each we attached either a value of 1 if the song came from our primary playlist, or a 0 if the song came from our “Disliked Songs” playlist. Then they were combined together into a large dataset. This being set, we dropped out a few reference columns to keep them out of the way for our later modelings, and then we were off to the races!

Methods

The first step we took in our modeling process was to break our data into test and train splits, which we chose a 75/25 division for. Once complete, we tried out leveraging a simple Logistic Regression model to test out the accuracy of our models in predicting our test set. As we evaluated that outcomes of listening to a disliked song (False Positive) and not listening to a song we’d like (False Negative) were equally unwanted, we determined that the f1 score of our model would be the most ideal metric to evaluate against. Our initial modeling scenario produced the following results:

```
              precision    recall  f1-score   support

           0       0.00      0.00      0.00        13
           1       0.92      1.00      0.96       142

    accuracy                           0.92       155
   macro avg       0.46      0.50      0.48       155
weighted avg       0.84      0.92      0.88       155
```

Figure 2. Results of Initial Logistic Regression Model

While our F-1 score looks great there when you examine the liked songs in a vacuum, if you take into consideration the fact that our model predicted 0 disliked songs, the performance wasn’t nearly as good. As such, we moved forward into a different modeling technique: SVC. Implementation of a simple SVC classifier model produced the following results:

```
              precision    recall  f1-score   support

           0       0.08      0.62      0.14        13
           1       0.91      0.37      0.52       142

    accuracy                           0.39       155
   macro avg       0.50      0.49      0.33       155
weighted avg       0.84      0.39      0.49       155
```

Figure 3: Results of Simple SVC Classifier Model

We were still not pleased with the result even though this model actually predicted some disliked songs. This is likely due to the fact that the 500+ song playlist largely outweighs our 50 song disliked playlist. As such we tried two different methods of addressing this, namely: adjusting the weights on our datasets to accommodate for the large imbalance in playlist sizes, or simply taking a smaller sample of the liked songs playlist to work with. We tried both out, however we believe it would make more sense to have a solution that worked in the former context such that knowing more songs that an individual likes will often be the case in most scenarios. The outcome of the smaller samplings (50 each of liked and disliked tracks) came out as follows:

Linear Regression:

```
              precision    recall  f1-score   support

           0       0.47      0.67      0.55        12
           1       0.50      0.31      0.38        13

    accuracy                           0.48        25
   macro avg       0.49      0.49      0.47        25
weighted avg       0.49      0.48      0.46        25
```

Figure 4: Resampled Linear Regression Outcome

SVC Classifier:

```
              precision    recall  f1-score   support

           0       0.33      0.25      0.29        12
           1       0.44      0.54      0.48        13

    accuracy                           0.40        25
   macro avg       0.39      0.39      0.38        25
weighted avg       0.39      0.40      0.39        25
```

Figure 5: Resampled SVC Outcome

Lastly, the outcomes from adjusting the weight of our inputs was generated by iterating all samplings of weight distribution (i.e. iteration 1 with disliked weighted at 0% and liked at 100%, iteration 2 with disliked weighted at 5% and liked at 95%, etc). The best results occurred at a distribution of disliked songs weighted at 85% and liked songs at a weight of 15%. These results are as follows:

```
              precision    recall  f1-score   support

           0       0.08      0.15      0.11        13
           1       0.92      0.84      0.88       142

    accuracy                           0.78       155
   macro avg       0.50      0.50      0.49       155
weighted avg       0.85      0.78      0.81       155
```

Figure 6: SVC Model Optimized for F-1 Score

Analysis

Out of all of our outcomes, no F-1 Score was particularly attractive as it lay based on macro avg between FPs and FNs. Of those identified, the best possible algorithm seemed to be the 0.49 macro-avg F-1 Score found by our iterative manipulation of the weightings using our massive liked songs playlist. It still has a much higher tendency to label a majority of songs as liked or likely to like, but out of those shown above, the average likelihood of it being correct is the highest.

Conclusion

From our modelings, we were able to extrapolate song details from the Spotify API about our favorite and least favorite songs, and from these details, build out a modeling algorithm with ideal weightings to maximize F-1 scores, and compare this against our logistic regression model F-1 scores. Using the best of these outcomes, which in our case ended up being a 15/85 split of liked to disliked songs, we can infer songs that we might be interested in listening to and, to a lesser extent, which songs we might dislike.

Assumptions

One assumption we made during our modeling and evaluation period was that the best method of evaluation was based on comparing F-1 scores. For other users it may be more important to ensure they don’t hear songs they’ll likely dislike as opposed to missing out on songs they *will* like. Many companies seem to operate on the opposite mindset with some balance in place, however prioritizing that users hear songs they may like instead. Additionally, one huge leap is that the model that works best for our listening applies equally to others’ listening patterns.

Limitations

Some limitations involved within our dataset is that the song interests are based on what the primary participant disliked. If particularly the disliked elements were able to be extrapolated based on others playlists as well, we could replicate the modeling experiment to see how well this applies to others listening habits and the accuracy of such predictions.

Challenges

Early challenges for this project were getting into and accessing the Spotify API. Once these hurdles were cleared, the only remaining logistical challenges were identifying modeling issues seen in which consistently all algorithms were returning all songs identified as liked, with 0 dislikes for all modeling implementations. Outside this, the project was relatively challenge-free.

Future Uses / Additional Applications

Future uses of this project could be expanded with replication of the modeling algorithms against other users' music interests in the same fashion. Additional classification modeling methods could be of interest as well to identify their accuracies. Lastly, attempting to build this modeling without attention to FP's will allow for algorithms such as nearest neighbor implementations.

Recommendations

Moving into fixes for this model in order to bring it to a successful implementation, more data as always is at the top of the list. This could again come from other users experimenting the same way, or in the shape of additional liked/disliked songs.

Implementation Plan

In order to implement this model according to the recommendations, the steps needed to be taken include:

- Test against other users
- Try out additional classification modeling algorithms
- Proceed trying the most accurate out on new tracks
- Evaluate accuracy via blind test

Ethical Assessment

Ethics here aren't of major consequence, however privacy in user data usage as well as ensuring samplings are inclusive of many different regions and ethnic backgrounds could be points of contention depending on implementation uses.

Questions:

1. What steps were involved in setting up your Spotify API access session?
2. What was your reasoning behind using F-1 Score as your primary method of evaluating your models?
3. What other classification models would be applicable to this situation?
4. If you could go back and start this analysis over, what would you do differently?
5. Spotify actually doesn't have dislikes built into their application, why is that?

6. Based on that, is there a way your model can be modified to strictly work off of liked songs?
7. Besides more data, are there any other ways you can think of to improve the predictive accuracy of your model?
8. Can this modeling be carried over to other implementations such as movies or TV shows?
9. What kind of sampling techniques did you use against your dataset?
10. How would you implement this modeling from a corporate standpoint?

Rickord, Jake JJ
Applied Data Science
Final Report

Appendix

- Chang, E. (2022, January 4). Building a Song Recommendation System with Spotify - Towards Data Science. Medium.
- https://towardsdatascience.com/part-iii-building-a-song-recommendation-system-with-spotify-cf76b52705e7 Hu, C. (2021, December 2). Why Spotify's music recommendations always seem so spot on. Popular Science. https://www.popsci.com/technology/spotify-audio-recommendation-research/
- Marius, H. (2022, January 4). Uncovering How the Spotify Algorithm Works - Towards Data Science. Medium. https://towardsdatascience.com/uncovering-how-the-spotify-algorithm-works-4d3c021ebc0
- Statista. (2015, February 9). Most popular leisure activities among adults in the U.S. 2013. https://www.statista.com/statistics/382623/most-popular-leisure-activities-among-adults-us/