FINAL REPORT:

Spotify Music Recommendation Engine

The Problem

Ever wonder how Youtube knows exactly what kind of funny cat videos you will enjoy? How it is able to recommend a recipe for your favorite type of food, or a tutorial for a videogame you are playing, or even a song you enjoy? Whether it be Netflix, Youtube, Apple Music, Facebook, or Amazon Prime; recommendations play a massive part in nearly every industry in every corner of the internet. From facebook friends you may know, to the perfect pair of socks on amazon, all the way to the newest romantic comedy on netflix; recommendation engines are responsible for introducing people to so many new things. In a time where there is more content than ever to be discovered and growing daily, it can be overwhelming and exhausting to scroll through potentially 1000s of artists, or hand bags, or movies to find the perfect one for you. This is when it can be helpful to have someone (or something) that knows you so well that it can suggest something for you that can save you time, energy, and resources. And that is exactly where recommendation engines come in.

Recommendation engines are a type of machine learning algorithm that takes all of your data, so that it can attempt to predict the perfect thing specific to you. And they are everywhere. I am sure you have had to agree to something about “cookies” on a website and may not have known what it meant. Everytime you go to a website now you must agree to let the website track your information, so it can learn your habits. If you have Google Chrome. it is actually easy to see exactly what your google profile knows about you. When looking at mine, google knows my age, my gender, the foods and music I like, the video games I play. Google can then use this information to deliver the perfect ads to me that they know will interest me, all to try to get me to buy something. The value of this type of algorithm and data is almost unquantifiable. What can you do with the power of a model that knows you better than yourself? So I thought to myself, is there a way I can cut out the middleman and build my own recommendation engine?

The Solution

For my recommendation engine, my goal was to learn more about my own musical data and preferences, and attempt to predict some new songs I would like that I can add to my daily music playlists, using a classification based recommendation system. For these models, I used my own music data, collected from my own personal Spotify profile, using the Spotify API, Spotpiy.

Data

With the Spotify API, I was able to collect data on: my top artists, my top songs, my playlist data, my saved tracks data, and even recommendation tracks by Spotify's own algorithm. Some examples of qualities that I was able to collect for my song data includes: genre, danceability, energy, key, loudness, mode, speechiness, acousticness, instrumentalness, liveness, valence, tempo, and more. With this data, I knew it would be a good foundation to begin exploring my data and eventually to train my models.

This was one of the most fun projects to work on, because the data was personal to me. It was fun and interesting to break it all down and learn more about my own listening preferences. When people ask me who my favorite musical artist or band is, I usually tell people my favorite is Eminem. But when breaking the datadown, it tells a different story and he was not even in my top 5!

After exploring my data more and learning more interesting trends about myself (EDA can be found in the project's github folder), the first thing I had to do was figure out a rating system and apply it to the main dataset of songs that I would be using to train my models. I experimented with both top songs and top artist data in a rating system, and found that applying a rating using my top artists data would give me both a more complete rating system and more accurate model prediction. To apply a rating I gave a rating of 1 to every song in the dataset that was by an artist in my top artist, and a rating of 0 to every song that was not by an artist in my top artist dataset. This rating system I was able to rate 399 songs out of a total of 1133 songs in the dataset.

After rating my whole playlist, I moved right into training my models. To train my models, I used all of the song quality variables in my dataset:

'danceability', 'energy', 'key', 'loudness', 'mode', 'speechiness', 'acousticness', 'instrumentalness', 'liveness', 'valence', 'tempo', 'time_signature', and 'genres'.

I was not interested in using certain variables such as explicit song, song length, or popularity. I did not want a song's length or if it used profanity to contribute to whether I would like it. And I definitely did not want to use popularity, because I am building this model to hopefully discover not popular songs that I still enjoy. Some diamonds in the rough. After using these variables for my predictor variables, I wanted to try to predict the “rating” of a song when given a new dataset. After designating my X and y variables for my train_test_split, I then one hot encoded all of the genre columns into numerical columns to be more easily read by the models.

After splitting up my data and training my variables, I then fit a Random Forest Classifier with the data to get an idea of feature importance. For my data, the top 10 features of importance came out to be:

1. rap 0.07338067434111464
2. speechiness 0.050407626497393415
3. energy 0.04533777337736288
4. danceability 0.043416393481863165
5. instrumentalness 0.042697105051097604
6. loudness 0.04092880252721668
7. acousticness 0.03507823743659393
8. detroit hip hop 0.034302485156577356
9. tempo 0.03183338031885002
10. valence 0.03137924787590353

Looking at this at first glance, I thought it was fairly accurate. Because the data was rated based on top artists, and the majority of my top artists and genre of music listened to was rap, it makes perfect sense to see both rap and Detroit hip hop in the top 10 of importance. When looking at the top 50 features of importance, the overwhelming majority of features 11-50 was different types of rap and hip hop genres. From here I moved on to fit my models with the training data.

Modeling

I fit 3 different Classification Models with my training data: Logistic regression, kNN, and a Random Forest Classifier. The accuracy and classification report of each model can be seen below:

Logistic Regression (After scaling data with a min_max scaler):

The accuracy score is: 0.6651982378854625

	precision	recall	f1-score	support
0	1.00	0.67	0.80	227
1	0.00	0.00	0.00	0
accuracy			0.67	227
macro avg	0.50	0.33	0.40	227
weighted avg	1.00	0.67	0.80	227

---

kNN (After determining optimal n_neighbors of 2):

The accuracy score is: 0.6563876651982379

	precision	recall	f1-score	support
0	0.85	0.70	0.77	185
1	0.26	0.48	0.34	42
accuracy			0.66	227
macro avg	0.56	0.59	0.55	227
weighted avg	0.74	0.66	0.69	227

Random Forest Classifier (after determining optimal max_depth of 20, and optimal min_samples_leaf of 1):

The accuracy score is: 0.947136563876652

	precision	recall	f1-score	support
0	0.99	0.94	0.96	159
1	0.87	0.97	0.92	68
accuracy			0.95	227
macro avg	0.93	0.95	0.94	227
weighted avg	0.95	0.95	0.95	227

After fitting 3 models it was clear that the Random Forest Classifier was the top performing model, and the model I would use to attempt to predict which songs I would like from a validation dataset. I used Spotify's recommended tracks for me as the validation set to see how the model performs. I created a variable called “prob_ratings”, and used the trained Random Forest Classifier model to predict how likely I would like the song.

Results

After running my trained Random Forest Classifier model on the validation dataset, there was a total of 116 songs in the dataset that had a probability rating of over 0.75 of me enjoying the song! Success! I was really excited to see this and explore the dataset and listen to some recommended songs.

Some Takeaways:

-The first thing I notice is that the song recommendations are very similar to what I listen to on a regular basis. As I originally predicted, because I used my top artist data to rate the songs, the model recommends a lot of songs by these same artists. However, there are several different artists who are not in my top artists who I still recognize and enjoy, as well as some artists I have never heard of.

-My top artists dominate my song recommendations. Perhaps if I used specific song data as well as top artist data I would get a bit more diversity in the recommendations, but in doing that it may also reduce the probability ratings of the songs suggested.