Sean McLean

Capstone Project – Final Submission (Music Recommendation Systems)

Execution Summary

Key Takeaways

The main focus was building a recommendation system that provided 10 songs for a user based on the odds that they will be listening to those songs. Looking at the final model after implementing all the algorithms and coding there were many aspects that I believe helped improve it and make it a much better recommendation system. Eliminating a large portion of the users and songs that had very few interactions was helpful in that it made it easier on running models so it could make them more relevant interactions. If there is little user or song history then it won't have much of an impact on building a model, so condensing and filtering the values was very useful and is easier to locate interactions with higher values.

The final model after capping values showed how very few repeat listens there were to the songs in the dataset. This could be an indication that when people are using a music streaming provider that they are not seeking out songs again and that they are likely wanting the provider to recommend songs that they will listen to. Users want the streaming service to find the songs they would likely prefer to listen to for the convenience factor instead of having to seek out the music themselves. This makes building a recommendation system more vital in that having a stronger link to user and song interactions will make it more likely that the user will continue using the music platform for its streaming purposes.

Another trend I thought was interesting after running a code that looked at songs per year and constructing a bar plot was how the majority of the songs were from the last decade or so of

the dataset. It really shows that even though it is a streaming provider with a large database of music that what is new is still the most common in terms of what users want to listen to. It can also identify what type of users are using the platform and how these demographics can play a role in building a well-functioning recommendation system.

The final model has been constructed in an efficient way where techniques like filtering and implementing different algorithms have made it stronger. From this juncture I think the first step would be to conduct a trial run to see how the recommendation system is operating and whether it is highly effective or if it needs finer tuning. It would be interesting to get a glimpse into how the metrics change and the user-song interactions will be improved from the initial prototype built. I would curious to know what the F1 scores will look like after the trial run and if there is any movement with having any increases where you can feel comfortable knowing that relevant items are being recommended and recommended items are relevant.

I would choose a trial of one to three months to see how the model is working and then do further testing to find any areas that can be cleaned up. Some more research then where possibly some new algorithms could help the model would be one option if there are some areas of concern in that the model needs substantial upgrades. Based off the metrics from the algorithms used in the model, if they are other techniques that can be incorporated to boost the values then I do think that would be the next logical step in this process.

Problem and Solution Summary

The task at hand was from a provided music dataset to build a recommendation system that would find the 10 best song recommendations. Some of the methods used to tackle and solve this problem is looking at past user-song interactions, locating latent features from users, similarities in users and items, and also how song information could be an asset. One of the

biggest challenges is from the beginning where you have a very large dataset that can affect the algorithms used as well as fine tuning the hyperparameters that would improve the performance of the model. The initial step in having a large database to work with requires finding the most important data that will make it easier to solve this problem, which is where filtering the dataset is the most useful at condensing the files.

Using the provided algorithms in the notebook, finding the best approach to the final model is the next essential ingredient. Each algorithm came up with different metrics that were improved from fine tuning the hyperparameters, but none of them changed enough to say that one algorithm really stood out from the rest and be used toward the final model. The user-user similarity based collaborative filtering method had better metrics after the fine tuning than the other algorithms while the item-item similarity based collaborative filtering method showed very little movement in its metrics from being optimized.

Before deciding on which algorithm to choose, it is probably important to recognize what type of streaming provider you want to build. Some possibilities include wanting to recommend more popular songs so the dataset could be reduced even more after filtering, or having more variety with the recommendation system where songs with smaller play counts are included so there isn't as much filtering required. A balance of both might be the best route so you aren't leaving out certain types of users who might only listen to certain artists or music from a particular time period.

Like the user-user similarity based collaborative filtering method that had solid metrics to work with, the final results also showed that the cluster-based recommendation system had higher predicted ratings than other algorithms used. The fine tuning of the algorithm though showed the least amount of improvement. The model based collaborative filtering method also

did not improve very much from the hyperparameters being fine-tuned but did also have predicted ratings for play counts on songs that were higher than some other algorithms that were used. This technique would be a good opportunity to look at past user and item behaviors to see if that could affect the model, although in this case it didn't seem to have much impact.

While each algorithm displayed different metrics and have their own unique qualities that could benefit the final model for predicted interactions, the content-based recommendation system is one method where you can look at provided song information to find patterns. This could serve as a music recommendation system search engine that locates key words or values in the data and be able to narrow down possible user-song interactions that could be recommended. Looking at the query code in the notebook example with the band Foo Fighters and their song 'Learn to Fly', there was play counts that were higher than one which might suggest that if the recommendation system is effective than users may go back to listen to that band or song again.

While the final solution to the model isn't perfect since predictions are never always perfect, I think the process of using different algorithms after filtering the dataset and then fine tuning the hyperparameters is a good starting point for building a recommendation system. I think it improves on the initial values that came from the dataset and uses the interactions and play counts with higher values for building the final model. If those interactions indicate that the song or artist is popular, then that will be crucial to having a good recommendation system and will continue to recommend those songs and artists.

The final model also alleviates useless information that is not needed and will make it weaker if it is kept in the filtering process. Some examples are getting rid of columns that are either redundant or won't impact the final model and also relabeling certain features like song id and user id so it is more user friendly and easier to understand. Merging the song identifications

from the two files also made the filtering process less complicated and took out the duplicates. An important element of this is also making sure that the users that are using the dataset are the ones being kept in the model building instead of users that haven't listened to many songs. Overall, these processes help locate what is working in the initial dataset and presents a model using different algorithms for finding the 10 best songs.

Recommendations For Implementation

If I had to choose one algorithm in terms of metrics and ratings then I would choose the user-user similarity based collaborative filtering method. The technique shows the past ratings of an item based off how similar users have rated it which I believe is still one of the stronger indications that past behaviors from similar users can influence future outcomes. It also showed some of the better improvements from fine tuning of the hyperparameters which shows that finding neighbors for this algorithm is impactful. Using the clustering-based recommendation system is the other possible algorithm I would lean toward in that it helps group together users who have similar traits with songs and I think that could help build a reliable model because the ratings will be close to each other.

The implementation of the cluster-based recommendation system and the content-based recommendation system are two of the most valuable algorithms because they look at other features of the dataset that could bring extra value to the model. This could be an interesting way to include themes into the final model where users could be in search of a particular type of song or genre and these two algorithms would play a role in finding those songs from past interactions. An example of this could be listening to songs pertaining to the holiday season in December, so song information about events like Christmas would help users find these songs and then compute an effective recommendation system on the streaming platform.

If you are going to present this model to stakeholders, then making sure you know your demographic for the music dataset is a good first step toward making a final recommendation model. The bar chart that shows what year the songs come from and how many interactions are attached to those songs in the dataset gives a pretty good idea of what era of music that the users are listening to and want to listen to. That could also provide some background information on what demographic the listeners are. If the music dataset was condensed to just the most recent decade, then the model would still be pretty strong because the majority of the songs are from that time period. So, knowing the demographic can be a part of the equation in how you want to approach the recommendation system.

Another key aspect toward demonstrating to the stakeholders the keys to the final model is showing how the algorithmic changes via fine tuning can improve its metrics. This I think can build some trust then to the people that have interest in the model in that there are techniques like cross validation and the grid search that can make the algorithm more reliable and increase the predicted and corrected ratings. And I also believe that conveying that the algorithms can be adjusted in its parameters to see if that would beneficial to any of the outputs. This kind of flexibility is a major asset if the outputs of the model are still subpar and therefore are a work in progress toward building a better system.

The benefits toward a good recommendation platform are that it can gain popularity if it builds a reputation of providing strong recommendations for its users. If the algorithms create a model where only a small total of songs are played from past behaviors and interactions, then users will lose interest from hearing the same songs over and over again. Having a large enough dataset that provides variety but also plays well known songs to its users will increase the subscription base and will also lead to more profit if the dataset becomes available to subscribe to for a fee.

Having a large dataset with a lot of music will also add more exposure to less known artists and their albums and songs. If the model is built with the right algorithms installed then the diversity of the recommendations will be more evident and will bring up play counts for songs. This will obviously depend on what kind of model you want to create and if past behaviors of users and items indicates that only songs that are well known to the public should be recommended. These kinds of streaming services do offer immense opportunities for artists to have their songs heard and can see more exposure if the recommendation system is implemented.

One of the challenges when making a recommendation system is worrying if there could be some model bias in terms of who is being recommended. While the data and the filtering and algorithms put in may not initially have any issues with certain biases toward artists and songs, the outputs could have some inadvertent biases. This could lead to maybe only certain genres of music being played or only music from a certain era being recommended. I'm unsure of how to tackle this challenge if the final model had this issue, but revising the algorithms and restructuring the filtering of the datasets could be a starting point in seeing if that could alter any of the outputs. This could also place a burden on how many users will subscribe to a streaming provider if are they recommending songs or artists that are not relevant to the user.

Looking at the future analysis there are some aspects of the current dataset that I think could benefit the model long term. The rankings of the songs and interactions by year shows that the most for one year are the songs with a number 0 attached to it, meaning that the algorithms could potentially be missing a song that does not have a year. One solution to this problem would be removing the year column and seeing if that would have any effect on the final model. If the content-based recommendation system algorithm remains strong or even improves without the year column then maybe this is an option worth pursuing. If the user cares about the era of music

they are listening to then removing all songs with the number 0 as the year might be a way to have better recommendations toward songs that cater to that user.

Another aspect of future analysis with regards to the music scene is following trends and patterns to see how user behaviors evolve and how that will affect user-song interactions. Over time maybe certain artists or genres will not be as popular as they used to be and newer ones will replace them as to what users want to listen to. Those kinds of concerns will mean that the recommendation system will need to adjust to those changes and that some items in the dataset could become obsolete. They would need to be filtered out and new items would be filtered in due to the increasing popularity. That is one element of any recommendation system in that it is completely based off human behavior and therefore it always needs to be worked on so it provides the best recommendation system to its users.