Tom Bohbot
Machine Learning Recommender System
May 24, 2021

<u>Methods of building a recommender system:</u>

As I increasingly researched into which type of recommender system I wanted to build to for my simple model, it seemed like I had infinite possibilities. The different types of recommender systems had an overarching split of collaborative versus content-based, and within those subsets were many different ways to implement.

Since we were required to create a collaborative recommender system I came to the decision of whether I would like to use a model or memory based approach. At first I wanted to use a knn algorithm which would fall under the memory based approach as knn is a memory based algorithm. However, I realized that an algorithm like this may not be able to scale as it does depend to on memory, and I wanted to use a model-based approach.

<u>Why did I choose SoftMax:</u>

Naturally, I considered a logistic regression SoftMax approach as we have been using this type of algorithm throughout the semester. The other contender was matrix factorization. However, I decided to go ultimately go with SoftMax DNN because it is better for cold start as it can efficiently handle new queries, and it is scalable since the item embeddings are static. After the train/test split I was able to achieve a RMSE ranging between 0.5 and 0.6. I only used a train/test split per the update in response to my question on the slack channel.

<u>What I did for Hybrid Model:</u>

For the hybrid model I combined my SoftMax DNN with a TF/IDF. The reason I chose to use TF/IDF over other models for the content based approach is that TF/IDF has the ability to place certain weights on words to prioritize them as more important than others. I thought this could be very important for our recommender model as not every word should be worth the same when recommending movies. I used the TF/IDF on the genres and then stacked this model on top of the SoftMax DNN I used in the simple model. I was also inspired to use the TF/IDF because we heavily covered it in our data cleaning course, and it was interesting to see the overlap into machine learning.

<u>Progress on the Report:</u>

Initially I wanted to do the collaborative model using a user-user and item-item matrix. I was able to make this model which I call Simple Model 1 in my code. However, I was unsure of how to accurately test it using RMSE. I did show that it can have output that seems functional though. Following my struggles to test it the required way I made an alternative model, called

Simple Model 2. This model was able to test RMSE as I built a custom function which I passed into my keras model. The RMSE score ranges between 0.5 and 0.6 with a learning rate below one. These scores seem low which may mean that my model is quite efficient. This is how I built the simple model.

Before advancing onto the hybrid model, I would like to state that I used different datasets that were approved for the project.

For the hybrid model I used a stacking approach where I took the simple model's output and also built a different model that uses TF/IDF as a content-based recommender system. Once I had both outputs, I stacked them on top of each other and attempted to choose which ever was the better choice between the models. I found this model very interesting as I used different approved data sets to attempt to better the model.

Overall, this project was very enjoyable, and it influenced me to read about recommender systems and learn about DNN's, content vs. collaborative models, SVD, etc. Although I did not use it in the project, as it is not permitted, this project also taught me a lot about the surprise package python has for recommender systems which I found very interesting.