Recommender Systems CUNY SPS

Tulasi

6/18/2017

DATA643 Discussion 2

by Tulasi Ramarao

Please complete the research discussion assignment in a Jupyter or R Markdown notebook. You should post the GitHub link to your research in a new discussion thread.

For this discussion item, please watch the following talk and summarize what you found to be the most important or interesting points. The first half will cover some of the mathematical techniques covered in this unit's reading and the second half some of the data management challenges in an industrial-scale recommendation system.

https://www.youtube.com/watch?v=3LBgiFch4_g

How to find good recommendations.

Christopher Johnson from Spotify discusses how his company does music recommendations and the company's experimentation with Apache Spark. He explains that there are many ways to sort through a catalogue of 40 M songs to recommend music to users that they like.

He simplified the complicted task of finding a good music recommendation to four categories:

1. Manually curate attributes - not scalable, but good for small catalog - (Ex:Songza)
2. Manually tag attributes by bringing in music experts to tag all the catalogues and obtain 200+ attributes. Then sort thru them to perform machine learning. This sytem does not scale well and takes up a lot of time from musicians.
3. Audio context and Text Analysis - The company echonest bought by spotify uses this method. Browing twitter articles to parse what artists are talking and extract information like userA talking about artistA and userB talking about artistA and then building a relationship between these artists based on this information.
4. Collaborative filtering is the fourth category which is historically used by Spotify. This method looks at what users are listening and finds a relationship and builds recommendations based on that observation.

The best explanation of an idea is when it can be narrowed down to one line or a picture. Christopher did a wonderful job in reducing the whole idea of collaborative filtering into a picture, where two men are exchanging their music albums. The man with the silver helmet has music tracks P,Q,R and S and the man with the golden helmet has music tracks Q,R,S and T. The former is suggesting the later to try track P and later is suggesting the former to try T. They both have similar interests and they can be seen suggesting the track the other is missing. This really represents the core idea of collaborative filtering, which is a method that finds a relationship like these two men and builds recommendations based on other observation.

Christopher also talks about the implicit matrix factorization that Spotify is using, where rather than ratings, they implicitly infer based on what the users are listening to. Implicit way assigns binary ratings(1 for streamed 0 never streamed). Adds weights so that if the user listened to Michael Jackson 10 times and Daft Punk once, then MJ gets more weight. To solve, ALS is one method that goes alternative back and forth and solves the least squares regressions. That is, it fixes the song vectors and becomes a weighted regression. It then fixes the song use and solves the user vectors. Once the user vector is solved, it fixes the user vector and solves for the song vector. So it repeats this procedure until the convergence.
Two interesting things to note in the video:

In the ALS method, it was interesting that removing $X^T X$ means that now to solve for the optimal song vector for the given song, only the ratings that the user actually streamed that song is needed.(other ones are filtered out in $X^T X$ - no zeroes)

The other impressive thing was the IO bottleneck that Spotify faced with Hadoop (reading & writing from disk). Spotify got around it using Apache Spark. They load the rating matrix in memory, instead of performing IO to disk and then they cache the rating matrix, while performing the iterations.