FIT5212 Data analysis for semi-structured data - S1 2021

Assignment 2

NAME: PRAJYOT NAGRALE
STUDENT ID: (31132324)

MONASH UNIVERSITY

Task 1: Recommender System Challenge :

Amid the final few decades, with the rise of YouTube, Amazon, Netflix and numerous other such web administrations, recommender frameworks have taken increasingly put in our lives. From e-commerce (propose to buyer's articles that seem intrigued them) to online notice (recommend to clients the correct substance, coordinating their inclinations), recommender frameworks are nowadays unavoidable in our everyday online journeys.

In an awfully common way, recommender frameworks are calculations pointed at recommending pertinent things to clients (things being motion pictures to observe, content to peruse, items to purchase or anything else depending on businesses).
Express feedbacks are appraisals positioned on a concrete rating scale (such as from 1 to 5 stars). Users' inclinations are expressly demonstrated when they rate a thing that they like higher than a thing they don't like. Certain feedbacks are inferred within the client activities. The basic suspicion is that "if a client clicked/viewed/spent time on a thing regularly, it is a sign of its inclination for that item". Certain criticism comes in numerous shapes, browsing, clicking, writing, though express criticism exclusively comes from a few sorts of appraisals scale.

I'm using 4 different Recommendation algorithm to get the predicted items for each user:

1. Alternating Least Squared
2. Matrix Factorization
3. Matrix Factorization with bias
4. Neural Network

1. Alternating Least Squared:

ALS is an iterative optimization prepare where we for each cycle attempt to reach closer and closer to a factorized representation of our unique data. We have our unique framework R of estimate u x i with our clients, things and a few sort of input information. We at that point need to discover a way to turn that into one lattice with clients and covered up highlights of estimate u x f and one with things and covered up highlights of estimate f x i. In U and V we have weights for how each user/item relates to each highlight. What we do is we calculate U and V so that their item approximates R as closely as conceivable: R ≈ U x V.

I'm using the ALS model to get the recommend items for each user as per their rating in the training data. I'm using this recommendation from training data and sorting them for the highest recommend score for each user in descending format. Once I've done that, I'm checking the recommend items for each user from the training dataset and getting the top 15 recommendation items for each user. Once I get that I've used creating a .csv file to upload in the Kaggle.

When I used the default parameters for the ALS model it give me accuracy of 0.17754. After below tuning of the hyper parameters:

1. factors=32, regularization=0.1, iterations=50: Kaggle Score = 0.19
2. factors=64, regularization=0.1, iterations=50: Kaggle Score = 0.17
3. factors=32, regularization=0.01, iterations=50: Kaggle Score = 0.18
4. factors=32, regularization=1, iterations=50: Kaggle Score = 0.19
5. factors=32, regularization=1, iterations=100: Kaggle Score = 0.195
6. factors=32, regularization=10, iterations=100: Kaggle Score = 0.19336
7. factors=32, regularization=1000, iterations=100: Kaggle Score = 0.07
8. factors=10, regularization=1, iterations=1000: Kaggle Score = 0.19296f
9. factors=10, regularization=0.5, iterations=500: Kaggle Score = **0.22181**

I got a score of 0.22181 in Kaggle by using the above parameters. I also changed the value of alpha to 35, to get a better fit for the testing data.

2. Matrix Factorization:

Matrix factorization is the breaking down of one system into a thing of diverse systems. It’s incredibly well inspected in number-crunching, and it’s exceedingly profitable. There are various unmistakable ways to calculate systems, but specific regard weakening is particularly profitable for making recommendations. It reduces the dimension of the recommender system to increase the calculation power and decrease the size.

The Matrix Factorization may be a neural network recommender demonstrate which gives us the leading things for each client as per their appraisals sometime recently. I utilized this demonstration to test our testing dataset into Kaggle. The yield which I got for this demonstration is 0.5835. This can be lower at that point the ALS show which we saw over. But the advantage of this demonstration is that it is much quicker at that point the ALS show.

3. Matrix Factorization with bias

Inclination terms depict the effect of one estimation on the surrender. For outline, inside the Netflix challenge case, the inclination of a motion picture would depict how well this movement picture is assessed compared to the typical, in general movement pictures. This depends because it was on the motion picture (as a to start with figure) and does not take into thought the interaction between a client and the motion picture. Additionally, a user's predisposition compares to that user's affinity to convey better or more terrible examinations than the typical.

Matrix Factorization with predisposition is comparable to the network factorization demonstrate which we utilized prior. As it contrasted in typically that it calculates the predisposition term for each client to provide more preferred recommendation for each client. It is utilized to assist the exception within the evaluations. After uploading this demonstration in Kaggle I was able to see that it gave away better score at that point ordinary lattice factorization which is **0.6459**. So we can say that this demonstration is way better than the MF show without inclination.

4. Neural Network:

This is often a Profound inclining neural organize show which is diverse the lattice factorization. It has to insert which are diverse in estimate at that point the framework duplication. The Neural network model has given us the least Kaggle score in which is 0.05803.

Conclusion:

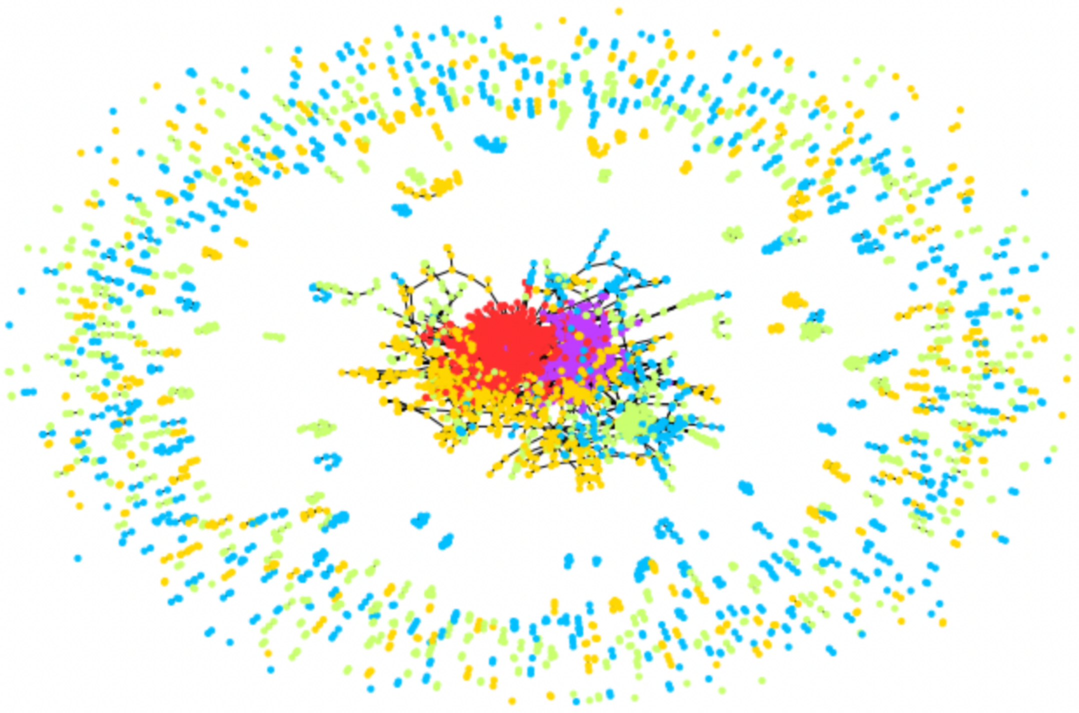
From all the over models, we were able to see that models have different inserting and calculation ways to discover the suggestion for clients. Till presently, I got the most excellent score from the ALS show which is 0.22181. So I prefer to utilize that as my best show for this dataset. This happened since ALS has a more complex calculation to get the expectation of each client many ways better at that point other proposal calculations.

2. Task 2: Node Clustering in Graphs

A Chart may be a collection of centres (vertices) at the side recognized sets of centres (called edges, joins, etc). In Network, centre points can be any hash able address e.g., a substance string, a picture, an XML address, another Chart, a customized centre challenge, etc.

One of the controls of charts remains the nonattendance of vector highlights. A bit like in NLP, we go up against organized information. But or maybe like in NLP, able to learn an embedding of the chart! There are many levels of embeddings in a chart: Embedding chart components (centre points, edges, features…) (Node2Vec). Embeddings sub-parts of a chart or an entirety chart (Graph2Vec) After learning embeddings, it can be utilized as highlights for several errands: classification, recommender systems.

K-Means Clustering can be essential be that as it may compelling calculation in data science There are plenty of real-world applications of K-Means Clustering (numerous of which we'll cover here) This comprehensive coordinate will display you to the world of clustering and K-Means Clustering at the side utilization in Python on a real-world dataset.



We will moreover see that the NMI score for the genuine and unsurprising label are exceptionally less that's 0.2876. This implies that the clusters are not appropriately recognized by the K means show.

Still, the NMI score is 0.2067 which is less at that point than the overscore we got. From this I was able to get it that in case we decreased the hubs for the cluster, it is difficult for the demonstrate to discover the clusters within the data. This suggests it'll diminish the NMI score for the demonstration.

At long last, for Node2Vec inserting, I attempted all the demonstrate and edges that are given within the dataset. Which doesn't have names show in it as well. This has expanded the number of nodes and edges within the chart but, too expanded the time to compute the demonstration. As its learning each inserting that doesn't have a name connected to it, the demonstrate can learn more from the dataset.

which is why we are getting the most noteworthy NMI score among all the over models. For this, we are getting a 0.3369 NMI score. Which is less in itself, but is distant much way better than the over show. This chart has hubs that do not have names connected to it. We cannot make a clustering chart for this demonstration.

Laplacian Matrix:

Laplacian network is another Chart inserting strategy. I utilized to compare another implanting strategy to discover the NMI score for another inserting at that point Node2Vec inserting. But from the comes about, able to see that Node2Vec inserting has done great work to assist k-means clustering to urge the good NMI score which was 0.3369. While the Laplacian network inserting doesn't offer assistance much to k-means to cluster the hubs. This inserting must deliver us a less NMI score which is 0.0852. This score is distant much less the Node2vec inserting. From this, we will clearly say that the Node2vec implanting is distant much way better than Laplacian network implanting for this dataset.

Conclusion:

1. Node2Vec implanting is way better at that point in the Laplacian network.
2. Keeping inserting of all the hubs and edges which doesn't have names will allow us way better clustering.