Intelligent Web-based Information Systems

ASSIGNMENT 2 - RECOMMENDER SYSTEMS

Shatil Masud

101040995

COMP 4601

04/11/2021

Dave McKenney

Overview

This document will provide an analysis on the efficiencies of different recommender system algorithms. The types of two types of recommendation systems that will be investigated here are User-Based Systems and Item-Based Systems. The dataset which the algorithms will be tested on, is the assignment2-ratings-data.txt file which is a file containing historical movie review data. Examining this information, this report will attempt to answer four questions which are: 1) Is a user-based or item-based recommendation more accurate for this data? 2) Is top-K (i.e., selecting the K most similar users/items) or threshold-based (i.e., selecting all users/items with similarity above some threshold X) more accurate for this data? 3) Which parameter values produce the most accurate results for this data (e.g., is 2 neighbours best? 10? 100? a threshold value of 0? 0.5?, etc.) 4) Based on the previous responses, what algorithm/parameter combination would we use for a real-time-online movie recommendation system?

Item-Based or User-Based

For extremely large datasets, it is not computationally fast to perform a recalculation of all the data and derive predicted scores every time new data is entered. The main piece of data that needs to be extracted from these changing datasets is the user average rating or the item average rating (depending on the approach taken). On the other hand, it is important to derive predicted scores based on information extracted from the current dataset. To address this problem, item-based systems are preferred. Because items generally have a large number of reviews (larger in comparison to the number of reviewed items by users), the removal or addition of one or a few ratings of that item will not significantly change the average rating of the item. On the flip side, the user average score will indeed change a lot if a user-based approach is chosen. Because the dataset is so large, the item-based recommendation is more efficient and accurate.

The accuracy is due again to the fact that the data is being inspected in terms of item units rather than user units. There are many things to consider with users. First of all, most do not rate items, which makes it difficult to compare users based on what we assume are movies they liked. The unknown ratings can be movies they really liked or really disliked. The problem is that it is unknown what to do with these items. In the case of a user-based recommendation system, this data is simply ignored which easily leads to a huge lack of information being considered when rating items.

Moving forward, we will observe further reasons for why the item-based recommendation system is the preferred method.

Top-K or Threshold

This question required more experimentation. We will begin by investigating the item-based system and then follow up with the same analysis on a user-based system.

The next question in place is whether the item similarities should be filtered using a certain threshold coefficient value or if a top neighbour based implementation should be used instead.

Threshold (Item-Based):

This section will examine this issue as well as find the optimal K and threshold values. A series of tests were done using both approaches and the observed data was plotted in the diagrams below. There are 4840 items in the dataset. Using the 'Leave-one-out' approach, the Mean-Absolute-Error was calculated for each test to determine the strength of the predicted results.

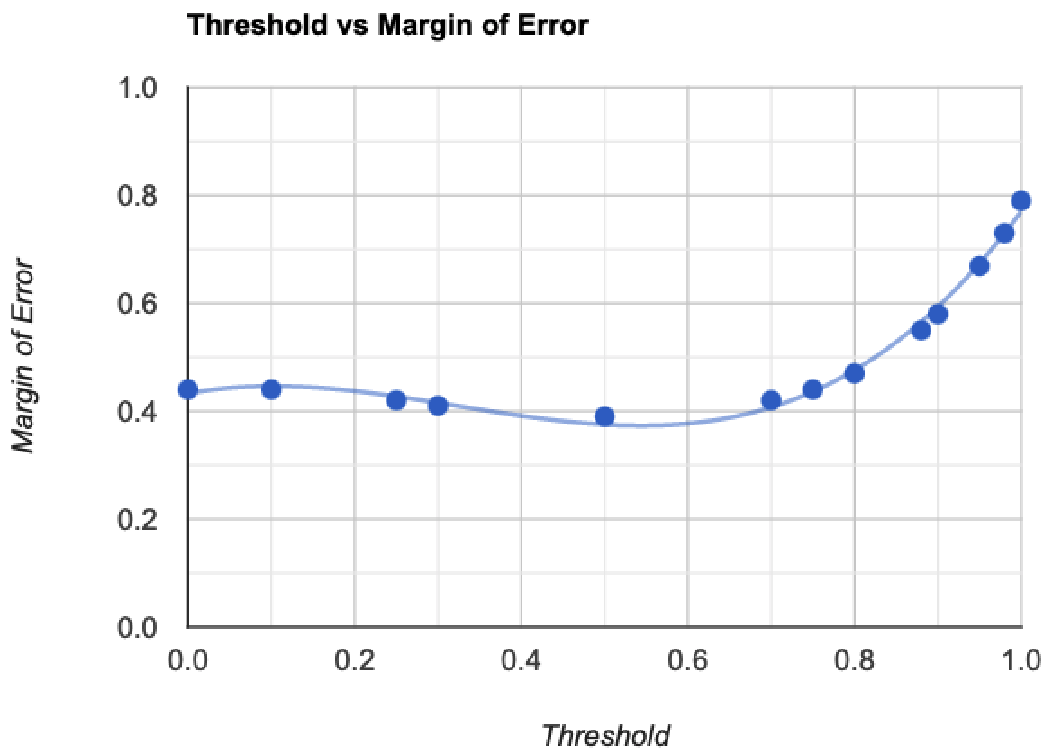


Figure 1: Item-Based Thresholds

Multiple different threshold values were tried from the range 0.00 to 1.00. From the graph above, it appears that threshold values from the range 0.0 to 0.7 yield a mean absolute error value of what looks to be an average of 0.43. After the 0.7 threshold, the mean absolute error rises up very quickly and eventually reaches a value of 0.79. Although the advantage of a high threshold is to only utilize values of high similarity, the associated disadvantage is that there is less data to work with when calculating the predicted values. This results in heavily biased results. For this reason, it is important to find the sweet spot which is a fair threshold with a reasonable amount of data (similar items). The diagram above appears to favour a higher threshold from values 0.0 to 0.5. After this point, the higher thresholds begin to be counter productive.

The lowest mean-absolute error is 0.39 which comes from the threshold of 0.5. The mean absolute error does however seem to be more or less the same at a higher threshold of 0.6. At this higher value, the program can also run faster by having to analyze less, yet still a reasonable amount of data. For this reason, if choosing the threshold approach, a value of 0.6 seems to be a fair choice to make.

Top K Neighbours (Item-Based):

For the neighbour-based approach, a number of tests were also conducted. As stated previously, there are 4840 items which means there are 4839 neighbours. Similar to the threshold approach, a 'sweet spot'/optimal range needs to be found. A low K value would keep the absolute most related items however, a K value which is too low would eliminate a lot of data which would be helpful to our calculations.

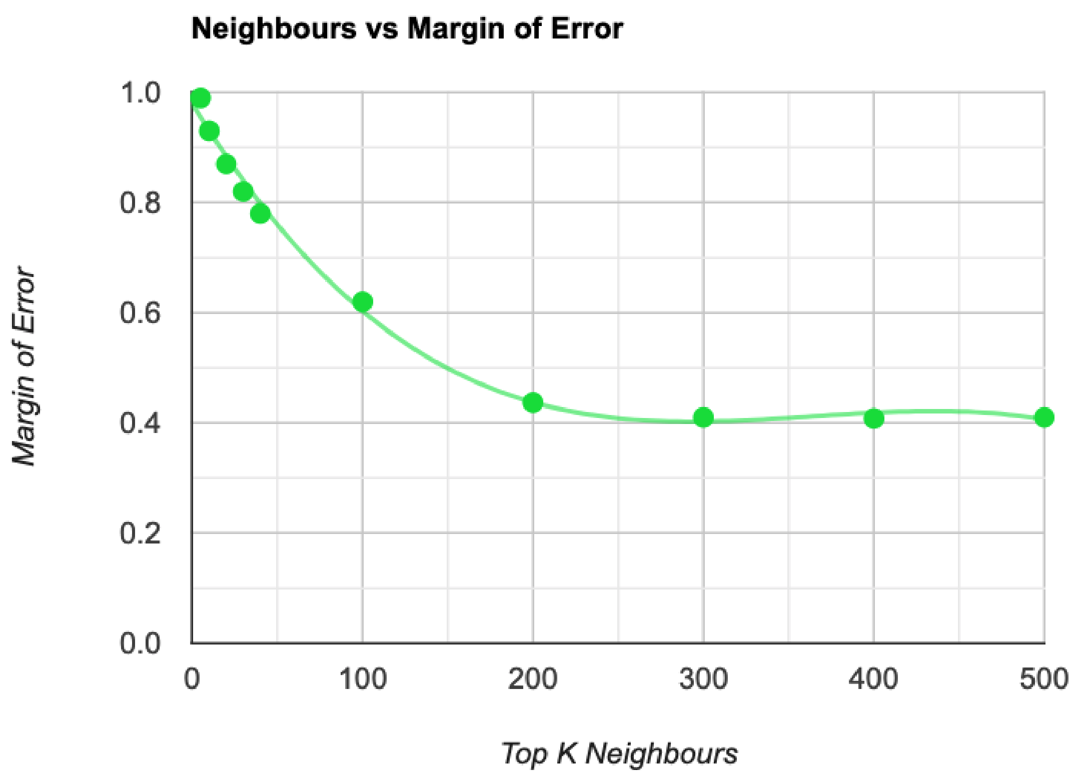


Figure 2: Item-Based Top Neighbours

From the graph above, it appears that a K value from the range 200 and above produces around the same mean absolute error of approximately 0.41. For this reason, a value of 200 seems to be a fair choice as any value above will unnecessarily utilize more time despite producing similar predictions.

Threshold (User-Based):

The next set of results to be examined will be the user-based approach. Overall, the predictions were much worse as expected because a user-based approach does not fit this kind of data. The following diagram illustrates the findings of different thresholds in relation to its resulting mean absolute error. The mean absolute error in comparison to the item-based system, was over triple the value.

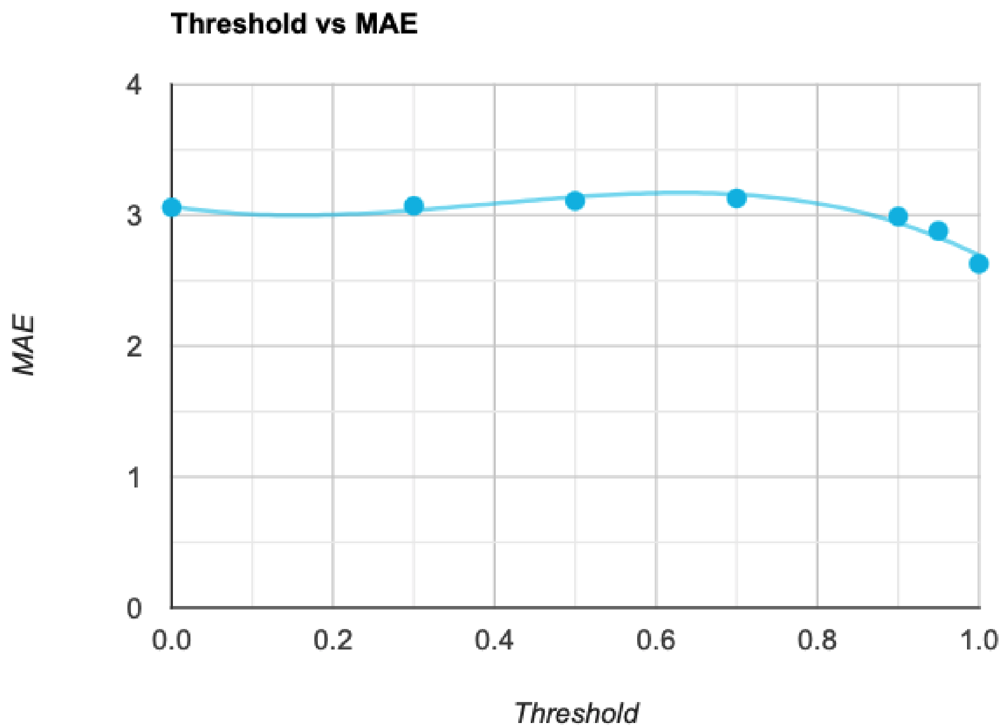
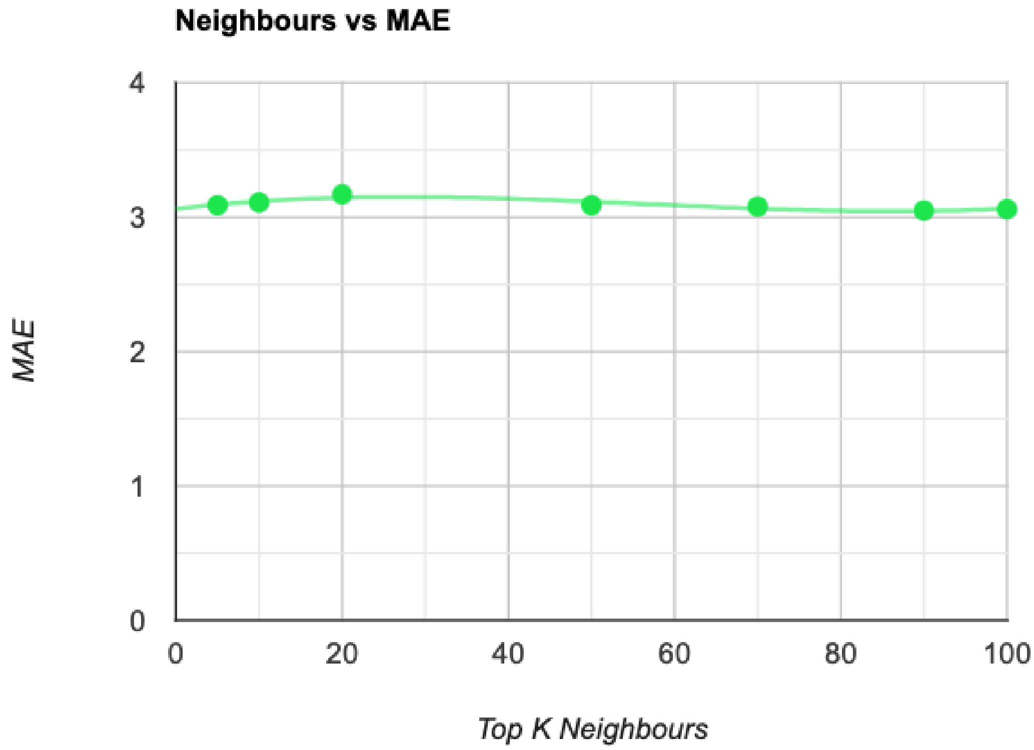


Figure 3: User-Based Threshold

The general trend of this graph remains at a high value of around 3 until a threshold of 0.7 after which the mean error begins to decrease. Because most users rate a very small amount of items, it is difficult to relate users given such sparse data. Unlike the item-based system where items could indeed have a strong correlation, it is not correct to assume that user similarities would produce accurate results. For this reason, the far greater value of the mean absolute error in comparison to the item-based recommendation system comes as no surprise.

Top K Neighbours (User-Based):

By a very similar token, the top K neighbours approach is not so great either. Observing the Figure 4 from the diagram below, the trend appears to be quite consistent in this graph too. From 5 to 100 neighbours, the mean absolute error appears to stay very consistent again at a value of 3. There does not seem to be much of an optimal value here. Between the threshold and the top k approach, it appears that using the threshold method with a value of 0.8 would be fair. However, it is important to keep in mind that the data is very consistent throughout both graphs.



To summarize the user-based recommendation findings, the data confirms the early suspicions of this approach not being a good choice. Between choosing top neighbours or threshold, a threshold of 0.7 might be the option to go for if a user-based system is chosen.

The Best Method

Judging by the experiments above, the item-based recommendation system with a threshold value of 0.5 seems to be the optimized method. The benefits to this method are as follows:

- Item-based similarities for a large dataset like this will produce a list of similarities that are much stronger than user similarities. This in turn will result in more accurate data to be passed in as parameters to the subsequent functions.
- The threshold of 0.5 appears to be a fair threshold taking into account a fair amount of data while also considering strong users. This threshold produces the mean absolute error of 0.39.
- The ability to generate all similarities once and save them to a file. This would allow for a fast and efficient program which would not need a whole lot of processing.
    - Any filtering by a threshold can be done by reading the array of similar items and performing a quick sorting operation to filter out the values below the threshold.