Abstract

In order to improve both prediction accuracy and the time required to recommend items to a user based on collaborative filtering, a simple algorithm is proposed. The algorithm is different than traditional algorithms only in some simple steps.

1 Introduction

A common problem with typical filtering based recommendation systems they usually require a lot of users similar to a user for prediction (such users are called **neighbors**). This heavily affects performance since the complexity gets incredibly high as well. Herlocker et al. [3] uses a memory based collaborative filtering technique that uses similarity between users to predict recommendations using KNN algorithm. This algorithm suffers from some problems such as when too few items are common between two users, predicting accurately becomes harder. Bell, Koren, and Volinsky [1] uses a model based learning which is inherently very expensive. Shen, Wei, and Yang [6] introduces a way to improve calculating similarity more accurately but they do not offer all that much improvement for the cost of learning similarity, so in this paper such improvement has been ignored and more traditional approaches have been used. Jamali and Ester [4], Pavlov and Pennock [5] uses hybrid collaborative and content filtering or memory and model based algorithm for recommendation. While these works have their merits, most of them are complicated and offer little value to be used in practice. In this paper, we have showed that it may be possible to gain significantly better results by simply modifying traditional approaches a little bit using some principles which are mathematically sound. We will show that using our proposed modifications, we get a flat line in accuracy metric graphs much earlier.

2 Algorithm

Let us first describe the assumption problem that we want to solve. Let $\mathcal{M}$ be a set of movies and $\mathcal{U}$ be a set of users. We are given a set of triplets $(u, m, r) \in \mathcal{S}$ such that $u \in \mathcal{U}, m \in \mathcal{U}$ and $1 \leq r \leq 5$ which denotes the rating r of movie m given by user u. We are also given a set of pairs $(u, m) \in \mathcal{T}$ such that $u \in \mathcal{U}, m \in \mathcal{M}$ and we want to predict the rating r given by user m to the movie m. Let r_{um} be the rating of movie m given by user u and C_{uv} be the set of items that both u and v rated. We use the *Pearson Correlation Coefficient* (see Freedman, Pisani, and Purves [2]) as the measure of correlation between user u and v. However, we will use a normalized version of it for practical consideration as follows:

$$S_{uv} = \frac{\sum_{m\in C_{uv}}(r_{um} - \bar{r_u})(r_{vm} - \bar{r_v})}{\sqrt{\sum_{m\in C_{uv}}(r_{um} - \bar{r_u})^2}\sqrt{\sum_{m\in C_{uv}}(r_{vm} - \bar{r_v})^2}}$$

$$S_{uv} \leftarrow \frac{S_{uv} + 1}{2}$$

This normalization is done using the fact that Pearson correlation coefficient p_{uv} is always in the range $[-1, 1]$. We call two users u and v similar if $S_{uv} \geq s$ for some positive real number s such that $0 \leq s \leq 1$. Typically, we want s in the range $[.5, 1]$. For this paper, we will consider $s \in \{.7, .8, .9\}$. For a movie m, let $\mathcal{U}_m$ be the set of users who rated m and M_u be the set of movies rated by user u. For a set or tuple A, let $\bar{A}$ denote the average of the numbers in A. For a tuple of weights $\mathbf{w} = (w_1, \dots, w_n)$ such that $0 \leq w_i \leq 1$ and $\sum_{i=1}^{n} w_i = 1$ and a tuple of positive real numbers $\mathbf{a} = (a_1, \dots, a_n)$, the *weighted harmonic mean* of $\mathbf{a}$ is defined as

$$\mathfrak{H}(\mathbf{a}, \mathbf{w}) = \frac{\sum_{i=1}^{n} w_i}{\sum_{i=1}^{n} \frac{w_i}{a_i}}$$

Usually, *weighted arithmetic mean* is used to predict the ratings in a recommendation system. But in this paper, we have investigated the results using harmonic mean. Next, we describe the rating prediction algorithm for a pair (u, m).

Algorithm 1: Algorithm to predict rating

```
Input: Test data in the format $(u, m)$, Threshold $t$, similarity $s$, $T$ to
       take first $T$ neighbors for a user $u$
Output: A single integer in the range $[1, 5]$ denoting the predicted
        rating
Data: Train data in the format $(u, m, r)$
$W \leftarrow []$
$X \leftarrow []$
$tot \leftarrow 0$
$R \leftarrow U_m$
res = 0
for $v \in R$ do
    if $|C_{uv}| < t$ then
        continue
    end
    if $S_{uv} < s$ then
        continue
    end
    if $S_{uv} \geq s$ then
        $W \leftarrow [W, s]$
        $X \leftarrow [X, r_{vm}]$
        $tot \leftarrow tot + 1$
        if $tot > T$ then
            break
        end
    end
end
if $tot > 0$ then
    res = $\mathfrak{H}(X, W)$
end
else
    if $|M_u| > 0$ then
        res = $\bar{M_u}$
    end
    else
        res = $\bar{R_m}$
    end
end
res = res+.5
res = floor(res)
return res
```

2.1 Algorithm Principles

Our algorithm is based on the following principles.

First principle *Two users u and v are more relevant for each other if it is ensured that $|C_{uv}| \geq t$ for some large enough positive integer t.* It is obvious how this principle helps us establish better correlation between users since two users u and v can have very high correlation with very low number of common items between them.

Second principle *If first principle is established, then it is possible to get an accurate estimation of predicted rating with a lower number of neighbors instead of using a very large number of neighbors.* This helps us predict a rating a lot more efficiently with better accuracy. It is also possible to get a mathematical sense of why this works in practice. Let m and n be positive integers such that $m > n$. Let u be a user and $\mathcal{N}_m = \{v : |C_{uv}| \geq t\}$ and $\mathcal{N}_n = \{u : |C_{uv}| \geq t\}$ be two sets of neighbors of a user u such that $|\mathcal{C}_m| = m$ and $|\mathcal{N}_n| = n$. Using the condition $|C_{uv}| \geq t$, it can be assumed that if $s < t$ for a positive integer s, then

$$P(\sigma^2(A) \leq \sigma^2(B)) \tag{1}$$

should be very high where $A = \{r_{vm} : |C_{uv}| \geq t\}$, $B = \{r_{vm} : |C_{uv}| \geq s\}$ and $P(x)$ denotes the probability of the random variable x. Since higher value of the threshold t ensures better similarity between two users, we can say in a non-rigorous way that the *Pigeonhole principle* ensures (1) is high enough in practice more often than not.

Third principle *If two sets of ratings A and B have similar variance and consist ratings given by similar users of u only, then $P(|\bar{A} - \bar{B}| < \epsilon)$ is very high for some positive real number ϵ which is considerably smaller than* 1. We can show that a special case holds very often in practice. Since A and B has ratings from similar users only, assume that both A and B consist of 4 and 5 only (our data set rating range is $[1, 5]$). If $|A| = m$ and $|B| = n$ and a is the number of 4 in A whereas b is the number of 4 in B, then

$$\begin{aligned}
\bar{A} &= \frac{4a + 5(m - a)}{m} \\
&= \frac{5m - a}{m} \\
&= 5 - \frac{a}{m} \\
\bar{B} &= \frac{4b + 5(n - b)}{n} \\
&= \frac{5n - b}{n} \\
&= 5 - \frac{b}{n}
\end{aligned}$$

As we can see, it does not matter if $m \gg n$ or $n \gg m$ since the difference $|\bar{A} - \bar{B}|$ or the ratio $\frac{\bar{A}}{\bar{B}}$ only depends on the ratio $\dfrac{a}{m}$ and $\dfrac{b}{n}$. So as long as these ratios are similar, $\bar{A}$ and $\bar{B}$ are similar as well.

3 Results

We show the results of our algorithm on the Movielens 1 million data set and backup our claims with empirical evidence below. The quality of performance is measured using the $F1$ score. The $F1$ score is the harmonic mean of precision and recall.

$$P = \frac{TP}{TP + FP}$$
$$R = \frac{TP}{TP + FN}$$
$$F1 = \frac{2PR}{R + P}$$

We have chosen $F1$ as an accuracy metric for couple of reasons.

1. It is well known that the harmonic mean of n positive real numbers $a_1, \ldots, a_n$ is less than $\min\{a_1, \ldots, a_n\}$ for $n > 1$. So an $F1$ value of x indicates that the model has at least x precision and x recall.

2. $F1$ score can penalize bad precision or bad recall scenarios properly where arithmetic or geometric means may fail. For example, if we consider a recommendation where we do not predict anything, technically, we do not have any errors. So the precision would be 100 percent whereas recall would be 0 percent. An arithmetic average would give us an accuracy of 50 percent which is clearly wrong. $F1$ score penalizes all such scenarios accordingly. For this particular example, we would still have an $F1$ score of 0 percent.

For this paper, we call a test data point *true positive* if a certain movie m is to be recommended to user u based on the rating. If the rating is over k for some fixed k, then the movie is to be recommended, otherwise it is to be discarded. We have showed results for both $k \geq 3$ and $k \geq 4$ in this paper.

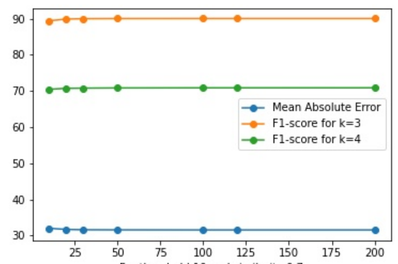
We have also used *mean absolute percentage error* as a metric in place of the more traditional *mean absolute error.* Mean absolute error is calculated using the formula:

$$MAE = \frac{1}{n} \sum_{i=1}^{n} |y_i - y_i'|$$

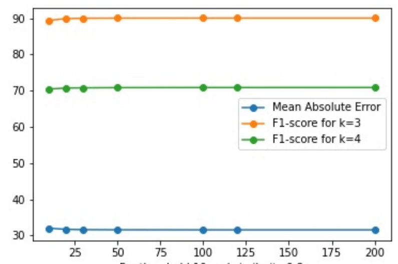
whereas mean absolute percentage error is calculated using the formula

$$MAPE = \frac{1}{n} \sum_{i=1}^{n} \frac{|y_i - y_i'}{y_i}$$

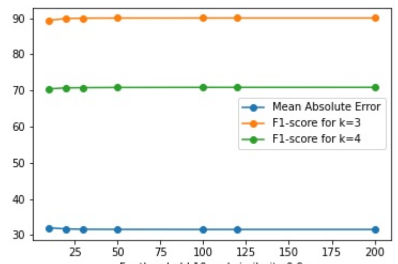
where y_i is the actual value for the data point at i and y_i' is the predicted value of this data point. It is easy to see that mean absolute error puts equal weights on all ratings whereas mean absolute percentage errors mitigates that problem to some extent. So we felt it important to also show the results in terms of mean absolute percentage errors.



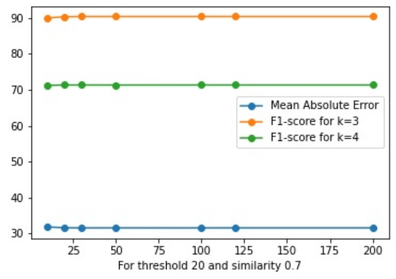
For threshold 10 and similarity 0.7

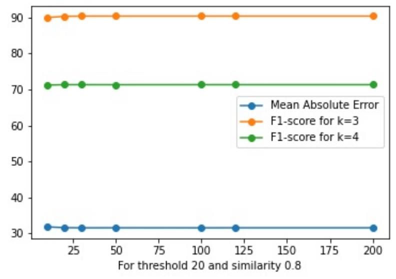


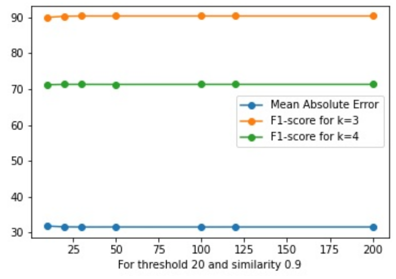
For threshold 10 and similarity 0.8

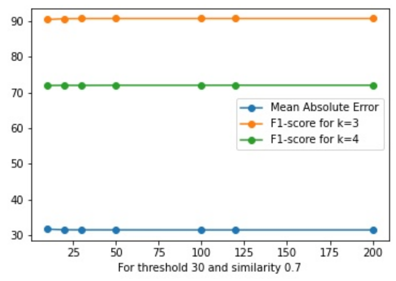


For threshold 10 and similarity 0.9

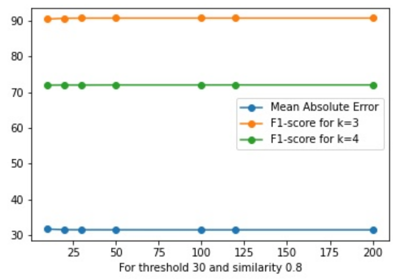

Mean Absolute Error
F1-score for k=3
F1-score for k=4
For threshold 20 and similarity 0.7


Mean Absolute Error
F1-score for k=3
F1-score for k=4
For threshold 20 and similarity 0.8

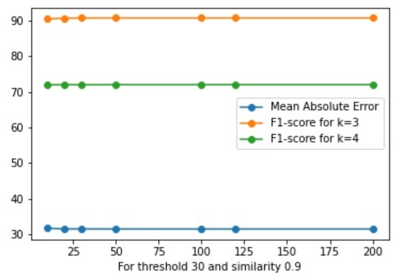

Mean Absolute Error
F1-score for k=3
F1-score for k=4
For threshold 20 and similarity 0.9



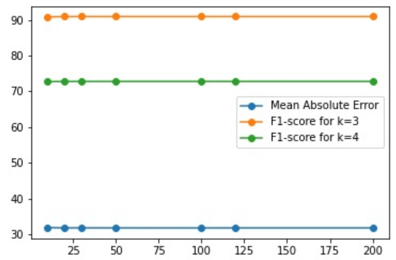
Mean Absolute Error
F1-score for k=3
F1-score for k=4
For threshold 30 and similarity 0.7



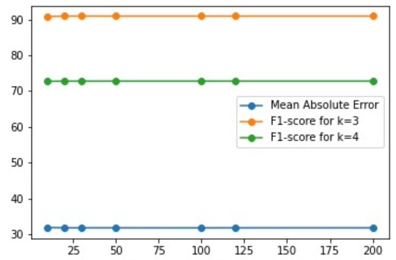
Mean Absolute Error
F1-score for k=3
F1-score for k=4
For threshold 30 and similarity 0.8



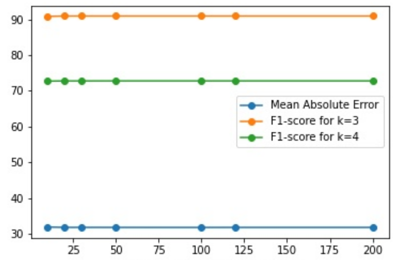
Mean Absolute Error
F1-score for k=3
F1-score for k=4
For threshold 30 and similarity 0.9



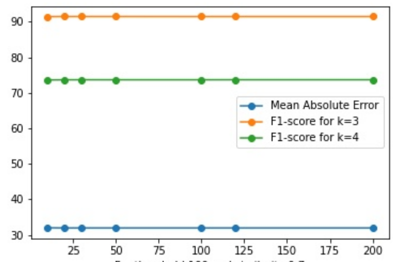
For threshold 50 and similarity 0.7



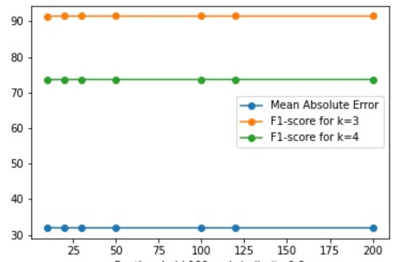
For threshold 50 and similarity 0.8



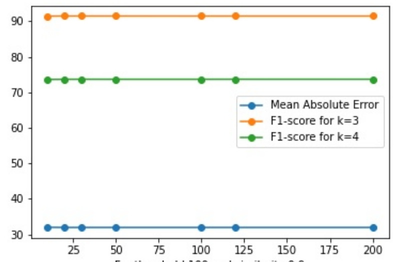
For threshold 50 and similarity 0.9



For threshold 100 and similarity 0.7



For threshold 100 and similarity 0.8



For threshold 100 and similarity 0.9

4 Conclusion

As we can see in the result section, all the graphs tend to give us a flat line very early on. usually, we do not get much improvement in accuracy after 30 neighbors and we do not have to go beyond .7 similarity threshold for considering neighbors.

References

[1] Robert Bell, Yehuda Koren, and Chris Volinsky. "Modeling relationships at multiple scales to improve accuracy of large recommender systems". In: *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '07* (2007). DOI: 10.1145/1281192.1281206.

[2] David Freedman, Robert Pisani, and Roger Purves. "Statistics (international student edition)". In: *Pisani, R. Purves, 4th edn. WW Norton & Company, New York* (2007).

[3] Jonathan L. Herlocker et al. "An algorithmic framework for performing collaborative filtering". In: *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval - SIGIR '99* (1999). DOI: 10.1145/312624.312682.

[4] Mohsen Jamali and Martin Ester. "TrustWalker". In: *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '09* (2009). DOI: 10.1145/1557019.1557067.

[5] Dmitry Y. Pavlov and David M. Pennock. "A maximum entropy approach to collaborative filtering in dynamic, sparse, highdimensional domain". In: *In Advances in Neural Information Processing Systems, Cambridge, MIT Press, Mass, USA* (2002), pp. 1441–1448.

[6] Jian Shen, Yan Wei, and Yupu Yang. "Collaborative filtering recommendation algorithm based on two stages of similarity learning and its optimization". In: *IFAC Proceedings Volumes* 46.13 (2013), pp. 335–340. DOI: 10.3182/20130708-3-cn-2036.00068.