**Personalization mini project**

**Documentation**

**Business objective:**

Our recommendation system comes up with the list of candidate movies and recommends the highly-rated ones to a specific user to achieve personalization goal.

**Methods:**

Dataset

We use MovieLens 100K dataset for building the recommendation system. It contains 100,000 ratings from 1000 users on 1700 movies.

Surprise package

https://surprise.readthedocs.io/en/stable/index.html

Collaborative filtering models

a) Baseline approaches

The algorithms in this section try to minimize the following regularized squared error:

|  |
| --- |
| https://lh3.googleusercontent.com/bSkEuT1AexBFYFD-iOQ61uqV2en3u6hG_VX_VdShh_Hwksn7S78djBO2g9mCGD1vRbpkJcdsDtEJBn6eZQyeLT6o3SwLmi0zU0W2aItC-WNW0jbVIZygWoEegLPSZyIuUZAH9fBw |

: the observed deviation of user *u*

: the observed deviation of item *i*

Baselines can be estimated in the following two ways:

Alternating Least Squares (ALS)

The hyperparameters to be tuned are:

reg\_i: the regularization parameters for items

reg\_u: the regularization parameters for users

n\_epochs: the number of iteration of the ALS procedure

Stochastic Gradient Descent (SGD)

The hyperparameters to be tuned are:

reg: the regularization parameter of the cost function that is optimized

learning\_rate: the learning rate of SGD

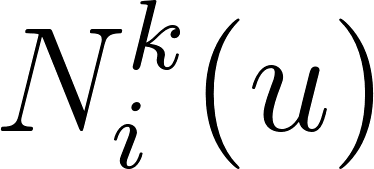
n\_epochs: the number of iteration of the SGD procedure

b)    Neighborhood-based

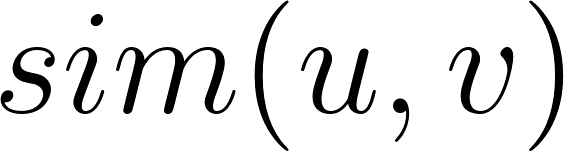
We used four different variations of KNN. They are KNN Basic, KNN with Means, KNN with Z-Score, KNN Baseline.

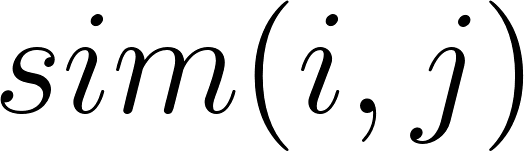
Notation:

*k*: The maximum number of neighbors to take into account

 : The set consisting of at most k neighbors of user *u* who have rated item *i*

: The set consisting of at most k neighbors of item *i* rated by user *u*

: Similarity of user *u* to user *v*

: Similarity of item *i* to item *j*

We used the following similarity metrics: Mean Squared Difference similarity (MSD), Cosine Similarity, Pearson Correlation Similarity, Pearson Correlation with Baseline Similarity.

Accuracy Metric used: RMSE (Root Mean Squared Error)

i) KNN Basic:

: Predicted rating of user *u* for item *i*

User Based:

https://lh6.googleusercontent.com/jcM_EL6x-oVr8UgFDhoLxmvQyB982YZX_NRDRI56jYNejexumqGAEDzDRBkVv-Ud5gYCuoZOI7_fl5UUzONvsvzdJIQnXe4EqSbTMIfuabpljw0V0Kj6szH5HicdQSqSrGrXm3ft[https://lh4.googleusercontent.com/pXXh78jmz_95w59O4twn5TbFY9mdBpHES0JFylX6ST9WsRhWVf8oXAY-Ds0y29BYA24X65JuPKhtysOFzGznNlFptFQQDc2O6NTtMkDIssuTF1FnTg2dXV-t27sdQibYYnCH7qkVhttps://lh4.googleusercontent.com/Nk7RCvWZRCN0ZuOis8vEBCfLQwJgQ3XCUvjFrvXV9oEM2el08Hg0BHpgbDLdhPs2EhUkwc1fL-ZoNBNaNWilZasW-1qqhAl-0cmMF3-zn0DJUW5zT4ySoysmTGxh4JCKXZttLcpthttps://lh5.googleusercontent.com/_6COe3l_h9qYeizBHJshpTSb9Mi0xn3_lDmCP_lkvlPcqhuIT906bt9-QCMIxFRd9jIyZWPwMrsujQY4tPiG_8cdhza1uaTdb3BXpbNtDEeNphfi4dfV0oA9Qzf4g94IKJdrQtRS](https://www.codecogs.com/eqnedit.php?latex=%20%20%250)

|  |
| --- |
| https://lh6.googleusercontent.com/007tWO-VgSzYSAuulfz3l0ga4UYzfl14kazAxKAK1EwVOlfjRLN5OKnE6_h8ckJ8iSmXsDO4GyEn191fvkxaS3QS7p9_nxX3Fmq8mLvRedaxy3ExLR_aVbdf_MOjo4ogsatLZubY  Item Based:  https://lh3.googleusercontent.com/Wdv76BmbPc880U5wLJi_MChdyPo5X8mg0pxm51f0MOFPwMDPdB03rIR8hq1Q2rYWMhNRuY8b3oFtsPd3fBCEgyZDgt-qS74oKWcj4NePmEMRb6SUWhoBfkeEZz6rMMUQ9ZQW91_z |

ii) KNN with Means:

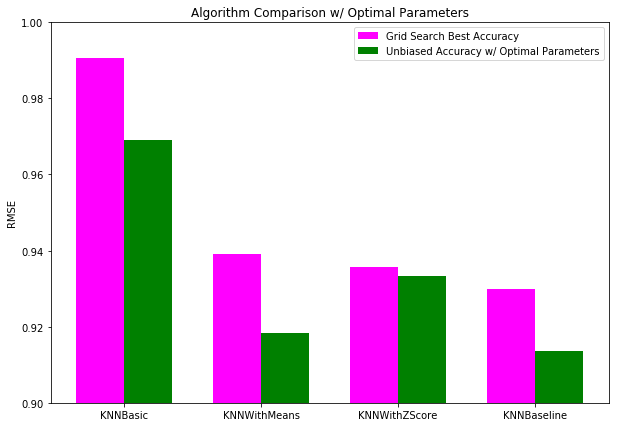
Here we take into account the mean ratings of each user and mean ratings for each item in user-based and item-based approaches respectively.

User Based:

|  |
| --- |
| https://lh6.googleusercontent.com/OdTWrRiMmuhA_LlirfNLMqRewA3hF7XFRJVjpzLiTep3i2g0x7b1AY435fxkbuct9IKyCV6UyVOexmGQ9P31-aUpGFrzhhqqVj5gInqKj6GltKG3w46efApG8ImOtZkxBszmrfog |

Item Based:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| https://lh3.googleusercontent.com/wmnCNsU25ov-HIsJWPsOL0ekcuHJLDWOUj2mctPaiKh8EJE0ne1gROi-dvuqC8NillAJ22_xbAKmcVNFDIAuL--ZgI26CZHcXzD3k-9YHTQcg6Ca3Fo-0hp7p1lEdruS1t-qjlVp  iii) KNN with Z score:  Here we take into account the z-score normalization of each user.  User Based:   |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | https://lh5.googleusercontent.com/HT0mAhshBnOqvELNGzkAK35NYEuP8h999WI_8SBw2AM4XoFrKdDjAs1ObuNMx_K0w1AQG6MMjnsovizFlJwchArMEcwduAfq74fnQdbgrPPij01dA9ETaTXPA2MTSrKxH8Dfar38  Item Based:   |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | https://lh5.googleusercontent.com/IOeZc-4goUJz9EnQnMsNON8bmsrLImpVpoGYmOANoC0QsckuKAzOcPoTSSy_LBQ4JSm169KNHBlBnh2f3dDJbzh3ddRAOOASdotueFBg5twJGv4q7yYPyEIH8LUt75a8cauUQh3h  iv) KNN Baseline:  Here we take into account the baseline ratings.  User Based:   |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | https://lh3.googleusercontent.com/h5mBMnavKu_XlWGXydelFzBLE1F6S641gEpotpbBfo5sX-HAkKv3yrs4pfCgOUjubVSPwWbVdcnVjtTmqG_PJGdg8Ma-R0ug7L08AaGr1Rk6JRnR9k11ks5oPU6vsolRkKQ0yeyv  Item Based:   |  | | --- | | https://lh5.googleusercontent.com/XjWOY9zRz4J1PYGvvB17-Xda3eE6UeoLLOVV0xeMg4_khfgaiV7X32deIFKKS1Txuk_TnMKUoTkhrHaDBY3Csq87MiNNeFrir9df7xnC6wSz90uRY9aqWmg6K_cpcmrFbBQJIxpd |   c)    Model-based:  i) NMF  NMF is an algorithm based on Non-negative Matrix Factorization. It is similar to SVD, but ensures that user and item factors are kept positive. The prediction is set as:  https://lh6.googleusercontent.com/7Bw_eU1F731OT5jhcrB7SonG7GHAwGGujdZ8eYxF1_v7D2H9Z_J_9Gdp51nDA6-2F2SLO94tNGvRnJLTzGu96aQq21hRZPTGor14zqklaZc-s4dDOQ4ENlbdcoWrWqXgWxGYEAGJ  When baselines are considered, the prediction is defined as:  https://lh5.googleusercontent.com/KvfEs7Ik2nFWr9L5ht6oD699UF1_zv5CSjV8Ps1wfbH2jFygAxx3LtkgFe2FKHSf9eLES3eL-_Hg46impn5v4ewNcacextApi8sG5oZVyLq1yDkQDxbicKpX85V-fs8qulZBu5BD  The optimization procedure is a regularized stochastic gradient descent, where the step size is specially chosen so user and item factors are positive, provided their initial values are also positive. At each iteration of SGD, the factors *f* or user *u* and item *i* are updated as follows:  : regularization parameters.  Baselines in NMF are optimized in the same way as SVD (see the following section). A number of hyperparameters can be tuned to achieve the best accuracy.  n\_factors: the number of factors  n\_epochs: the number times of SGD repeated  reg\_pu: the regularization term for users  reg\_qi: the regularization term for items  reg\_bu: the regularization term for  reg\_bi: the regularization term for  lr\_bu: the learning rate for  lr\_bi: the learning rate for  biased: use biases or not  ii)SVD  The prediction of the SVD algorithm is defined as:  https://lh5.googleusercontent.com/KvfEs7Ik2nFWr9L5ht6oD699UF1_zv5CSjV8Ps1wfbH2jFygAxx3LtkgFe2FKHSf9eLES3eL-_Hg46impn5v4ewNcacextApi8sG5oZVyLq1yDkQDxbicKpX85V-fs8qulZBu5BD  We try to minimize the following regularized squared error:  https://lh3.googleusercontent.com/KVeT3DS8j7b17irwBZwJUbtpHggRgwjV2Fr-KzQ5WZezNZaBWDWJobPpahh-nrMbwQC5-a8suMiiA89s1RzW69RUVXbTrLX1EDbFAU7Uca3GoI6wvGuQJgR77DCLJDoisgjGWu9i  Stochastic gradient descent is used to minimize the error:  https://lh4.googleusercontent.com/fy5xv9HJH4Sl--5ifFUOZoLsapjzeH0-GuuMxPiCfGEFlw3JeU0ktahn3EQNMcmSaVsXtnDZoU8m-YSMl4MER4YrJLAov0_ym-eBbby8mLFMbDVEDKnQowaj1NtkVmHtBF3Z7aEv  https://lh6.googleusercontent.com/s9jo7XdG-0FORZjaPQHknVQTEs4zYMBwG0wgnLq9a7va79wI6U4ieUWVD_Pr4Z0YTuuAeWoaOE4UJ5v1C3z0hRwE2e6Bx_poyrqy7GLEVVUcHgZqIQR1owQ6tL-J_DBFi_ROHRH3  https://lh6.googleusercontent.com/j2PRryeQGk-lKNSr834RiztenSYd62FVQBIj5OPsC-xa8BCwizVdw8oQFVgBv6rIhbPQsz0bGYByB3tDzjvk9JALefdc541L6KkP3k0wDIbOwOo_kNbWbWZ6DccJ4K19hqnHBbXy  https://lh6.googleusercontent.com/2aZxGLXpvCha39J34OPLUVlC63skNYH_-bRALAq04r_1-PGsVIW966SbIZhGSIXxAHz-itYKbEbbCOGiW6G5tZFd3zRwVbAEoxYWfQUIaPj6YNfLCnwOui-jAtjjX6JhzYFAB9ub  where . User and item factors are randomly initialized as a normal distribution. We can decide if we choose to use baseline(biases) to the algorithm. The hyper parameters are tuned by grid search to find the best algorithm.  n\_factors: the number of factors  n\_epochs: number times of SGD repeated  lr\_all: the learning rate for all parameters  reg\_all: the regularization term for all parameters  biased: use biases or not  Sampling  We sampled 250K, 500K, 750K rating matrices from 1 million rating matrices. The above rating matrices are sampled by doing random sampling among the users and movies, thus reducing the size of 1 million rating matrix.  **Results:**  Baseline  In one of the KNN algorithms, KNNBaseline, we can take into account a baseline rating. We therefore did the grid search on baselines to find the optimal combination of hyper parameters giving the best performance and used the testing dataset to check the accuracy of our tuned model. While comparing the performance of two estimation methods, ALS and SGD, we found that the overall performance is better while using ALS (RMSE is lower), 94.33%, than using SGD, 94.41%.    ALS (RMSE: 94.33%)  n\_epochs: 20  reg\_i: 1  reg\_u: 5  SGD (RMSE: 94.41%)  learning\_rate: 0.005  n\_epochs: 50  reg: 0.02  KNN algorithms  Upon tuning the hyperparameters for each variation of KNN, we got the following results.   |  |  |  |  | | --- | --- | --- | --- | | Algorithm | Best Parameters | Grid Search Best Accuracy | Unbiased Accuracy (RMSE) | | KNN Basic | Sim: MSD, item based, k = 40, min k = 5 | 0.99071 | 0.96899 | | KNN with Means | Sim: Pearson Baseline, item based, k = 50, min k = 5 | 0.93907 | 0.91831 | | KNN with Z score | Sim: Pearson Baseline, item based, k = 50, min k = 5 | 0.93562 | 0.93329 | | KNN Baseline | Sim: Pearson Baseline, item based, k = 50, min k = 5 | 0.92993 | 0.91349 |   The following graphs show the performance of all variations of KNN over all the similarity metrics.  X-axis: parameter k  Y-axis: RMSE (Accuracy Metric)  Observe that in our case, KNN Baseline performs the best among all the variations of KNN.  /Users/chintu/Desktop/Acads_Fall_Term/Personalization Theory and Practice/knngraph  The following graph shows the comparison of algorithms after getting their optimal hyper parameters by tuning.  X – axis: Algorithms  Y – axis: RMSE  Observe that KNN Baseline algorithm performs the best. | |  | | | |

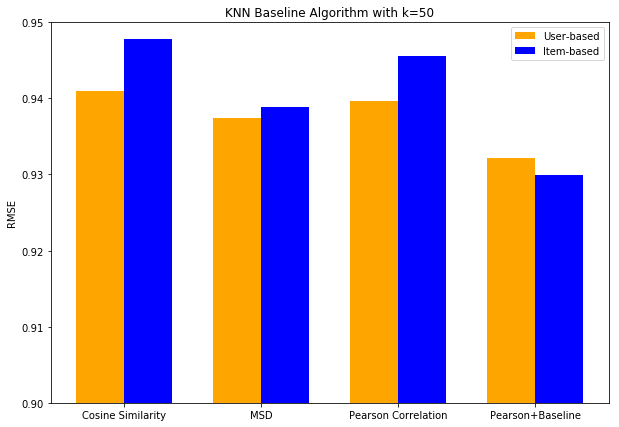


The following graph shows the performance of KNN Baseline algorithm over all the similarity metrics.

X – axis: Similarity Metrics

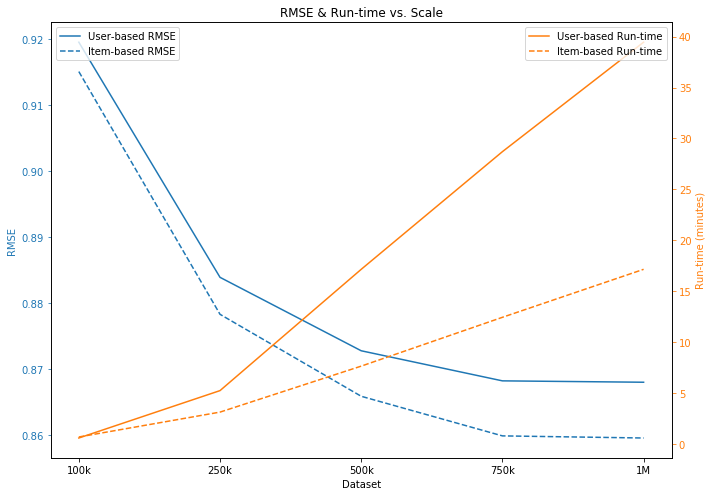
Y – axis: RMSE

Observe that Person + Baseline Similarity gives the best accuracy

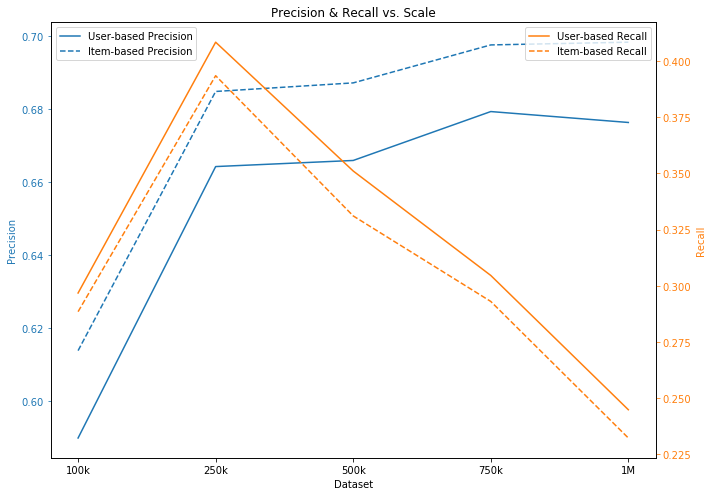


Note: As Scalability is an issue for memory based models, our business solution is to re-run the models every 24 hours instead of real time changes. So any interactions between a user and item for a given day will only be reflected in tomorrow’s model. We recalculate the matrix offline every 24 hours.

The following graph shows the accuracy, runtime of user and item based algorithms with scale of the data set.

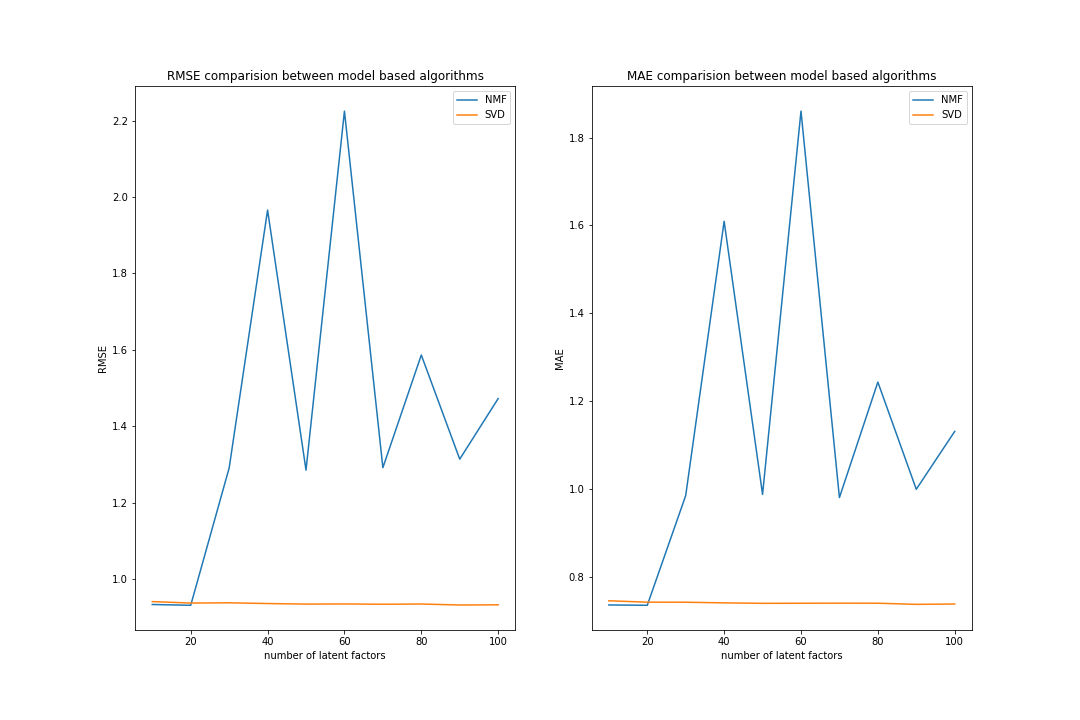


The following graph shows the Precision and Recall vs Scale variation.

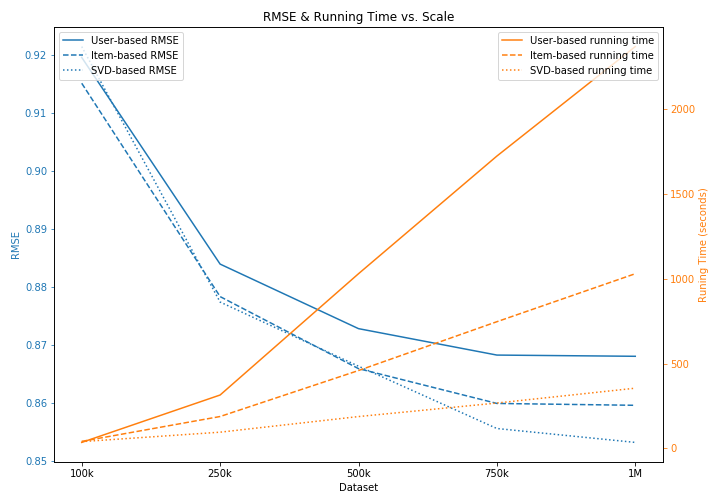


Matrix-Factorization algorithms

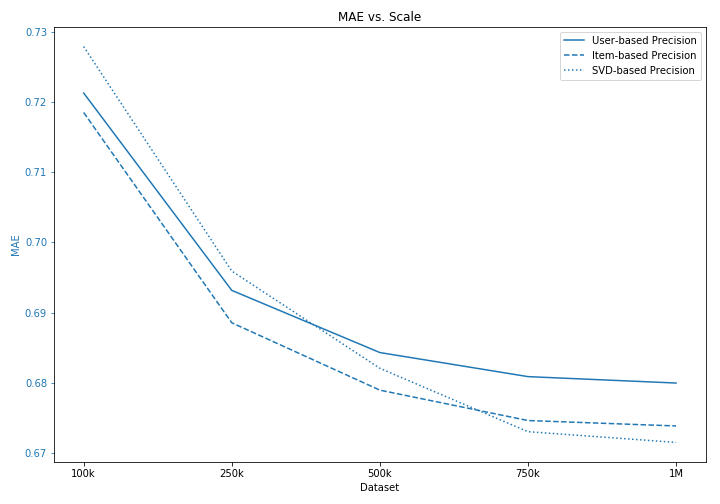
|  |  |  |  |
| --- | --- | --- | --- |
| Algorithm | Best Parameters | Grid Search Best Accuracy | Unbiased Accuracy (RMSE) |
| NMF | n\_factors=20, reg\_pu = 0.1, reg\_qi = 0.1, reg\_bu = 0.1, reg\_bi = 0.1, lr\_bu=0.001, lr\_bi=0.001, biased=True | 0.9364 | 0.9292 |
| SVD | n\_factors=80, reg\_all = 0.1, lr\_all=0.001, biased=True | 0.9216 | 0.9494 |



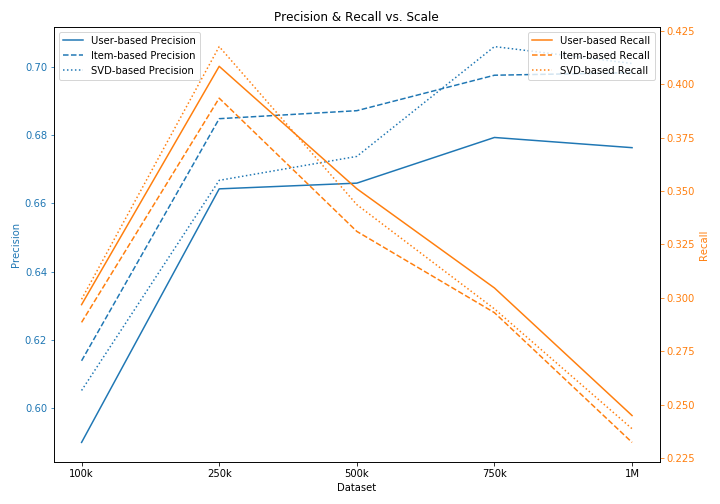
As the number of latent factors grow, NMF is more prone to overfitting and difficult to tune. However, SVD performs better with more latent factors and comparable to NMF. Therefore, based on the results of this analysis, we choose to work with SVD.



As the size of data grows, SVD performs better in terms of accuracy (RMSE) and time. For the smaller datasets, both model-based and memory-based algorithms perform equally well.



Here we use MAE as comparison metrics for the three different algorithms. Again, its trend is similar to the RMSE. When the dataset is larger, SVD generally performs better than memory-based algorithm KNN.



In comparison of memory-based and model-based algorithms, the trend of precision is similar to RMSE and MAE. On the other hand, while comparing recall of the three algorithms, the user-based algorithm performs the best.

Conclusion

As precision is more important in a recommendation system, we suggest to apply the model-based SVD algorithm, which performs better than the memory-based algorithms, to our personalized recommendation system.