A Hybrid Approach towards Improving Performance of Recommender System Using Matrix Factorization Techniques

R.G. Pawar[1], Dr. S. U.Ghumbre[2], Dr. R. R. Deshmukh[3], Dr. K. R.Kolhe[4]
[1]Department of Computer Science and Information Technology, Dr. Babasaheb Ambedkar Marathwada University, Aurangabad, India
rgpawar13@gmail.com
[2] Department of Computer Engineering, Government College of Engineering & Research,Avasari,Pune,India
shashi.ghumbre@gmail.com
[3]Department of Computer Science and Information Technology, Dr. Babasaheb Ambedkar Marathwada University, Aurangabad, India
rrdeshmukh.csit@bamu.ac.in
[4]School of Computer Engineering and Technology, Dr. Vishwanath Karad MIT World Peace University, Pune, India
kishor.kolhe@mitwpu.edu.in

Abstract

In today's digital world of information overload, the recommendation system plays an important role in the decision making of an individual as well as the collective societal level at large scale. The recommendation system is useful for leveraging the power of available data to create a better user experience and incremental revenue for the E-commerce companies. As per statistics from McKinsey about recommendation system gives 35 percent incremental revenue for Amazon, around 75 percent video consumption and 1billion $ saving for Netflix due to recommendations and around 60 percent of views on YouTube come from the leverage of powerful recommendations. This research paper shows, how a hybrid approach is useful to improve the prediction accuracy of the recommender system by using various matrix factorization techniques and regression on given sample data. Using all rating information from a large number of users and movies build a recommendation system that recommends/suggest a movie to concern user.

Keywords*: Baseline predictor, XGBoost regressor, Matrix Factorization, RMSE, MAPE*

Introduction

The recommendation system helps to connect people to items/products that they are interested in. Based on that recommendation engine predict how much people liked or disliked other items/products. In this research work, we introduce a unique approach to make a better prediction of movies from a given sampled dataset. A collaborative filtering approach is used to establish connections between users and movies based on the analysis of past transactions held. [2]

Surprise library that contains various matrix factorization techniques are combined with regression is used to predict the rating that a user would give to a movie that he has not yet rated.

Also, minimize the difference between predicted and actual rating in terms of root mean square error (RMSE) and mean absolute percent error (MAPE).

Data pre-processing and statistical data analysis

Dataset for movie recommendation taken from Kaggle for research purpose. A very huge dataset that contains four text files comprises four features like Movie IDs, Customer IDs, rating, and date. Movie titles are separating stored in movie_titles.csv file. Due to the large size of data huge storage space is required so data is stored on Google drive. Data does not have temporal nature that does not change with respect to time. Data pre-processing is the most important step of data analysis include various techniques like converting a CSV file to a data frame, removing NaN values, removing duplicates, etc.After pre-processing data frame contains around 2.6 million users and 17.7k movies. The Data frame contains a movie,user,rating, and datefeatures which are sorted in ascending order of date.Splitting data frame randomly into train data and test data in 80:20 percent so that most analysis is done on training data. After analysis on training data with respect to rating features, it clearly shows most users gives a higher rating shown in fig.1

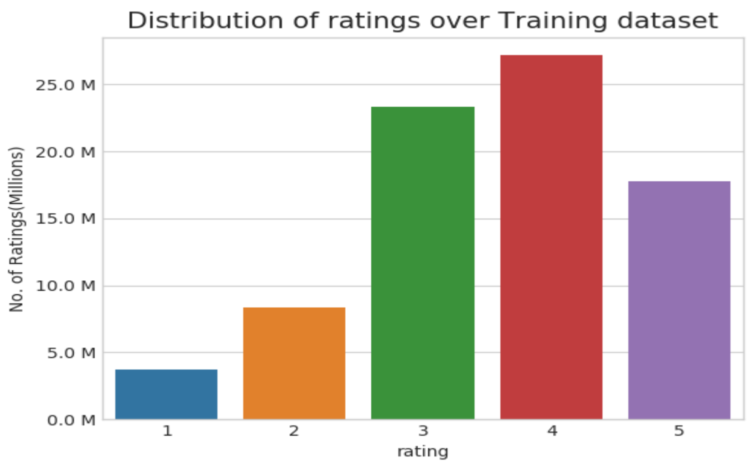


Figure.1 user's rating on movies

Adding a new feature like a day of the week to a given data frame as per the given date and checking their impact on movie rating. Finding the average rating given by the user on all available movies shown in the following table. [8] Adding new column week of the day which does not give significant values thus we dropped this features.

Table 1. Average Rating

Average Rating Day wise	
Sunday	3.594144
Monday	3.577250
Tuesday	3.574438
Wednesday	3.583751
Thursday	3.582463
Friday	3.585274
Saturday	3.591791

Sparse Matrix:from a given data frame a matrix is created which is very sparse for training as well as testing dataset. The sparse matrix contains comprised of mostly zero values. The Sparsity of the matrix is calculated by using the following formula

$$\text{Sparsity} = \left[1 - \frac{\text{No. of non} - \text{zero elements}}{\text{No. of users} * \text{ No. of movies}}\right] * 100$$

$$\text{Sparsity} = \frac{\text{No .of zero elements}}{\text{Total number of elements}} * 100 \qquad (1)$$

Table 2. Sparsity of data

Data	Sparsity	Execution time
Training data	99.829270%	01:13 min
Testing data	99.957317%	00:18 min

Sparse matrices can cause problems with space and time complexity. [5] Cold start problem hinders the performance of the recommender system. Cold start problem with users and with movies means there are some new users and some movies are not present in training data. [3] So, no rating information is available for new users, and movie in training dataset will cause difficulty in providing movie recommendation.

Table 3. Cold start problem training data

Training Data	Cold start percentage	Effect
Number of users	75148 ≈ (16%)	Severe problem
Number of movies	346 ≈ (2%)	Not Severe problem

This table indicates 16 % of users are new as compare with 2% of the new movie will cause difficulty in providing quality recommendations.

Computing Similarity Matrix

User-User similarity: With our large training dataset (405K * 405K) calculating user-user similarity matrix is not so easy. It requires hugecomputing power and lots of time. With this data, the system could crash or the program stops with Memory Error.

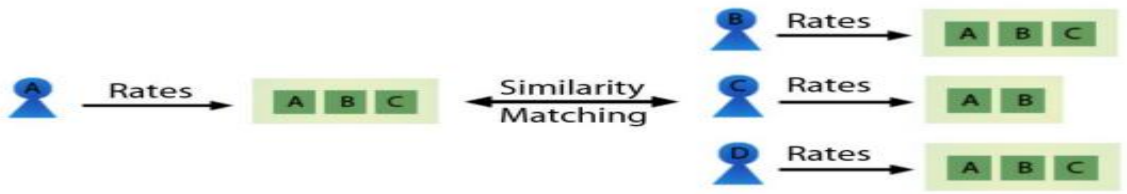


Figure.2 user-user similarity

Finding cosine similarity between user ui with ujusing following equation

cosine $(ui, uj)= ui^{t} *uj$ (2)

Finding a similarity matrix of 405 K *405 K is very expensive because of itsSparsity.

The time required to calculate similarity with k user using user-user similarity approach is given following figure 3. The time required to measure the similarity of users with k users increases as a value of k increases.

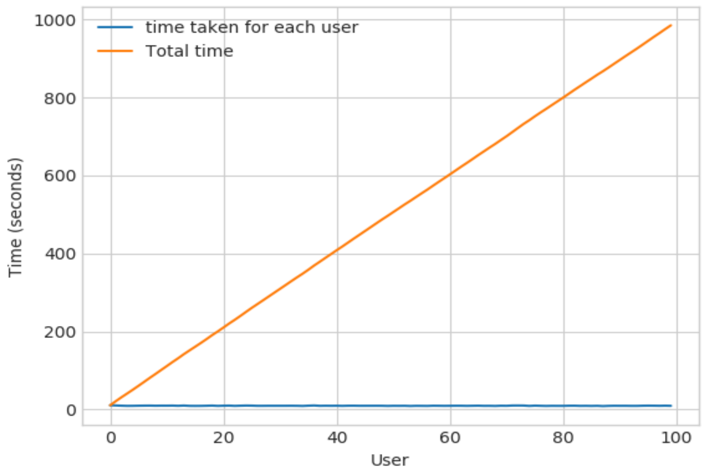


Figure. 3 the time required to measure similarity with k users

We are having 405 k users in training data set and computing similarity between them is very time-consuming. As per the plot, it took roughly 8.88 sec for computing similar users for one user so, for 405 k users it will take more than 40 days to compute similarity which is inefficient.

Using parallel computing with four core still, it will take more than 10 days which is also inefficient. So, to speed up the process we tried the singular value decomposition (SVD) method to reduce dimensions to 500 and analyzed the results. After dimensionality reduction, the time required to find similarity with 500 users still around 15 days which quite large because the matrix is dense.

Movie-Movie similarity: For this approach, we have 17k *17 k dimension matrix which is sparse. Finding cosine similarity between user mi with mj using following equation

$$\text{cosine}\,(mi,mj)= mi^{t} * mj \qquad (3)$$

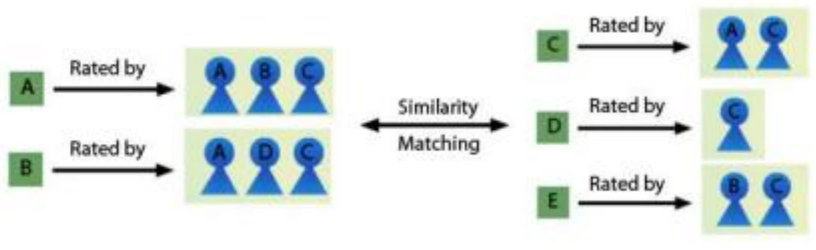


Figure. 4 Movie-Movie similarity

The time required to find movie-movie similarity on 17 K * 17 K matrix is 10 minutes which is verylessas compared with the user-user similarity approach that requires more than 10 days with four core parallel computing. Most of the time user wants to find the most similar 10 or 100movies. The time required to find the top 100 similar movies with a given movie is just around 33 seconds.CSR matrix representation is helpful to improve efficiency. Amovie-moviesimilarity approach is better than user-user similarity.[1] As a user-user similarity approach not suitable for recommendation due to time and space constraint on a large training dataset. Alsonot getting too much diversity and or surprise using a movie-movie similarityapproach, sorecommendations turn to obvious or boring.

A hybrid approach for recommendations

Various techniques of machine learning are combined systematically to train a model on the training dataset and then apply to improve the prediction accuracy of test data using a hybrid approach. Instead of using a large dataset, its sample is used so that time required to train model is reduced tremendously and improving prediction accuracy on test data by minimizing the root mean square (RMSE) value.

Figure. 5 Sample data

The following figure shows a hybrid approach is used to improve the performance of the recommended system using KNN and matrix factorization models. RMSE is one of the metrics for measuring the prediction accuracy of the model.

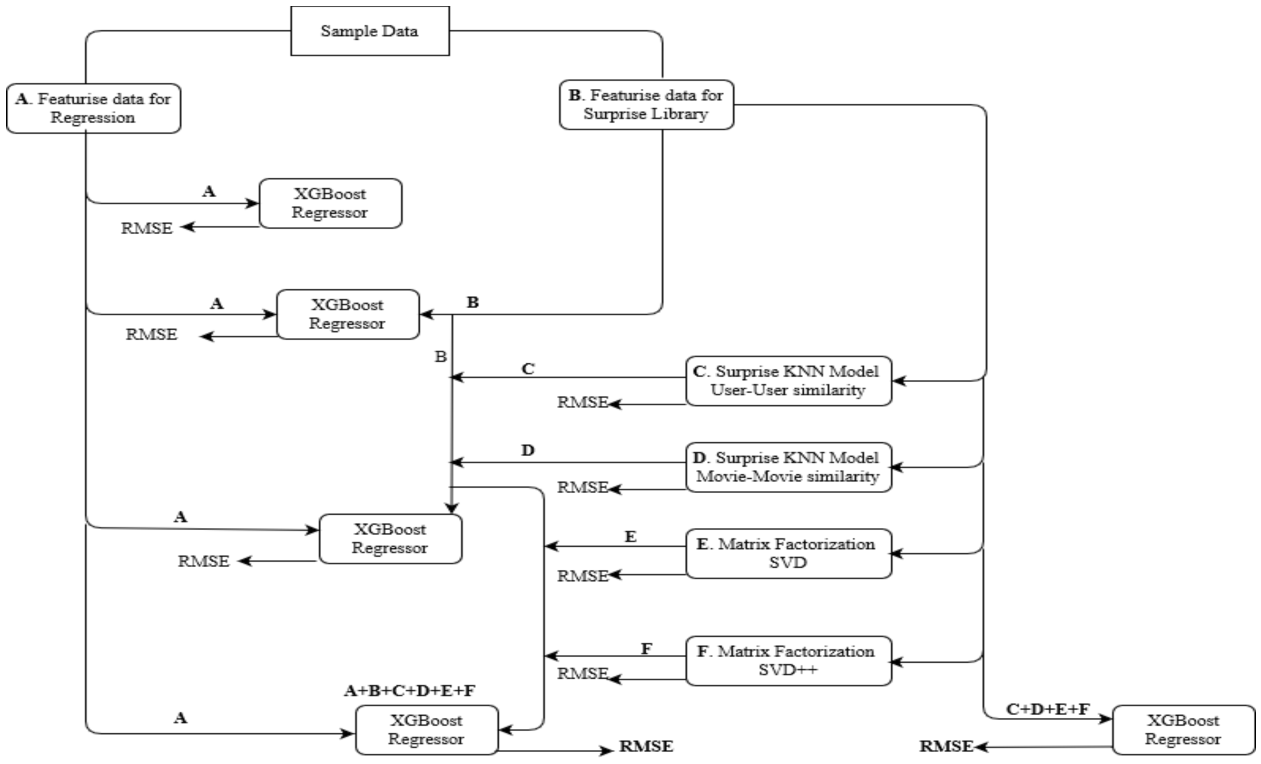

Sample Data
A. Featurise data for Regression
B. Featurise data for Surprise Library
XGBoost Regressor
RMSE
A
B
C. Surprise KNN Model User-User similarity
C
D. Surprise KNN Model Movie-Movie similarity
D
E. Matrix Factorization SVD
E
F. Matrix Factorization SVD++
F
A+B+C+D+E+F
C+D+E+F

Figure. 6 Hybrid approach of recommendation system

Various prediction algorithms come under surprise library is used to train the model along with XGBoost regressor. Surprise stand for simple python recommendation system engine which generates recommendations with explicit rating data in (ui, mj, rij) triplet format.[7]
Finding a global average of all movie ratings is the most important feature extracted from sample train data. Along with these finding average rating per user and average rating per movie on sample training dataset is calculated which is shown in following table 4.

Table 4. Features rating

Features	Rating
Global average of all movie rating	3.58167
Average rating per user(ui)	3.96551
Average rating per movie(mj)	2.64583

Baseline XGBoost regressor: The first step of a hybrid approach is to train a model that predicts the rating given by users to movies will be considered as a regression problem. Along with the global average of all movie ratings, average rating per user and movie, finding top 5 similar users who rated that movie $1u2, u3\ u4, u5$, and top 5 similar movies rated by this user like$m1, m2, m3, m4, m5$. These 13 features given as input to the baseline model. The output of this baseline model that is predicted rating is given to the XGBoost regressor as an input. The output of this baseline model that is predicted rating. This baseline trained model is applied to find the predictedrating for test data. The root mean square error (RMSE), as well as mean absolute percenterror(MAPE), is calculated to check prediction accuracy.RMSE and MAPE values are calculated by using the following equation. Here yi is actual rating and y^ is predicted rating by model.

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2} \quad \text{MAPE} = \frac{1}{n}\sum_{i=1}^{n}\frac{|Y_i - \hat{Y}_i|}{Y_i} \quad (4)$$

XGBoost regressor with the initial 13 features of the input gives following RMSE and MAPE value for test data shown in following table 5.

Table 5. Baseline XGBoost regressor output

Test data	Value
RMSE	1.07618
MAPE	34.504

A global dictionary is used to store the model name as a key and RMSE as value parameter.
Surprise baseline model: We cannot give raw data movies, users, and ratings to train the model in

Surprise library. They have a separate format for train and test data, which will be useful for training various surprise librarymodels. In Surprise library reader object able to parse the given data frame. In this work, the training dataset is formed using a Pandas data frame.Predicted rating using the surprise baseline model is calculated by using the following equation.
$\hat{r}ui=bui=\mu+bu+bi$(5)
μ is a global mean which is constant, user biasbu and movie biases bi is calculatedusing stochastic gradient descent. RMSE value and time is taken to execute train as well as test data using surprise baseline model is given in the following table

Table 6. Surprise baseline model prediction accuracy

Data	RMSE	MAPE	Execution Time
Train Data	0.93471	29.38957	1.11 min
Test data	1.07303	35.04995	0.07 min

XGBoost with initial 13 features + Surprise Baseline predictor
Training and testing dataset is updated by adding baseline predictor value from surprise baseline predictor model to XGBoost initial 13 features. XGBoost model is trained with these 14 features to find out prediction accuracy in terms of root mean square error.
RMSE value for test data using this model is 1.076341
Surprise KNN Predictors
ImportingKNNBaseline package from a surprise library. Baseline predicted rating for given user u on movie i is $\hat{r}ui=bui=\mu+bu+bi$.
Surprise KNN Predictors uses two approaches to find predicted ratings by the user on a given movie are surprise KNN baseline with user- user similarities and Surprise KNN baselinewith movie- movie similarities. Predicted rating based on user-user similarity using surprise KNN model is calculated by the following formula

$$\hat{r}_{ui} = b_{ui} + \frac{\sum\limits_{v \in N_i^k(u)} \text{sim}(u, v) \cdot (r_{vi} - b_{vi})}{\sum\limits_{v \in N_i^k(u)} \text{sim}(u, v)} \quad (6)$$

Where sim (u, v) that is a similarity between user's u and v is calculated by using Pearson's correlation method,r_{vi} a rating given by the user on a movie$i$ and bvi is the output of baseline model. [4]
Similarly predicted rating using surprise KNN baseline movie-movie approach is calculated by the following formula

$$\hat{r}_{ui} = b_{ui} + \frac{\sum\limits_{j \in N_u^k(i)} \text{sim}(i, j) \cdot (r_{uj} - b_{uj})}{\sum\limits_{j \in N_u^k(j)} \text{sim}(i, j)} \quad (7)$$

The similarity between movies i and j is sim (i, j) calculated by using Pearson's correlation method. We compare the prediction accuracy of both the KNN baseline user-user and movie-movie similarity approach for RMSE, MAPE value, and total time required to execute the particular algorithm shown on the following table.

Table 7. KNN baseline user-user and movie-movie approach

Approach	Dataset	RMSE	MAPE	Execution time
KNN baseline user-user algorithm	Train	0.33642	9.14509	2.06 min
	Test	1.07264	35.02094	
KNN baseline movie-movie algorithm	Train	0.32584	8.44706	0.9 min
	Test	1.07275	35.02269	

The time required to execute the KNN baseline movie-movie is less than the user-user algorithm because the number of users are more than movies.

XGBoost with initial 13 features + Surprise Baseline predictor + KNN Baseline predictor
Along with the initial 13 features from baseline XGBoost regressor and predictor value from surprise baseline predictor model, two more features that are prediction values come from user-user and movie-movie approach used to update training and testing dataset. These total 16 features are applied as an input to train the XGBoost regressor model and finding prediction accuracy. RMSE of test dataset using the above approach is 1.0763602
Matrix factorization techniques
Latent factor model realization depends upon matrix factorization in which high correspondence between movie and user factors leads to the recommendation. [3]
Predicted rating of Matrix factorization techniques using surprise shown below.

$$\hat{r}_{ui} = \mu + b_u + b_i + q_i^T p_u \quad (8)$$

Singular value decomposition (SVD) is a matrix factorization technique is used to solve this problem $qi^T p_u$. Here qi is a representation of movie in latent space whereas p_u representation of a user in new latent space. Matrix Factorization technique SVD is usedto reduce the number of features of the dataset from n to k dimensions.Time taken to train the model with k equal to 100 dimensions is 7.29 seconds. Prediction accuracy using this technique in terms of RMSE value on test data is 1.07260 which is better than any other algorithm still used.
SVD Matrix Factorization with implicit feedback from the user (SVD++): Explicit and implicit feedback are key parameters for movie recommendations. Implicit feedback is more valuable than explicit feedback. [6] Users spend some time on the movie page where they read information related to the movie also user clicks but such information is not available or shared by the company with external users. Still having user rated movies irrespective of its value. If the user gives some rating to a movie means user spend some time to watch movie irrespective of he like or not like it considered as implicit feedback. [9]
Predicted rating using the SVD++method is calculated by using the following formula that contains

baseline model, explicit feedback, and explicit feedback where Iu the set of all movie rated by user u and yj is a new set of item factors that capture implicit rating.

$$\hat{r}_{ui} = \mu + b_u + b_i + q_i^T \left(p_u + |I_u|^{-\frac{1}{2}} \sum_{j \in I_u} y_j \right) \tag{9}$$

RMSE value and time taken to execute train as well as test data using the SVD++ model is given in the following table

Table 8. SVD++ model prediction

Data	RMSE	MAPE	Execution Time
Train Data	0.60324	17.492850	06.38 min
Test data	1.07284	35.03817	00.07 min

XGBoost with initial 13 features + Surprise Baseline predictor + KNN Baseline predictor+ MF Techniques

Training and testing dataset updated with by adding new SVD and svd++ features from matrix factorization technique with previous data and XGBoost model are trained to predict accuracy on test data. Root mean square error (RMSE) of test data using this hybrid approach is 1.07636.

Using this approach checking the importance of all features with the F score shown in the following figure.The user average rating (UAvg) feature having the highest F score value 205 and baseline predictor (bslpr) feature with the lowest F score of value 1 indicates UAvg is the most significant of features among all other features shown below.

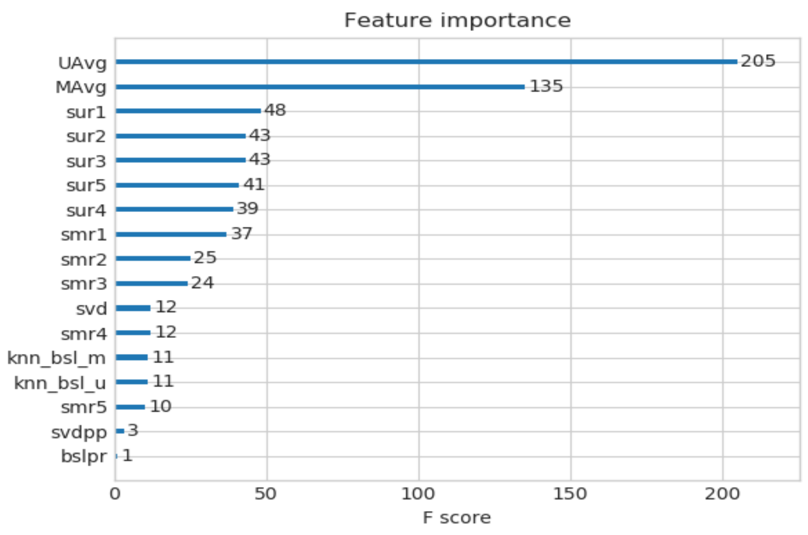
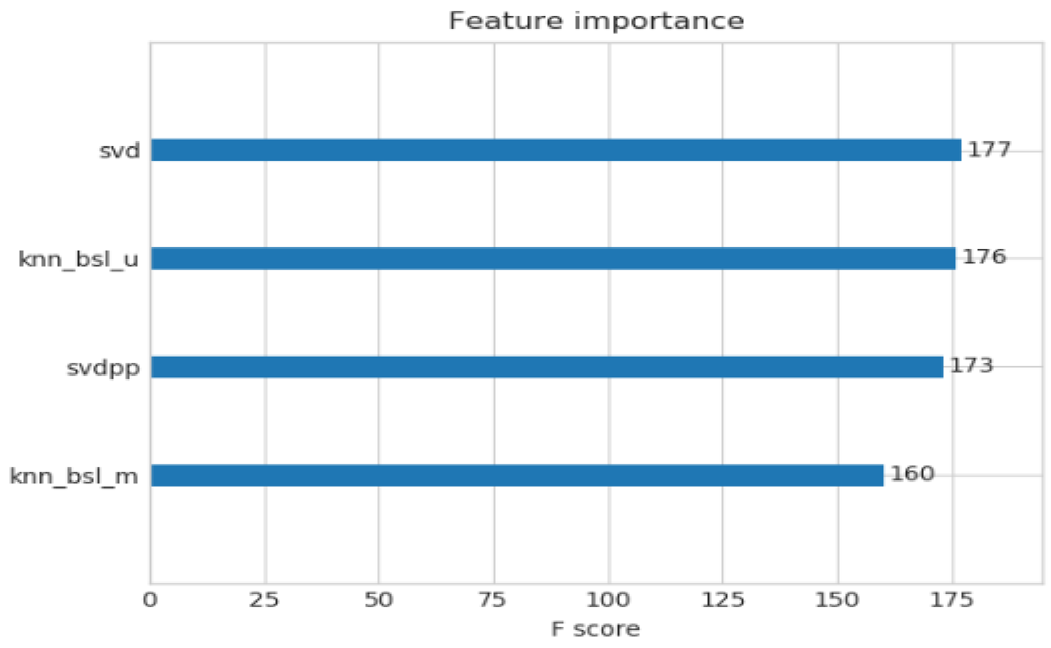


Figure. 7 F score values with all model feature

XGBoost with Surprise Baseline + Surprise KNN baseline + MF Techniques

Data set is prepared by taking all the features from the surprise library model only and XGBoost initial 13 features are excluded to train the model. The prediction accuracy of test data is calculated in terms of RMSE value. RMSE value on test data using this model is 1.075480.

Feature importancewith this approach is shown in the following figure by calculating the F- score of all features of all models under a surprise library.This indicates that SVD is the most important feature compared with all other features.



.

Figure.8 F score values with all surprise model feature

Comparison between all models

All models are arranged in ascending order of root mean square error (RMSE) as shown in the following table.

Table 9. RMSE value of models

Features	RMSE Value
Svd	1.072604687
knn_bsl_u	1.072649374
knn_bsl_m	1.072758833
svdpp	1.072849194
bsl_algo	1.073033026
xgb_knn_bsl_mu	1.075322928
xgb_all_models	1.075480664
first_algo	1.076185147
xgb_bsl	1.076341906
xgb_final	1.076358098
xgb_knn_bsl	1.076360247

From this table, we can conclude that singular value decomposition (SVD) of matrix factorization is a better model than other models as it has a lower RMSE value than other models.

Conclusion and Future Work

Singular value decomposition is a matrix factorization technique gives better prediction rating as compare with other technique.Comparing the performance of SVD model with thelastmodel xgb_knn_bsl then SVD has 0.34 % better RMSE value than xgb_knn_bsl withoutdoinghyper parameter tuning which is also significant in terms of the businessvalue of theCompany.

Feature work with this research includes training the model on the whole datasetinstead of taking sample data as well as doing hyper parameter tuning that leads to a significantreduction of RMSE value.

References

1. Kaggle Competition – https://www.kaggle.com/c/santander-product-recommendation
2. **"**Factor in the Neighbors: Scalable and Accurate Collaborative Filtering " by Hayuda Koren in ACM Transactions on Knowledge Discovery from Data, Vol. 4, No. 1, Article 1, January 2010
3. **"**Matrix Factorization Techniques for Recommender Systems" Published by the IEEE Computer Society 0018-9162/09/$26.00 © 2009 IEEE
4. Adomavicius, G. And Tuzhilin, A. 2005. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. IEEE Trans. Knowledge. Data Eng. 17, 6, 734–749
5. Sarwar B., Karypis G., Konstan J., Riedl J., "Item-based Collaborative Filtering Recommendation Algorithms," Published in the Proceedings of the 10th international conference on World Wide Web, Hong Kong, ACM 1581133480/01/0005, ©ACM, May 15, 2001
6. R. Bell and Y. Koren, "Scalable Collaborative Filtering with Jointly Derived Neighborhood Interpolation Weights," Proc. IEEE Int'l Conf. Data Mining (ICDM 07), IEEE CS Press, 2007, pp. 43-52
7. S. Funk, "Netflix Update: Try This at Home," Dec. 2006; http://sifter.org/~simon/journal/20061211.html
8. AselaGunawardana "A Survey of Accuracy Evaluation Metrics of Recommendation Tasks "Journal of Machine Learning Research 10 (2009) 2935-2962
9. Collaborative Filtering for Implicit Feedback Datasets; Yifan Hu, Yehuda Koren, Chris Volinsky 2008 Eighth IEEE International Conference on Data Mining
10. "Matrix factorization and neighbor based algorithms for the Netflix prize problem" by DomonkosTikkRecSys '08: Proceedings of the 2008 ACM conference on Recommender systems