Introduction to Computing in Artificial Intelligence and Machine Learning
Project

Latent Factor Models for Collaborative Filtering based Recommendation System

Prajakta Borse, Vinayak Doifode, Nandesh Kumar K M, and Ashish Rout

Indian Institute of Science

December 12, 2021

Abstract

Recommender systems have been used extensively in recent years in order to enhance customer satisfaction and business throughput. Existing recommendation systems can be broadly divided into two classes : Content Based Filtering and Collaborative Filtering. In this work, we experiment with a set of popular collaborative filtering techniques called latent factor models which are based on learnable matrix factorization. We begin with classical matrix factorization techniques such as Funk SVD, SVD++ and Asymmetric SVD followed by more recent deep learning based method called Neural Matrix Factorization to recommend personalized movies to each user. Further we implement a post-hoc explanation generation model called Fast Influence Analysis to make the recommendation system trustworthy. The results suggest that classical techniques like asymmetric SVD outperforms Neural Matrix Factorization. Through qualitative and quantitative analysis, we conclude that our recommendation system can generate reasonably good recommendations and back it up with explanations.

Keywords
Matrix Factorization, Stochastic Gradient Descent, Collaborative Filtering, Neural Networks, Recommendation Systems, Fast Influence Analysis

1 Introduction

With exploding amounts of products available in the market, it has become increasingly more important to suggest personalized and right product to every consumer as quickly as possible. This necessitates the need for building strong recommendation systems which can lead to better business outcomes. While recommendation systems can be built in a variety of ways, two schools of thought stand out: Content Based Recommendation Systems, in which item descriptions or attributes play a significant role in determining a user's preference for a product, and Collaborative Filtering (CF) Based Recommendation Systems, in which prior purchase behaviour or feedback from the user is taken into account. Further, collaborative filtering techniques can be broadly split into two sections : Memory-Based Methods, which consider similarity between users or items for recommendation and Model-Based Methods.In the Matrix factorisation technique, the user and item vectors are projected onto the shared latent space. The inner product of their latent vectors models the user's interaction for a given item.

2 Methodology

Preliminaries

A rating r_{ui} indicates the preference by user u of item i, where high values mean stronger preference. We distinguish predicted ratings from known ones, by using the notation $\hat{r_{ui}}$ for the predicted value of r_{ui}. Each user u is associated with two sets of items, one is denoted by $R(u)$, and contains

all the items for which ratings by u are available. The other one, denoted by $N(u)$, contains all items for which u provided an implicit preference.

Method

Matrix factorization methods are used in recommender systems to derive a set of latent factors, from the original user-item rating matrix, to characterize both users and items by these vectors of user and item factors. Let's denote the rating matrix as R, whose rows are users and columns are movies and values in the cells are the corresponding ratings. Matrix factorization associates each user and item with a real-valued vector of latent features. The user-item interaction is modeled as the inner product of the latent factor vectors. An approximation of the rating of a user u on an item i can be derived as the inner product of their latent factor vectors. Let p_u and q_i denote the latent vector for user u and item i, respectively; Matrix factorization estimates an interaction $\hat{r_{ui}}$ as the inner product of p_u and q_i :

$$\hat{r_{ui}} = {p_u}^T.q_i = \sum_{k=1}^{K} p_{uk}q_{ik} \tag{1}$$

where K denotes the dimension of the latent space.

There are multiple approaches to matrix factorisation like SVD, QR etc. However, these techniques work only with entirely filled (dense) matrix. This problem can be alleviated with the techniques employed in this work such as Funk SVD, SVD++ or Asymmetric SVD. The aforementioned techniques learn optimal parameters through minimization of regularized least squares error with the help of Stochastic Gradient Descent using the observed ratings. The aim is to build a generalizable model that can adequately capture user-item interaction in order to provide relevant and personalized movie recommendations.

Funk Matrix Factorisation[1]

Typical Collaborative filtering data exhibit large user and item effects – i.e., systematic tendencies for some users to give higher ratings than others, and for some items to receive higher ratings than others. It is customary to adjust the data by accounting for these effects, which we encapsulate within the baseline estimates.A baseline estimate for an unknown rating r_{ui} is denoted by b_{ui} and accounts for the user and item effects:

$$b_{ui} = \mu + b_u + b_i \tag{2}$$

where μ denote the overall average rating and The parameters b_u and b_i indicate the observed deviations of user u and item i, respectively, from the average , i.e user bias and item bias. Supposing that the predicted rating by user$u$ to item i is $\hat{r_{ui}}$, it can be calculated by the following formula:

$$\text{predicted Rating: } \hat{r_{ui}} = \mu + b_u + b_i + {p_u}^T.q_i \tag{3}$$

$$\text{(Loss function)} L = (\hat{r_{ui}} - r_{ui})^2 + \lambda({b_i}^2 + {b_u}^2 + ||q_i||^2 + ||p_u||^2) \tag{4}$$

A stochastic gradient descent technique can be applied to optimize the bias and latent factor vectors of users and items.

SVD++[2]

More accurate results can be obtained through integration of implicit feedback. Implicit feedback doesn't require users to explicitly rate movies rather collected automatically through observation of user behaviour such as click on advertisements or whether a user has watched movie trailer or nor. As we use MovieLens100K dataset here, such independent implicit feedback is not available to us. However, to demonstrate the proof-of-concept of implemented techniques, we generate a implicit

feedback data from rating matrix through binarization (i.e. implicit data matrix will contain 1 on cells on which user has provided ratings and 0 elsewhere).

$$\hat{r_{ui}} = \mu + b_u + b_i + {q_i}^T . \left(p_u + \frac{1}{\sqrt{|N(u)|}} \sum_{j \in N(u)} y_j \right) \tag{5}$$

$$\text{(Loss function)} L = (\hat{r_{ui}} - r_{ui})^2 + \lambda({b_i}^2 + {b_u}^2 + ||q_i||^2 + ||p_u||^2 + \sum_{j \in N(u)} ||y_j||^2) \tag{6}$$

where $N(u)$ is the set of implicit feedback (the set of items user u rated), and y_j is the implicit latent factor vector for item j. As in Funk SVD, the parameters can be optimized through Stochastic Gradient Descent technique.

Asymmetric SVD[2]

Previously described methods such as Funk SVD and SVD++ suffer from the problem of cold-start i.e inability to recommend movies to new users without model re-training. These models use user latent factor vector explicitly, rendering them incapable to find latent factor vector for new user without model re-training.

Asymmetric SVD bypasses this problem through the parametrization of each user through the implicit and explicit latent factor vectors of items that they have interacted with previously. Here each item i is associated with three factor vectors, q_i, x_i ,$y_i \in R_f$. Instead of providing an explicit parameterization for users, we represent users through the items that they have preferred previously. This model has the added advantage of being a low parameter model under settings where number of users far exceed number of items, which is pretty common in real world datasets.
Thus, user factor p_u was replaced by the sum $\frac{1}{\sqrt{|R(u)|}} \sum_{j \in R(u)} (r_{uj} - b_{uj}) x_j + \frac{1}{|N(u)|} \sum_{j \in N(u)} y_j$ leading to following model:

$$\hat{r_{ui}} = \mu + b_u + b_i + {q_i}^T \left(\frac{1}{\sqrt{|R(u)|}} \sum_{j \in R(u)} (r_{uj} - b_{uj}) x_j + \frac{1}{\sqrt{|N(u)|}} \sum_{j \in N(u)} y_j \right) \tag{7}$$

$$\min L = (\hat{r_{ui}} - r_{ui})^2 + \lambda \left({b_i}^2 + {b_u}^2 + ||q_i||^2 + \sum_{j \in N(u)} ||y_j||^2 + \sum_{j \in R(u)} ||x_j||^2 \right) \tag{8}$$

Neural Collaborative Filtering[3]

The previously described techniques such as Funk SVD, SVD++ and asymmetric SVD involved the use of inner product as interaction function between user and item latent factor vectors. However, because of the simplistic nature of inner product, it might act as a limitation to model the user item interaction. Hence, He et al proposed a deep learning based approach called Neural Collaborative Filtering (NCF).

As shown in Figure 1, NCF framework involves generation of randomly initialized user and item latent vectors, followed by their concatenation and multiple layers of neural networks in order to predict the rating that a user will provide to a movie.

In order to obtain the best of both linear and non-linear interaction between user and item latent factor vectors, Neural Matrix Factorization was proposed which involved generation of separate latent factor vectors for nonlinear and linear interaction function followed by their concatenation in the last hidden layer to predict the ratings. This procedure is represented in Figure 2.

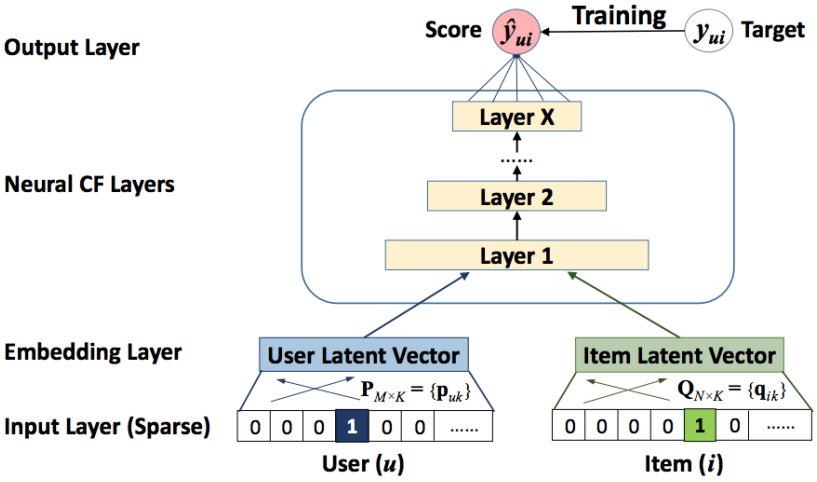

Figure 1: Neural Collaborative Filtering Framework

Fast Influence Analysis[4]

Influence analysis is a technique in statistics widely used for evaluating the effect of training data point on the prediction of a machine learning model on test data point. The influence is calculated through how the prediction on test data point will change if a particular training data point is removed from the dataset. However, Fast Influence Analysis (FIA) proposed by Cheng et al, overcomes this problem of retraining through calculation of an

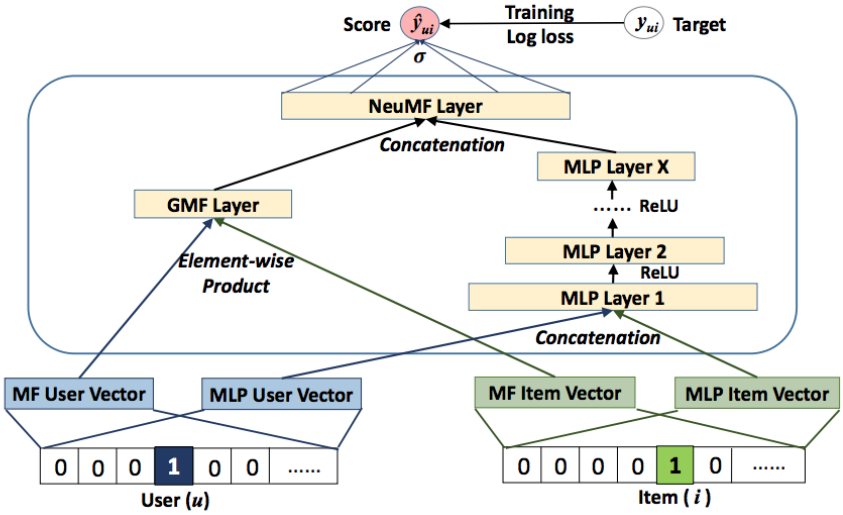

Figure 2: Neural Matrix Factorization

approximation to influence function with the use of gradient and hessian of loss functions with respect to the parameter set. Mathematically, influence of training data point z on x_t can be expressed as

$$INFL(z, x_t) = \frac{-1}{n'} \nabla_{\theta_t} L(z, \hat{\theta})^T H_{\theta_t}^{-1} \nabla_{\theta_t} \hat{y}(x_t, \hat{\theta}) \tag{9}$$

where n' is the number of training examples involving the particular user and test data point,θ_t is the concatenation of user latent factor vector and item latent factor vector for particular user and item, $\nabla_{\theta_t} L(z, \hat{\theta})$ is the gradient of loss function at z w.r.t θ_t, $\nabla_{\theta_t} \hat{y}(x_t, \hat{\theta})$ is the gradient of predicted

rating at x_t w.r.t θ_t and H_{θ_t} is the average of Hessian of the loss function at all the training points involving either the particular user or the particular item.

As the previously described matrix factorization or neural network based methods are black box models, FIA acts as a useful post-hoc explanation generation tool that can provide relevant explanations leading to a trusted recommendation system.

3 Results and Discussion

In this section, we present the results concerning the performance of different techniques described in the previous section. First, we discuss the performance of traditional learnable matrix factorization models such as Funk SVD, SVD++ and asymmetric SVD and then compare it with the results obtained from neural network based model called Neural Matrix Factorization (NMF). Finally, we present the explanations generated through the use of Fast Influence Analysis (FIA) for recommendations and evaluate it qualitatively.

The optimal hyperparameters and RMSE in rating prediction for Funk SVD, SVD++ and asymmetric SVD are presented in Table 1.

Method	Learning Rate	Reg. Parameter	Num. of Factors	Test RMSE
Funk SVD	0.1	0.2	40	0.9253
SVD++	0.1	0.2	20	0.9230
Asymmetric SVD	0.1	0.05	40	0.9141

Table 1: Optimal hyperparameters and performance of Funk SVD, SVD++ and Asymmetric SVD on MovieLens100K dataset

As described in the previous section, learning rate, l2 regularization parameter and number of latent factors are the three hyperparameters of these models. We performed grid search operation to optimize for the best combination of hyperparameters. As expected, it was observed that large learning rate (0.05 and above) leads to divergence of model from minima while small learning rate (0.001 and below) leads to slow convergence. Similarly, large regularization (0.5 and beyond) coefficient led to model underfitting while small regularization coefficient (0.005 and below) resulted in overfitting of the model. Further it was observed that when dimension of latent space extends beyond 40, model performance degrades (due to large number of parameters to be optimized) and computational complexity of model increases. These observations are consistent with theoretical understanding.

Further, the results in Table 1 suggest that asymmetric SVD performed best on our dataset, followed by SVD++ and Funk SVD. This is expected as Funk SVD is our baseline model and models like SVD++ and asymmetric SVD improve on this through incorporation of implicit feedback and alternate parametrization of user latent factor vector respectively. Further it was observed that the performance improvement achieved through these more advanced models are not that significant. This is probably due to the fact that we didn't use an independent implicit dataset and generated implicit data through binarization of explicit data. It's expected that with the use of other relevant and independent implicit data (e.g. ad-click data etc.), the model performance can further be improved. The performance of top 3 NMF model is presented in Table 2.

Sl. no	Learning Rate	Reg. Parameter	Dropout	Batch size	Test RMSE
1	0.001	0005	0.2	128	0.9239
2	0.001	0.005	0.05	512	0.9238
3	0.001	0.005	0.05	128	0.9282

Table 2: Optimal hyperparameters and performance of top 3 NMF models on MovieLens100K dataset

Learning rate and l2-regularization parameter had the same effect as described earlier for traditional learnable matrix factorization techniques. Further, we have used dropout to further regularize the

neural network. Dropout is a technique in which a fraction of hidden layer nodes is randomly removed during each iteration which results in regularization of the neural network. In our case, it was observed that dropout fraction more than 0.4 leads to slight underfitting of the model and optimal dropout fraction hovered between 0.05 to 0.20. Further we employed mini-batch gradient descent technique to optimize the neural network and hence experimented with batch size. As expected, larger batch sizes led to faster training time per epoch due to efficiency of matrix operations. This is the parameter which was observed to have the least effect on accuracy of the model. This is evident from the model 2 and 3 where despite significant change of batch size, these models were ranked close to each other among large number of all the trained models.

On comparing the performance of traditional matrix factorization techniques and NMF based models, it was observed that methods such as SVD++ and asymmetric SVD outperformed NMF models. This is consistent with prior empirical observations from other researchers [5]. We conclude that it's better to use classical learnable matrix factorization techniques for this problem as compared to neural network based models though the latter promises a lot of flexibility in modelling the interaction function between user and item latent factor vector.

In order to qualitatively estimate the quality of recommendation, a sample recommendation list has been presented in Table 3 for a particular user. The table involves a list of movies which have been rated 5/5 by the user and a list of movies recommended by our recommendation system. It was observed that our recommendation system can personalize the recommendations based on the user's past rating history.

In order to investigate the nature of latent factors learned, we present the first two dimensions of the item latent factor vectors for a few movies in Figure 3. It is expected that, similar movies should appear clustered together while dissimilar movies should be embedded far-away from each other. Point to note is that even though two movies appear together in low-dimensional 2D space, they may be far apart from each other in high dimensional latent factor space. From the figure, we observe that (A Close Shave, The Wrong Trousers) form the first cluster, (Star Wars, The Empire Strikes Back, Star Trek: The Wrath of Khan) form the second cluster while (Titanic, Forget Paris, Dirty Dancing) form the third cluster. It turns out that first cluster belongs to animation movies, while the second and third cluster belong to science-fiction and romantic drama movies respectively.

Highly Rated Movies	Genre	Recommendations	Genre
The Empire Strikes Back	Science Fiction	Star Wars	Science Fiction
Angel Baby	Romance/Drama	Titanic	Romance/Drama
Clear and Present Danger	Action/Thriller	Raiders of the Lost Ark	Action
Braveheart	Historical/Drama	Schindler's List	Historical
Dirty Dancing	Romance/Drama	A Little Princess	Family/Drama
Mr Holland's Opus	Drama	The Shawshank Redemption	Drama
Miracle on 34th Street	Comedy	It's a Wonderful Life	Comedy

Table 3: Highly rated movies by a particular user and sample recommendations by Funk SVD based model

While this representation of clustering suggests a good way to recognize similar movies, the independent latent factor dimensions are however non-interpretable. The non-interpretability of latent factor dimensions suggest that learnable matrix factorization models are black-box models where user-item interaction has been abstracted by the creation of latent factor vectors. This leads to the need of building a post-hoc interpretation engine in order to provide user with an explanation for a particular recommendation. To achieve this, Fast Influence Analysis (FIA) was implemented as described in the previous section. A sample explanation is shown in Figure 4.

The movie recommended in the Figure 4 is 'The Silence of the Lambs' which is a horror movie. The explanations involve user's prior liking towards movies like Bride of Frankenstein, Carrie and The Shining which are also horror movies. Hence we conclude that the explanations are qualitatively plausible and can be provided to users in real time with the use of Fast Influence Analysis.

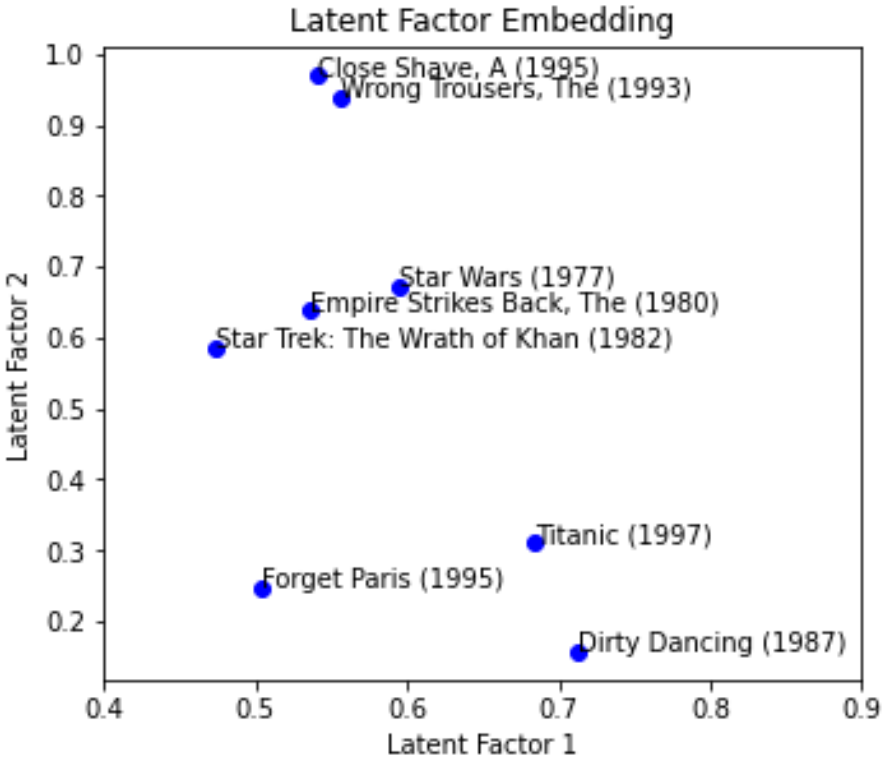

Figure 3: Projection of movies on first two dimensions of latent factor space for Funk SVD model

```
Recommended Movie is Silence of the Lambs, The (1991) for user 20


Explanations are

You rated Bride of Frankenstein (1935) as [5]/5.
You rated Carrie (1976) as [5]/5.
You rated M (1931) as [5]/5.
You rated Shining, The (1980) as [5]/5.
You rated Fargo (1996) as [5]/5.
```

Figure 4: Sample explanation generated through Fast Influence Analysis

4 Conclusions

Recommendation Systems are at the core of e-commerce and entertainment industries. In this work, several latent factor model based collaborative filtering methods such as Funk SVD, SVD++ and asymmetric SVD were implemented for the task of movie recommendation. Further, keeping up with the recent trends in collaborative filtering, a deep learning based model called Neural Matrix Factorization (NMF) was implemented and performance comparisons were made. Empirically, it was found that asymmetric SVD model performs best on our dataset despite its simplicity in comparison to NMF. Further, Fast Influence Analysis was implemented to provide useful explanations for a particular recommendation to the user, alleviating the problem caused due to the black-box nature of the aforementioned models. The results suggest that the implemented models can generate good movie recommendations and support it with necessary explanations leading to a robust movie recommendation system.

References

[1] *Simon Funk, "Netflix Update: Try This at Home", December 2006,*, https://sifter.org/simon/journal/20061211.html.

[2] *Yehuda Koren, "Factorization Meets the Neighborhood: A Multifaceted Collaborative Filtering Model", Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM Press, 2008, pp. 426-434.* https://dl.acm.org/doi/10.1145/1401890.1401944

[3] *Xiangnan He et al, "Neural Collaborative Filtering", Proceedings of the 26th International Conference on World Wide Web (WWW '17), 2017, pp. 173–182.* https://doi.org/10.1145/3038912.3052569

[4] *2. Weiyu Cheng et al, Incorporating Interpretability into Latent Factor Models via Fast Influence Analysis, KDD '19: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery Data Mining, 2019, pp. 885 – 893.* https://dl.acm.org/doi/10.1145/3292500.3330857

[5] *Steffen Rendle et al, Neural Collaborative Filtering and Matrix Factorization Revisited, RecSys '20: Fourteenth ACM Conference on Recommender Systems, 2020, pp. 240 – 248.* https://dl.acm.org/doi/10.1145/3383313.3412488