EE239AS Project 1 Report
Collaborative Filtering
Winter 2015

Team members:
Richa Shah (UID: 004408991)
Ritesh Varyani (UID: 904406389)

1. Part 1:

We made a matrix R from the MovieLens dataset of 100k ratings. The matrix had USER_ID on its rows and MOVIE_ID on its columns and had corresponding ratings of each user for the movies he had rated as the values of matrix.
There were a total of 943 users and 1682 movies, hence R became a matrix of size 943 * 1682.
Matrix R[i][j] had the rating of the movie j by the user i. If the user had not rated the movie the corresponding value of the cell R[i][j] in the matrix was put as 0.
Next we created a weight matrix W of the same size (943 * 1682). This matrix was of only 1s and 0s. We put W[i][j]=1 when R[i][j] >0 ,that is, the user i had rated that movie j. W[i][j]= 0 when R[i][j]=0: the user i did not rate the movie j.

We modified the wnmfrule.m file so that we could pass our weight matrix W to the function in that file along with our ratings matrix R. We passed the matrix R and W to the wnmfrule.m to get the least squared error which was returned as the "residual error" by the function. This rule performed a matrix factorization job which factorizes our ratings matrix R into two matrices U and V and calculates the least squared error by using the formula:

$$\min \sum_{i=1}^{m} \sum_{j=1}^{n} w_{ij}\left(r_{ij} - (UV)_{ij}\right)^2$$

We chose three values for k=10,50,100 and passed this value to the wnmfrule to obtain the least squared error. The residual error calculated by the wnmfrule function is the least squared error. The values obtained were:

k	Least Squared Error
10	234.0981
50	140.2524
100	78.1596

Table 1: Least Squared Errors for Part 1 (for different values of k)

As can be seen from the above table, the error reduces with an increase in the value of k.

---

2. Part 2:

In part 2, we swap the weight and rating matrices, since in this part we have to construct the ratings matrix in such a way that has a 1 wherever the user has rated the movie, and a 0 otherwise. Similarly, the weight matrix has the corresponding rating for each user and his corresponding movie.

When we pass this rating and weight matrix to the wnmfrule, we get the following least squared error:

k	Least Squared Error
10	1.4920
50	3.0943
100	4.7819

Table 2: Least Squared Errors for Part 2 (for different values of k)

As can be seen from the above values, the error is much lower than what was obtained in part 1.

As can be seen from the above table, the error increases with an increase in the value of k.

---

3. Part 3:

Here, we first randomly permuted the indices 1 to 100000, and then used these random ordering of indices to perform 10 fold cross validation.

For instance, for the first fold, we consider the first 1/10th of the data as testing data, and the remaining 9/10ths of data as training data. The matrix will have 0s in the place of the testing data.

We pass the matrix obtained above as the rating matrix to wnmfrule, and also a weight matrix that contains 1 wherever the rating matrix has a rating, and 0 otherwise. Using the factors obtained from wnmfrule, we construct 1/10th of the matrix containing the predicted values.

Simultaneously, we compute the average error for each fold, and each value of k.

The above procedure is repeated for 9 more folds.

At the end of 10 fold cross validation, we have 3 matrices containing predicted values of the ratings, one for each value of k.

We repeated this for the regularized version of the wnmfrule which made use of the following formula:

$$\min \sum_{i=1}^{m} \sum_{j=1}^{n} w_{ij} \left(r_{ij} - (UV)_{ij} \right)^2 + \lambda \left(\sum_{i=1}^{m} \sum_{j=1}^{k} u_{ij}^2 + \sum_{i=1}^{k} \sum_{j=1}^{n} v_{ij}^2 \right)$$

for the calculation of the error, and we also swapped the ratings and weight matrices thereby performing a 10-fold cross validation of parts 1 and 2 (in regularized form).

The average, maximum and minimum errors obtained across each fold, for each value of k for both the versions are as follows:

	Average Error	Minimum Error	Maximum Error
k = 10	0.665762	0.600867	1.193863
k = 50	0.556931	0.407526	1.861502
k = 100	0.491745	0.293973	2.224548

Table 3: Errors for Part 3 (for different values of k)

The average error reduces with an increase in the value of k.

The maximum error increases with an increase in the value of k.

The minimum error decreases with an increase in the value of k.

Lambda	k	Average Error	Minimum Error	Maximum Error
0.01	**10**	0.000673	0.000513	0.000801
0.1	**10**	0.001480	0.001256	0.001695
1	**10**	0.007970	0.007667	0.008307
0.01	**50**	0.002138	0.001927	0.002254
0.1	**50**	0.002454	0.002249	0.002574
1	**50**	0.008412	0.008085	0.008675
0.01	**100**	0.003306	0.003244	0.003375
0.1	**100**	0.003372	0.003277	0.003513
1	**100**	0.008268	0.008152	0.008426

Table 4: Errors for Part 3 after Regularization (for different values of k)

The average error increases with an increase in value of Lambda, and k.

The minimum error increases with an increase in value of Lambda, and k.

The maximum error increases with an increase in value of Lambda, and is quite similar for most values of k.

---

4. Part 4:

Part 4 involved finding the precision and recall of our algorithm and plotting graphs for precision versus recall for different values of threshold for all the values of k=10, 50, 100.

In part 3, we construct a matrix that stores the predicted values obtained for the test cases during 10-fold cross validation.

We consider 50 threshold values between 0.1 and 5 for the predicted values of ratings. For the actual ratings, we consider a fixed threshold of 3, and calculate precision and recall values, which are as follows:

	Recall Values			**Precision Values**		
Threshold Value	**k = 10**	**k = 50**	**k = 100**	**k = 10**	**k = 50**	**k = 100**
0.1	0.9999	0.9999	0.9999	0.5538	0.5538	0.5538
0.2	0.9999	0.9999	0.9999	0.5538	0.5538	0.5538
0.3	0.9999	0.9999	0.9999	0.5539	0.5538	0.5538
0.4	0.9998	0.9999	0.9999	0.5539	0.5538	0.5539
0.5	0.9998	0.9999	0.9999	0.5540	0.5539	0.5539
0.6	0.9997	0.9998	0.9999	0.5542	0.5540	0.5540
0.7	0.9997	0.9998	0.9998	0.5544	0.5541	0.5541
0.8	0.9996	0.9997	0.9998	0.5548	0.5544	0.5543
0.9	0.9996	0.9996	0.9997	0.5554	0.5552	0.5547
1.0	0.9996	0.9995	0.9996	0.5571	0.5595	0.5609
1.1	0.9995	0.9993	0.9995	0.5591	0.5657	0.5714
1.2	0.9994	0.9992	0.9993	0.5604	0.5685	0.5753
1.3	0.9993	0.9989	0.9991	0.5618	0.5711	0.5784
1.4	0.9992	0.9987	0.9989	0.5633	0.5737	0.5809
1.5	0.9990	0.9982	0.9984	0.5650	0.5762	0.5832
1.6	0.9988	0.9979	0.9980	0.5671	0.5787	0.5850
1.7	0.9987	0.9976	0.9974	0.5696	0.5817	0.5867
1.8	0.9985	0.9969	0.9968	0.5728	0.5852	0.5889
1.9	0.9982	0.9963	0.9962	0.5763	0.5897	0.5925
2.0	0.9976	0.9957	0.9953	0.5812	0.5990	0.6069
2.1	0.9969	0.9949	0.9943	0.5871	0.6117	0.6293
2.2	0.9959	0.9938	0.9935	0.5934	0.6205	0.6403
2.3	0.9948	0.9927	0.9921	0.6005	0.6296	0.6481
2.4	0.9930	0.9914	0.9909	0.6089	0.6389	0.6549
2.5	0.9904	0.9898	0.9887	0.6185	0.6481	0.6604
2.6	0.9871	0.9880	0.9869	0.6291	0.6593	0.6659
2.7	0.9821	0.9854	0.9849	0.6420	0.6713	0.6731
2.8	0.9751	0.9820	0.9824	0.6565	0.6872	0.6833
2.9	0.9656	0.9779	0.9798	0.6728	0.7070	0.7015
3.0	0.9535	0.9728	0.9768	0.6937	0.7448	0.7621
3.1	0.9368	0.9651	0.9732	0.7151	0.7888	0.8447
3.2	0.9156	0.9546	0.9693	0.7360	0.8216	0.8864
3.3	0.8881	0.9403	0.9634	0.7591	0.8523	0.9157
3.4	0.8537	0.9212	0.9556	0.7815	0.8788	0.9368
3.5	0.8119	0.8961	0.9437	0.8043	0.9022	0.9513
3.6	0.7623	0.8636	0.9252	0.8261	0.9198	0.9606

3.7	0.7045	0.8217	0.8982	0.8480	0.9339	0.9668
3.8	0.6418	0.7674	0.8557	0.8678	0.9447	0.9711
3.9	0.5746	0.7002	0.7876	0.8866	0.9524	0.9727
4.0	0.5031	0.6094	0.6377	0.9026	0.9569	0.9703
4.1	0.4296	0.5220	0.4998	0.9158	0.9602	0.9673
4.2	0.3633	0.4581	0.4423	0.9300	0.9632	0.9671
4.3	0.2995	0.4048	0.4060	0.9404	0.9662	0.9672
4.4	0.2404	0.3569	0.3801	0.9498	0.9673	0.9685
4.5	0.1901	0.3138	0.3569	0.9570	0.9674	0.9704
4.6	0.1455	0.2725	0.3340	0.9635	0.9672	0.9717
4.7	0.1095	0.2296	0.3059	0.9671	0.9654	0.9728
4.8	0.0803	0.1882	0.2693	0.9695	0.9630	0.9718
4.9	0.0564	0.1460	0.2191	0.9687	0.9594	0.9686
5.0	0.0369	0.0979	0.1268	0.9623	0.9464	0.9512

Table 5: Precision and Recall Values for Part 4 (for different values of k)

The Precision increases with an increase in Threshold value, while the Recall decreases with an increase in Threshold value.

We obtain the following plot for precision versus recall:

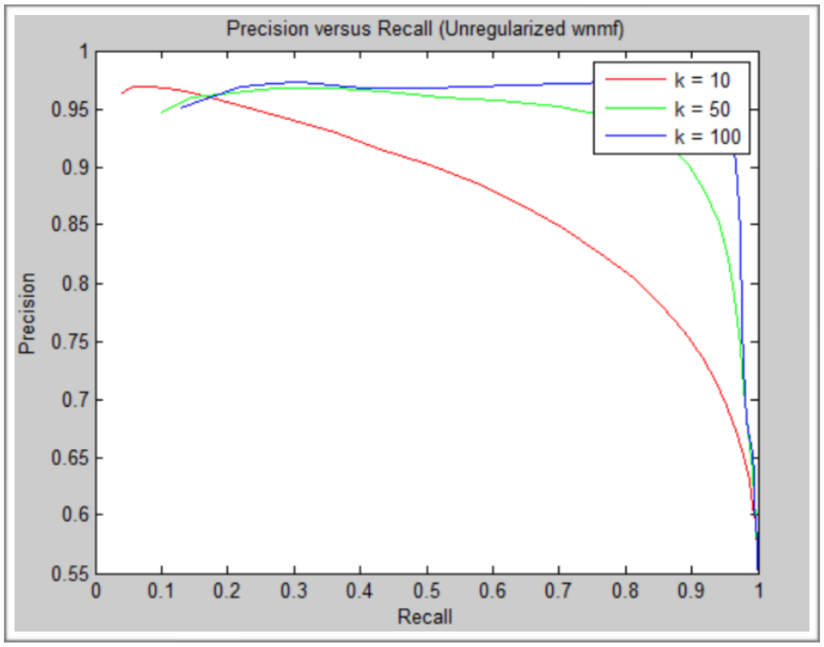

Figure 1: Precision versus Recall Plot (for different values of k)

The Precision versus Recall plot shows that Precision decreases as Recall increases, and with an increase in the value of k, the curve becomes further steep.

We obtain the following plot for precision versus threshold:

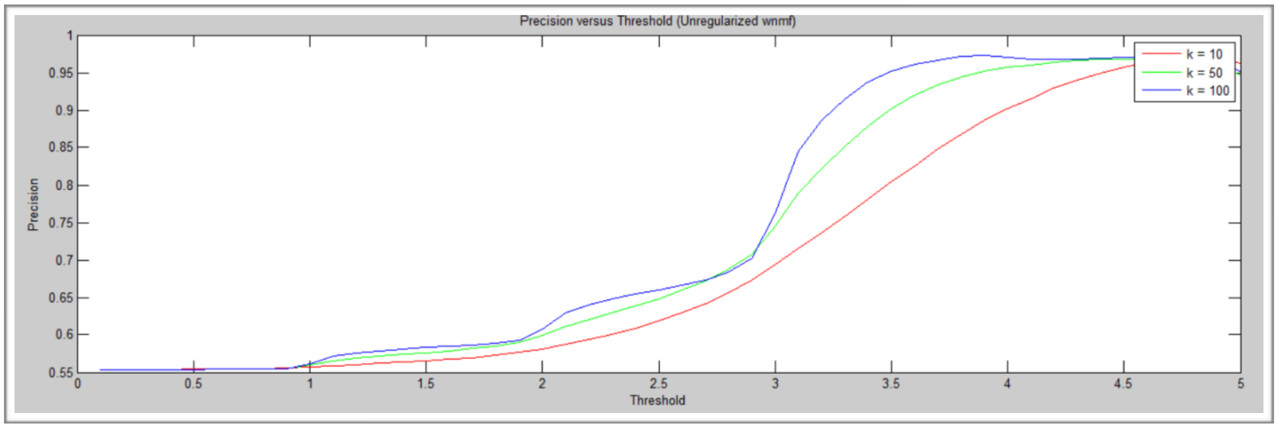

Figure 2: Precision versus Threshold Plot (for different values of k)

As can be seen above, Precision increases with an increase in threshold value. The curve becomes steeper for an increase in the value of k.

We obtain the following plot for recall versus threshold:

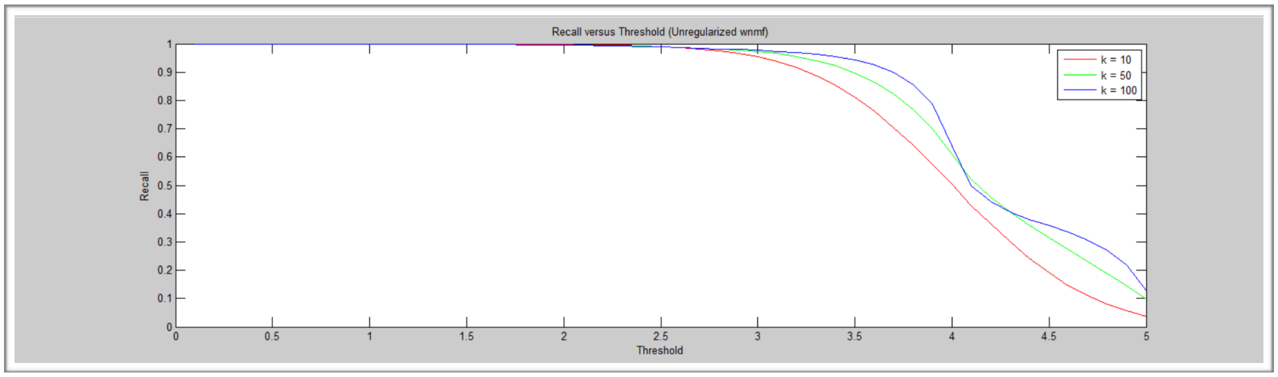

Figure 3: Recall versus Threshold Plot (for different values of k)

As can be seen above, Recall decreases with an increase in threshold value. The curve becomes steeper for an increase in the value of k.

Similarly, after swapping ratings and weights, the predicted values of the matrix we obtain, are in the range of 0 to 1.2. Hence, we consider 50 threshold values between 0.01 to 1, and compute precision and recall values, using the regularized wnmfrule function, considering three different values of λ. The mean values of Precision for different values of k and λ are as follows:

Precision	**k = 10**	**k = 50**	**k = 100**
Lambda = 0.01	0.5538	0.5540	0.5541
Lambda = 0.1	0.5541	0.5542	0.5542
Lambda = 1	0.5551	0.5551	0.5552

Table 6: Precision Values for Part 4 (for different values of k and λ)

The mean Precision value is almost the same for all values of k and λ.

The mean values of Recall for different values of k and λ are as follows:

Recall	**k = 10**	**k = 50**	**k = 100**
Lambda = 0.01	0.9882	0.9891	0.9889
Lambda = 0.1	0.9886	0.9888	0.9887
Lambda = 1	0.9878	0.9871	0.9870

Table 7: Recall Values for Part 4 (for different values of k and λ)

The mean Recall value is almost the same for all values of k and λ. But, if all values are seen for Precision and Recall, it can be seen that these values improve with an increase in threshold. For the last few values, Precision increases and Recall decreases.

We obtain the following plots for precision versus recall:

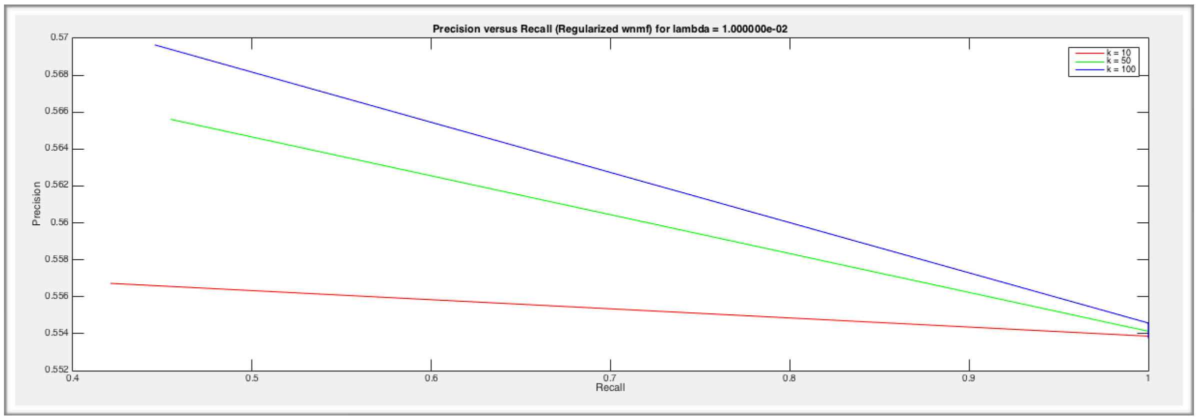

Figure 4: Precision versus Recall Plot for $\lambda = 0.01$ (for different values of k)

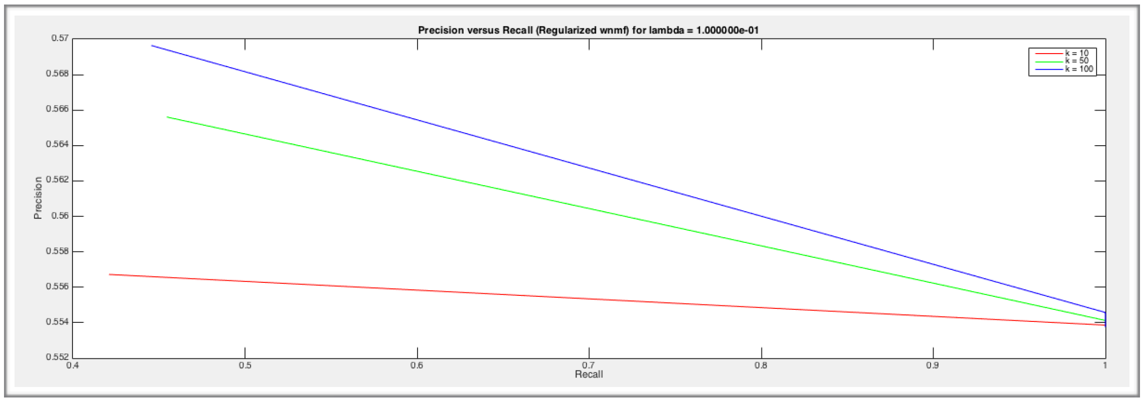

Figure 5: Precision versus Recall Plot for $\lambda = 0.1$ (for different values of k)

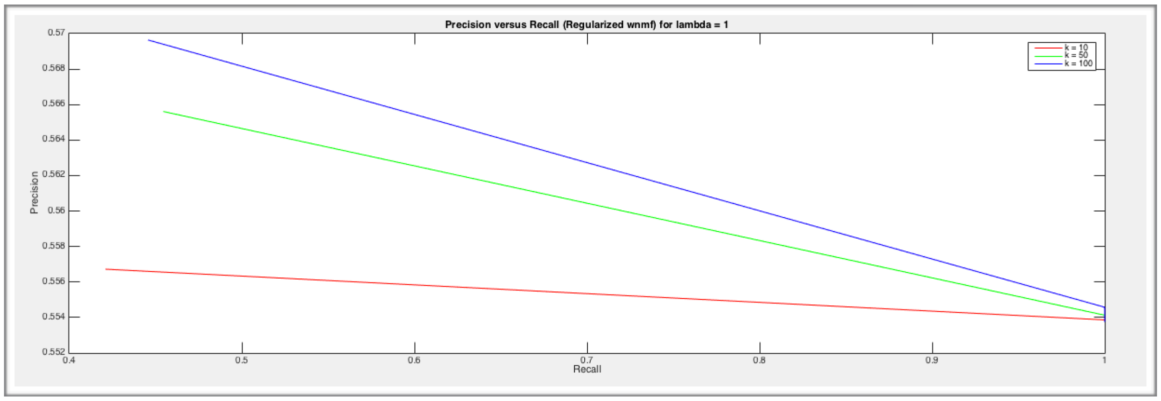

Figure 6: Precision versus Recall Plot for $\lambda = 1$(for different values of k)

The Precision versus Recall curves for all the values of λ are almost identical. With an increase in the value of k, the curve becomes more steep.

We obtain the following plots for precision versus threshold:

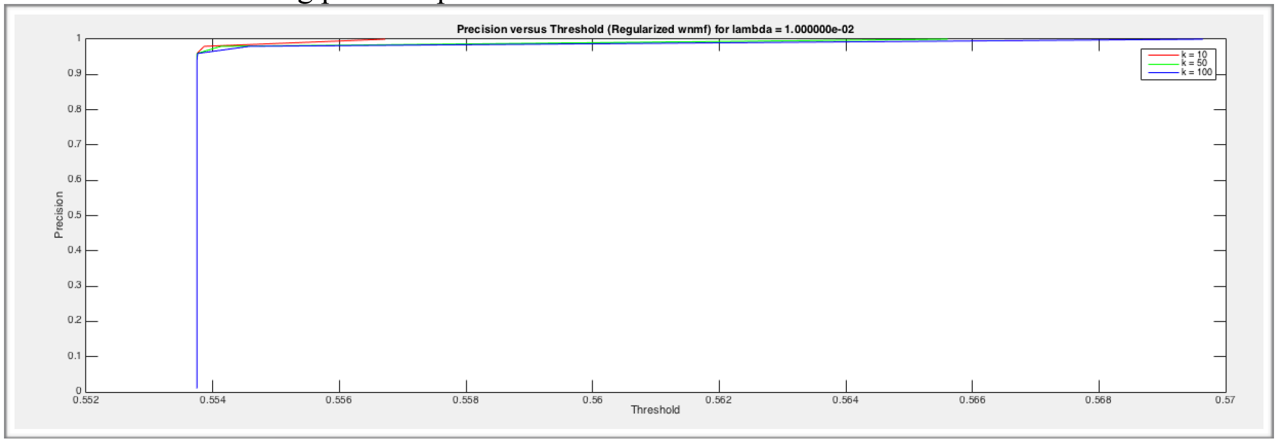

Figure 7: Precision versus Threshold Plot for $\lambda = 0.01$(for different values of k)

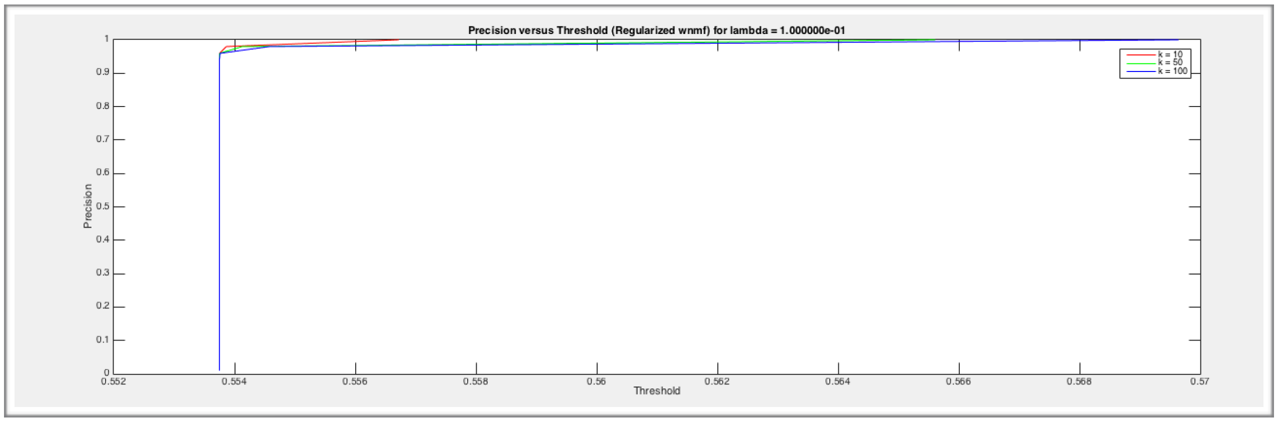

Figure 8: Precision versus Threshold Plot for $\lambda = 0.1$(for different values of k)

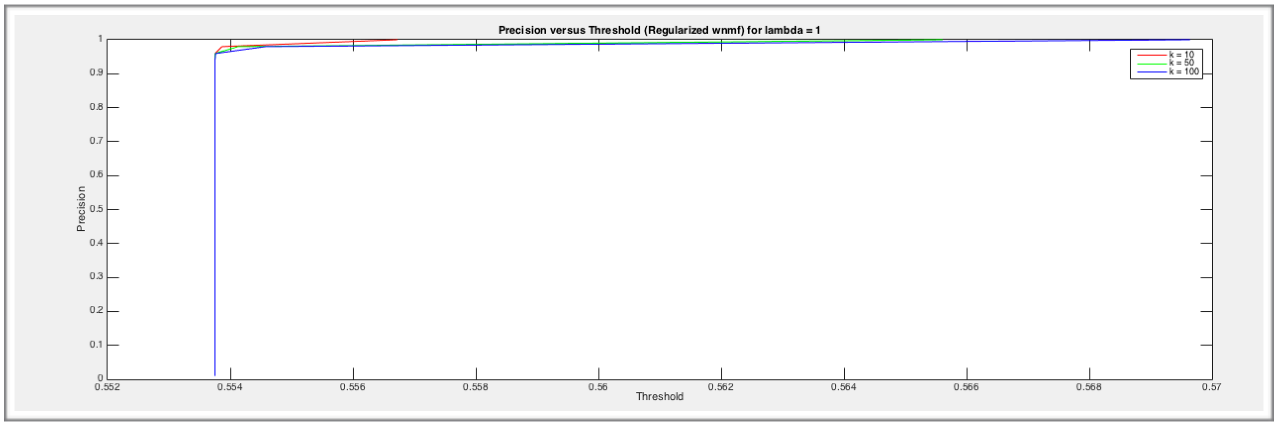

Figure 9: Precision versus Threshold Plot for λ = 1(for different values of k)

There is a sharp increase in the Precision with an increase in the Threshold, after regularization. Also, for all values of λ, the curves are almost identical. The curves for different values of k are very close.

We obtain the following plots for recall versus threshold:

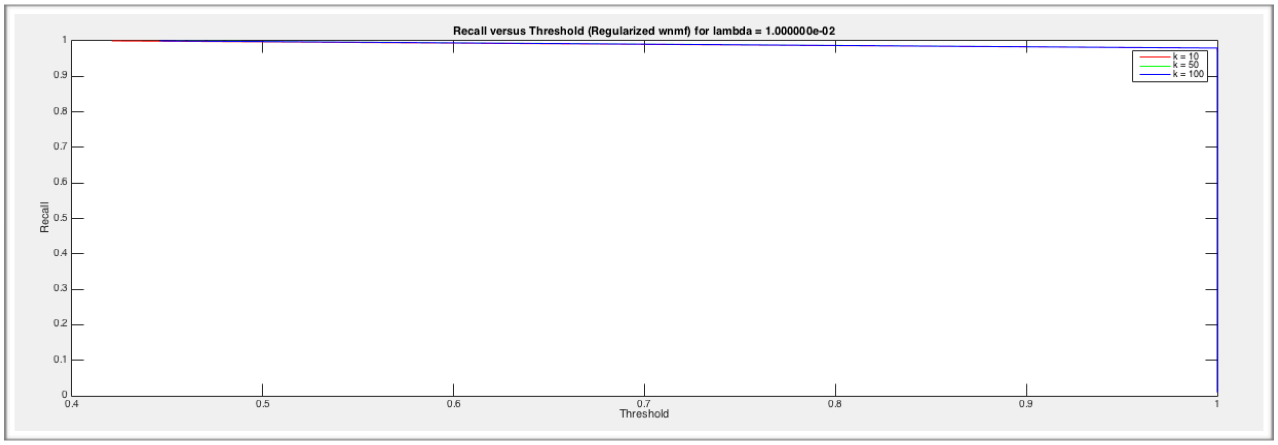

Figure 10: Recall versus Threshold Plot for $\lambda = 0.01$(for different values of k)

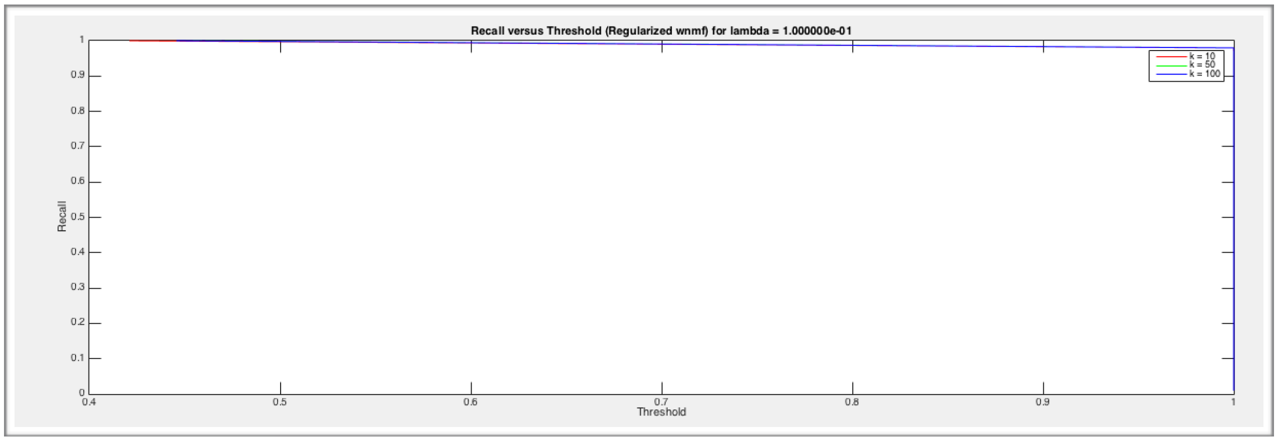

Figure 11: Recall versus Threshold Plot for $\lambda = 0.1$(for different values of k)

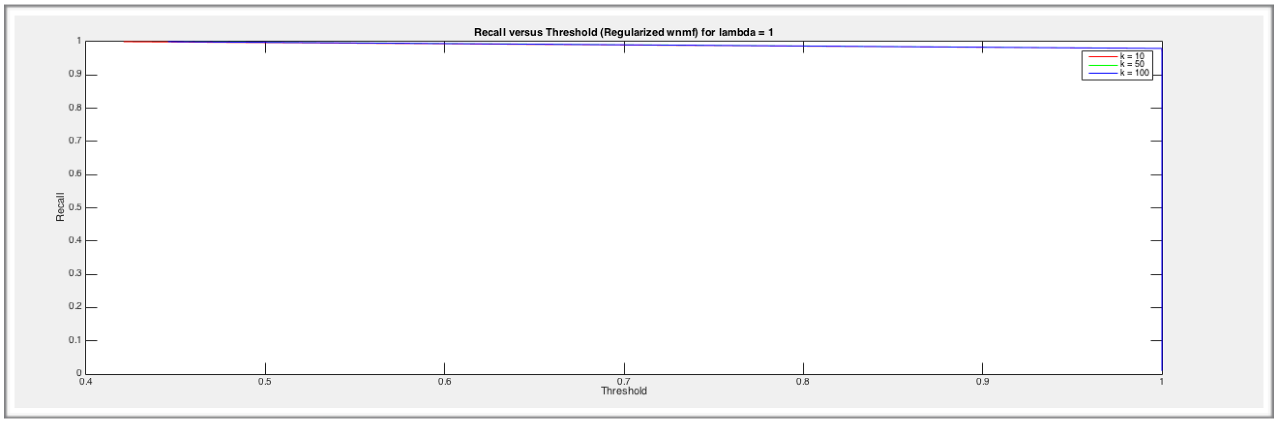

Figure 12: Recall versus Threshold Plot for λ = 1(for different values of k)

There is a sharp decrease in the Recall with an increase in the Threshold, after regularization. Also, for all values of λ, the curves are almost identical. The curves for different values of k are very close.

---

5. Part 5:

We modified the wnmfrule to add a regularization term λ.

$$\min \sum_{i=1}^{m}\sum_{j=1}^{n} w_{ij}\left(r_{ij}-(UV)_{ij}\right)^2+\lambda\left(\sum_{i=1}^{m}\sum_{j=1}^{k} u_{ij}^2+\sum_{i=1}^{k}\sum_{j=1}^{n} v_{ij}^2\right)$$

The cost function where the update occurred was changed to:

A=A.*(((W.*X)*Y')./((λ *A)+(W.*(A*Y))*Y'));

Y=Y.*((A'*(W.*X))./ (λ *Y)+(A'*(W.*(A*Y))));

where A and Y correspond to basis and coefficient matrices which are U and V in the given formulae.

We called this regularized version using different values of λ, and obtained the residual errors. These errors obtained for different values of λ and k are as follows:

Least Squared Error	**k = 10**	**k = 50**	**k = 100**
Lambda = 0.01	233.4140	232.9939	234.1133
Lambda = 0.1	139.4031	139.7761	144.2910
Lambda = 1	79.2420	79.5616	89.1737

Table 8: Least Squared Errors for Part 5 (for different values of k and λ)

After swapping the ratings and weights matrices, and calling the regularized version of the function again, we obtain the following errors:

Least Squared Error	**k = 10**	**k = 50**	**k = 100**
Lambda = 0.01	1.1635	3.1878	15.1893
Lambda = 0.1	3.0429	3.6444	15.0138
Lambda = 1	4.4442	4.5431	14.3435

Table 9: Least Squared Errors for Part 5, after swapping R and W (for different values of k and λ)

As can be seen above, the error reduces when we swap the weight and ratings matrices, but with an increase in the value of k, the error also increases.

---

6. Part 6:

This involves creating a recommendation system which recommends movies to the user.

We created a binary matrix having R[i][j]=1 when the user i had rated the movie j and 0s otherwise. We then performed 10 fold cross validation using this matrix, and using the regularized version of the wnmfrule function, and constructed the matrices containing predicted values of ratings.

This matrix was then sorted in descending order for each row, that is, for each user. We created a recommended_movies matrix which stored the recommended movies for each user for each value of lambda=0.01,0.1.1 for each k=10,50,100.

We repeated this for different values of L (ranging from 1 to 20).

Next we found the average precision of the algorithm for L=5. This was done by making use of the formula of precision:

Precision= True_Positive / (True_Positive + False_Positive)

We define the following terms for the calculation of precision, hit-rate and false-alarm-rate

True_Positive=number of instances (R>th & R_predicted>norm_th)

True_Negative= number of instances (R<=th & R_predicted<=norm_th)

False_Positive= number of instances (R>th & R_predicted<=norm_th)

False_Negative= number of instances (R<=th & R_predicted>norm_th)

where,

th= threshold set at 2

norm_th= (normalized) threshold set at 0.4

We found the precision averaged across each user for different values of k and λ, thereby obtaining 9 average precision values.

Mean Precision	**k = 10**	**k = 50**	**k = 100**
Lambda = 0.01	0.7688	0.7213	0.7160
Lambda = 0.1	0.7913	0.7552	0.7389
Lambda = 1	0.8064	0.8017	0.8036

Table 10: Precision Values for Part 6 (for different values of k and λ)

The mean Precision decreases with an increase in the value of k, and increases with an increase in the value of λ.

Next, the algorithm's hit-rate and false-alarm-rate were found out.

Hit_Rate= True_Positive/(True_Positive+False_Negative)

False_Alarm_Rate= False_Positive/(False_Positive+True_Negative)

We did this by varying the value of L from 1 to 20 by keeping the threshold and normalized threshold constant at 2 and 0.4 respectively.

We obtained the following values for hit rate versus false alarm rate, for different values of λ and k:

Mean Hit Rate	**k = 10**	**k = 50**	**k = 100**
Lambda = 0.01	0.9759	0.9630	0.9601
Lambda = 0.1	0.9790	0.9694	0.9670
Lambda = 1	0.9832	0.9778	0.9759

Table 11: Mean Hit Rate Values for Part 6 (for different values of k and λ)

Mean False Alarm Rate	**k = 10**	**k = 50**	**k = 100**
Lambda = 0.01	0.7338	0.7640	0.7553
Lambda = 0.1	0.7248	0.7268	0.7291
Lambda = 1	0.6971	0.6743	0.6619

Table 12: Mean False Alarm Rate Values for Part 6 (for different values of k and λ)

As can be seen from above, the hit rate and false alarm rate increase and decrease, respectively, with an increase in the value of λ.

We obtained the following plots for hit rate versus false alarm rate, for different values of λ and k:

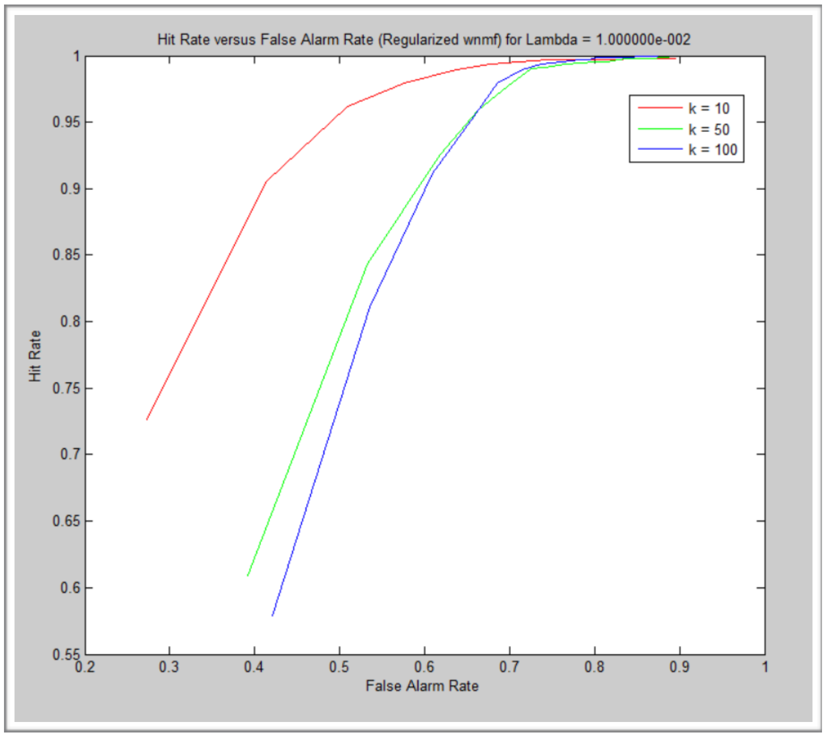

Figure 13: Hit Rate versus False Alarm Rate Plot for $\lambda = 0.01$ (for different values of k)

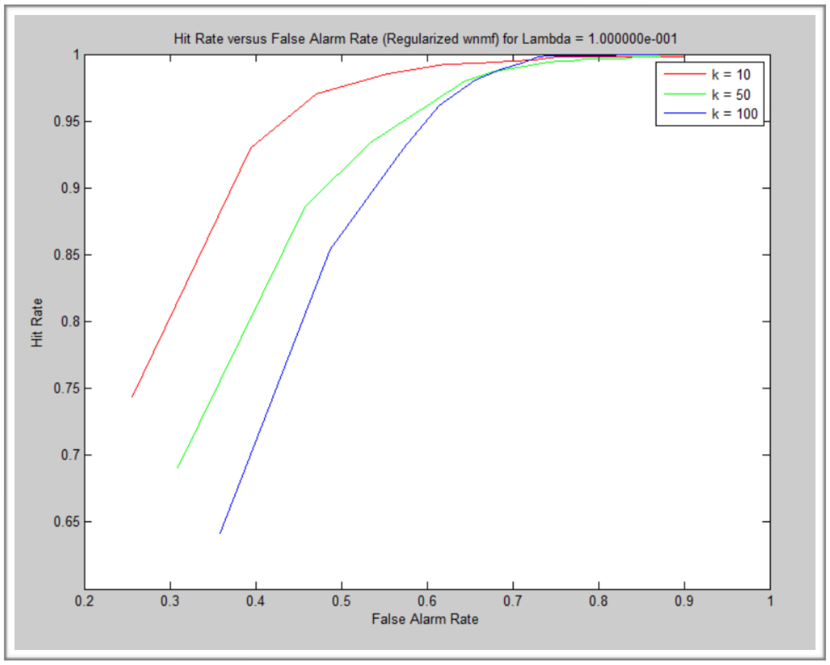

Figure 14: Hit Rate versus False Alarm Rate Plot for $\lambda = 0.1$ (for different values of k)

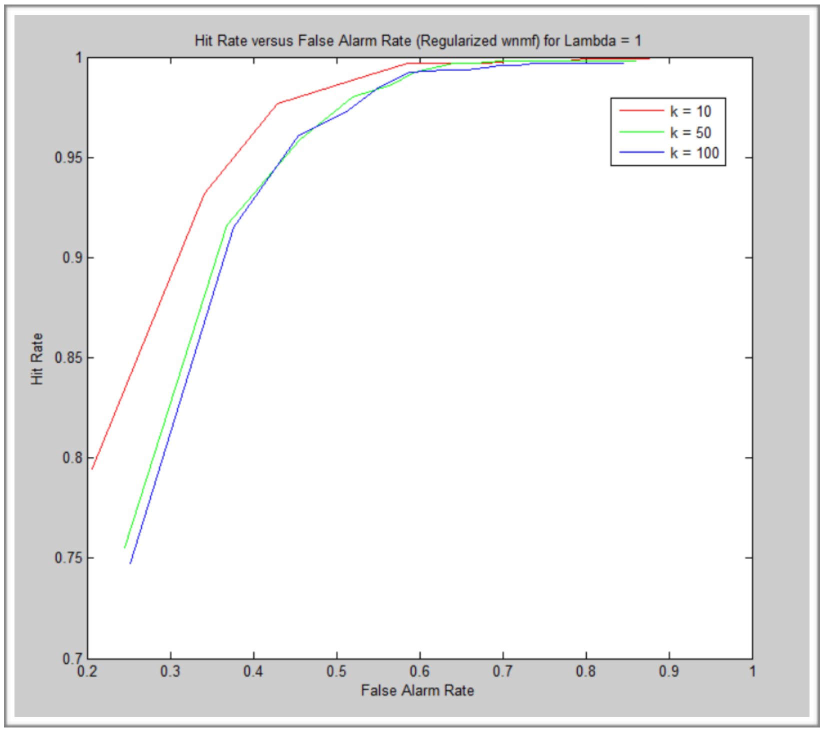

Figure 15: Hit Rate versus False Alarm Rate Plot for λ = 1(for different values of k)

With an increase in the value of λ, the Hit Rate versus False Alarm Rate curve improve, and the difference between the three different curves for different values of k decreases.
