EE239AS
Project3 Report
Collaborative Filtering

Team Members:
Yutong Zhang, 204538859
Junchi Zhang, 804593126
Zihao Zhang, 004593253
Yingze Qu, 104384503

2016 Winter

I. Part 1: Factorization Using ALS

In the first part, we used Alternating Least Squares to implement the movie recommendation system. We viewed the user opinion predicting problem as a matrix problem by putting users on rows and items on columns of the matrix. So we can formally state the problem as follows: suppose there are n users and m items, we can construct a $\mathrm{n} \times \mathrm{m}$ matrix R, where the entry (i, j) of the matrix represents the rating user i has given to item j.

However, the matrix R is very sparse, so we can do the matrix factorization to make $R_{m\times n} \approx U_{m\times k}V_{k\times n}$. We then calculate matrices U and V such that the squared error is minimized as the following equation 1-1 shows:

$$\min \sum_{i,j}(r_{ij} - (UV)_{ij})^2 \qquad \text{(equation 1-1)}$$

We can set the missing entries as 0 or NaN, and choose different k to calculate the corresponding least squared error. The results are illustrated in the following sections:

1) Set missing entries as 0

We put 0 in missing in the entries where data is missing and created a weight matrix to calculate the squared error. Assume the weight matrix $W_{m\times n}$ contains 1 in entries with data and 0 in entries where the data is missing, we then formulate the above problem as:

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

By setting k to 10, 50 and 100, we got different least squared error and residuals as illustrated in Table 1.1:

Table 1.1 squared error and residual with missing entries 0

K	10	50	100
Sq.error	2.4621e+04	8.8567e+03	2.7648e+03
Final Residual	232.6866	138.9816	78.6874

2) Set missing entries in R as NaN

By setting missing entries as NaN and using auto-generated W in wnmfrule function, the results are summarized in the following table:

Table 1.2 squared error and residual with missing entries NaN

K	10	50	100
Sq.error	2.4809e+04	9.0013e+03	2.7740e+03
Final Residual	2.3400e+02	1.4014e+02.	78.7678

As we can see from the two tables above, the factorization results and squared error are the same not matter what we set on the missing entries (NaN or 0). The reason is that if we set missing entries to NaN and use auto-generated W in wnmfrule function, the W will set 0 in its corresponding NaN entries and 1 in entries which have values.

We can see that as K increases, the squared error and residuals decrease, which is reasonable because smaller values of K imply less information which make the reconstruction less accurate but K becomes larger, the results will become more precise and more close to the original matrix. However, there is a trade-off that when K is large, it could result in overfitting in test dataset and the time consuming will also increase as a result.

II. Part 2: 10-Fold Cross Validation

In part2, we tested the recommendation system using “10-Fold Cross Validation”. The data is randomly split, 90% used for training and 10% used for testing. This can be simply implemented by putting 0 weight on the data for testing.
We trained our system and calculated the average absolute error $(|R_{predicted} - R_{actual}|)$ over testing data. By choosing different k values, we got the following results:

Table 2.1 10-fold Cross-validation error

K	cvFold	Absolute Error (Average over the testing entries)	Average Absolute Error (among 10 the tests)=
K=10	1	0.7892	0.8529
	2	0.7976	Max error = 0.7878
	3	0.8400	Min error = 1.2756
	4	0.7877	
	5	0.8025	
	6	0.8023	
	7	1.2756	
	8	0.7972	
	9	0.8169	
	10	0.8193	
K=50	cvFold	Absolute Error (Average over the testing entries)	Average Absolute Error (among 10 the tests)=
	1	0.9420	3.9003
	2	30.2449	Max error = 30.2449
	3	1.0015	Min error = 0.9407
	4	0.9977	
	5	0.9406	
	6	0.9674	
	7	1.0643	

	8	0.9494	
	9	0.9460	
	10	0.9483	
K=100	cvFold	Absolute Error (Average over the testing entries)	Total Average error = 0.9998 Max error = 1.2366 Min error = 0.9539
	1	0.9885	
	2	0.9621	
	3	0.9765	
	4	0.9967	
	5	0.9539	
	6	1.2366	
	7	0.9827	
	8	0.9649	
	9	0.9688	
	10	0.9667	

We can see from the above table that the absolute error increases with the increasement of K which is consistent with the squared error result in the previous part. Besides, we found that there is always one fold in the test sample that results a very abnormal error compared with other folds. In other words, there are some unusual error values which are quite large. In order to avoid these abnormal results, we made the upper bound of the$R_{predicted}$ 5. By adding adding $\mathrm{R_{pred}} = \min(R_{pred}, 5)$ we got the following results:

Table 2.2 optimized 10-fold Cross-validation error

K	cvFold	Absolute Error (Average over the testing entries)	Average Absolute Error (among 10 the tests)= 0.7576 Max error = 0.7689 Min error = 0.7476
K=10	1	0.7537	
	2	0.7617	
	3	0.7689	
	4	0.7644	
	5	0.7565	
	6	0.7519	
	7	0.7597	
	8	0.7467	
	9	0.7549	
	10	0.7575	

K=50	cvFold	Absolute Error (Average over the testing entries)	Average Absolute Error (among 10 the tests)= 0.9242 Max error = 0.9313 Min error = 0.9147
	1	0.9242	
	2	0.9313	
	3	0.9249	
	4	0.9309	
	5	0.9240	
	6	0.9229	
	7	0.9260	
	8	0.9147	
	9	0.9275	
	10	0.9159	
K=100	cvFold	Absolute Error (Average over the testing entries)	Average Absolute Error (among 10 the tests)= 0.9686 Max error = 0.9853 Min error = 0.9645
	1	0.9669	
	2	0.9675	
	3	0.9716	
	4	0.9645	
	5	0.9658	
	6	0.9653	
	7	0.9712	
	8	0.9570	
	9	0.9853	
	10	0.9714	

From the table above, we can see that after adding threshold 5, the result becomes uniformly. However, adding the threshold may jeopardize the idea behind our method. In order to make our results more reliable, we used the results and the method without threshold in Table 2.1 for the next parts.

III. Part 3: Precision and Recall

In this part, we tried to find a threshold on predicted entries in order to predict whether the user likes the movie or not. For the actual data points, we used the fixed threshold 3, which means that ratings less than or equal to 3 indicates the user dislikes the movie, while rating larger than 3 indicates the user likes the movie. We computed the precision and recall and plotted the precision and recall curve with different thresholds. We iterated the threshold from 0 to 5 with step of 0.01. The corresponding curves for different k are as below. Each curve corresponds to one test dataset in cross-validation.

A true positive (tp) is defined as an element in the R matrix where $rij>t\ AND\ rpij>t$ for some threshold t. A false positive (fp) is defined as an element of R where $rij \leq t\ AND\ rpij>t$ and so on. A true negative (tn) is defined as an element in the R matrix where $rij \leq t\ AND\ rpij \leq t$ for some threshold t. A false negative (fn) is defined as an element of R where $rij>t\ AND\ rpij \leq t$ and so on.

Prediction	Like	Like	Dislike	Dislike
Actual results	Like	Dislike	Like	Dislike
	True Positive(tp)	False Positive(fp)	False Negative(fn)	True Negative(tn)

$$precision = \frac{tp}{tp + fp}$$
$$recall = \frac{tp}{tp + fn}$$

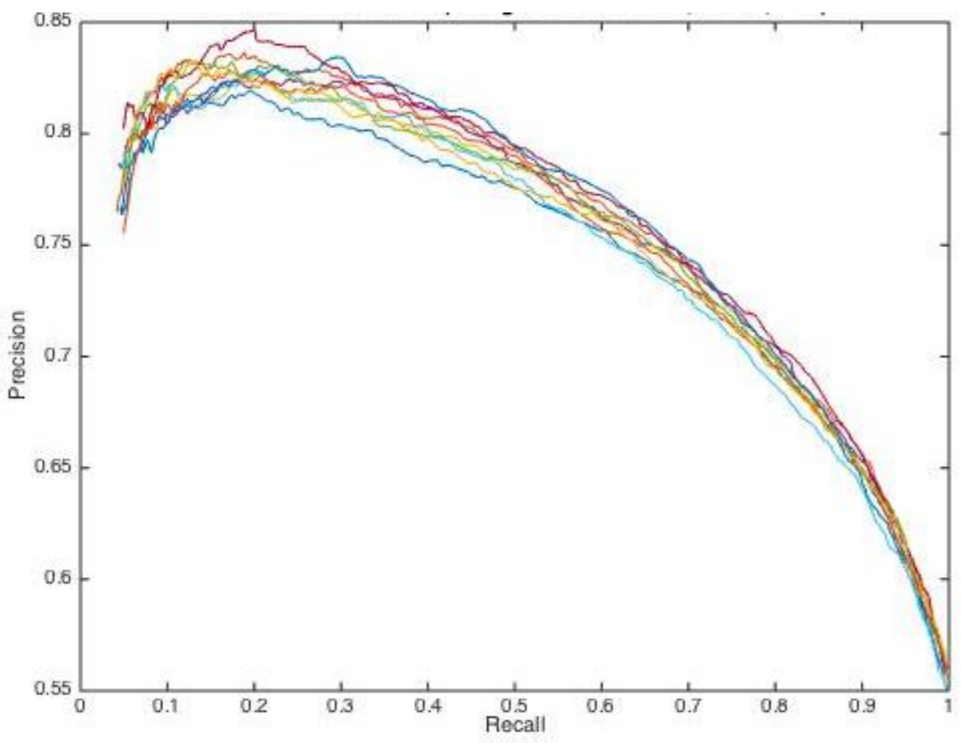

Figure 3.1 Precision vs Recall Curve (k = 10)

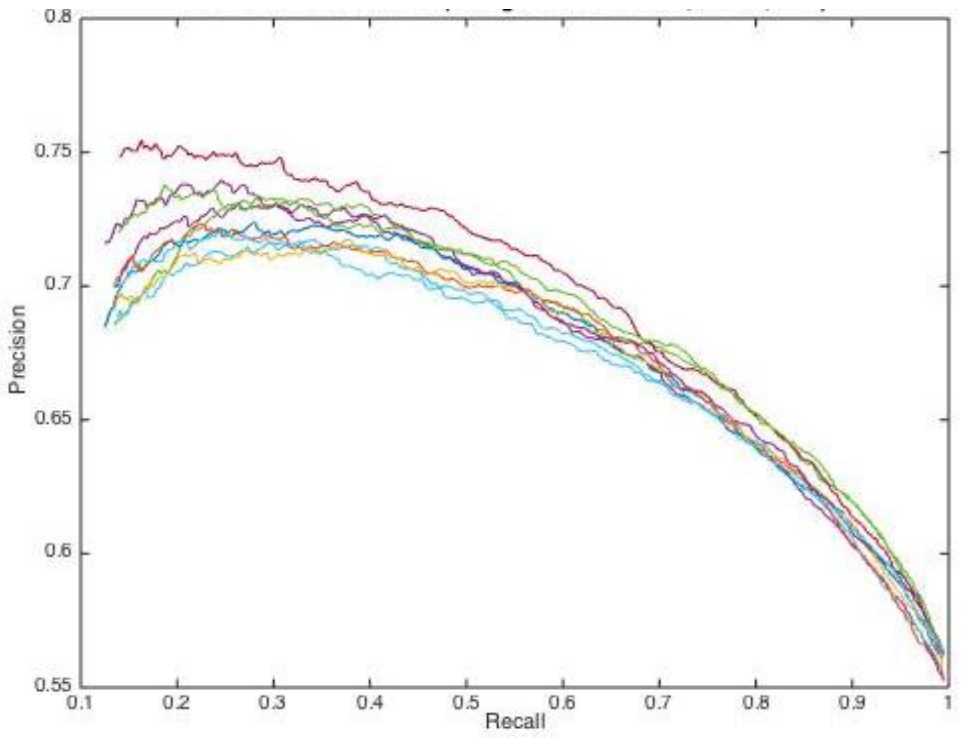

Figure 3.2 Precision vs Recall Curve (k = 50)

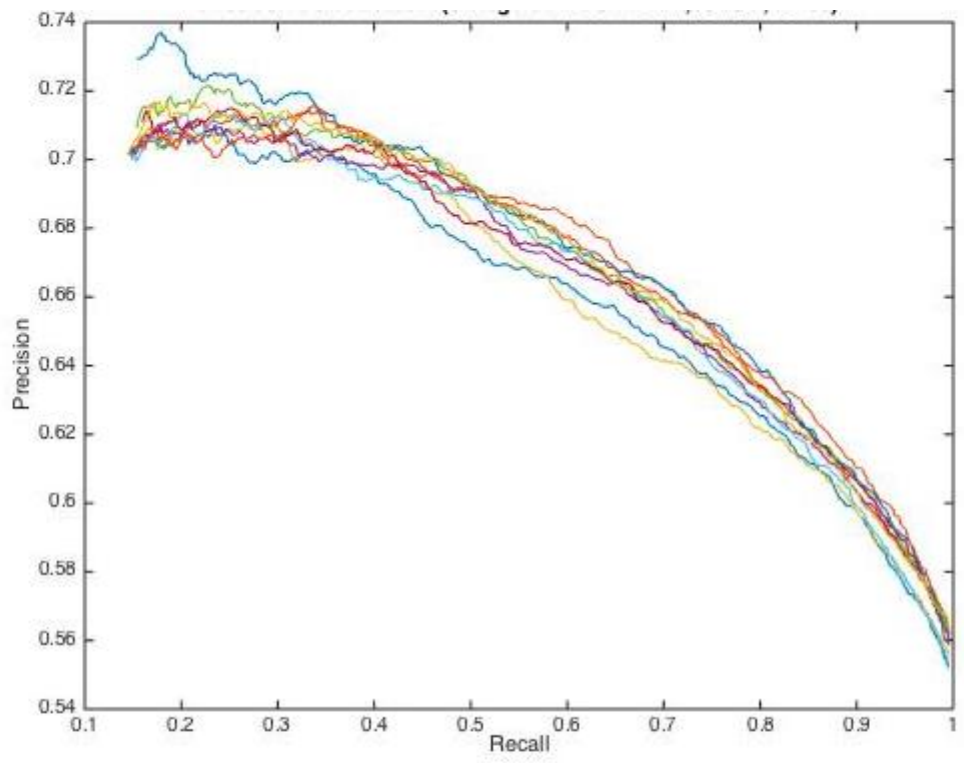

Figure 3.3 Precision vs Recall Curve (k = 100)

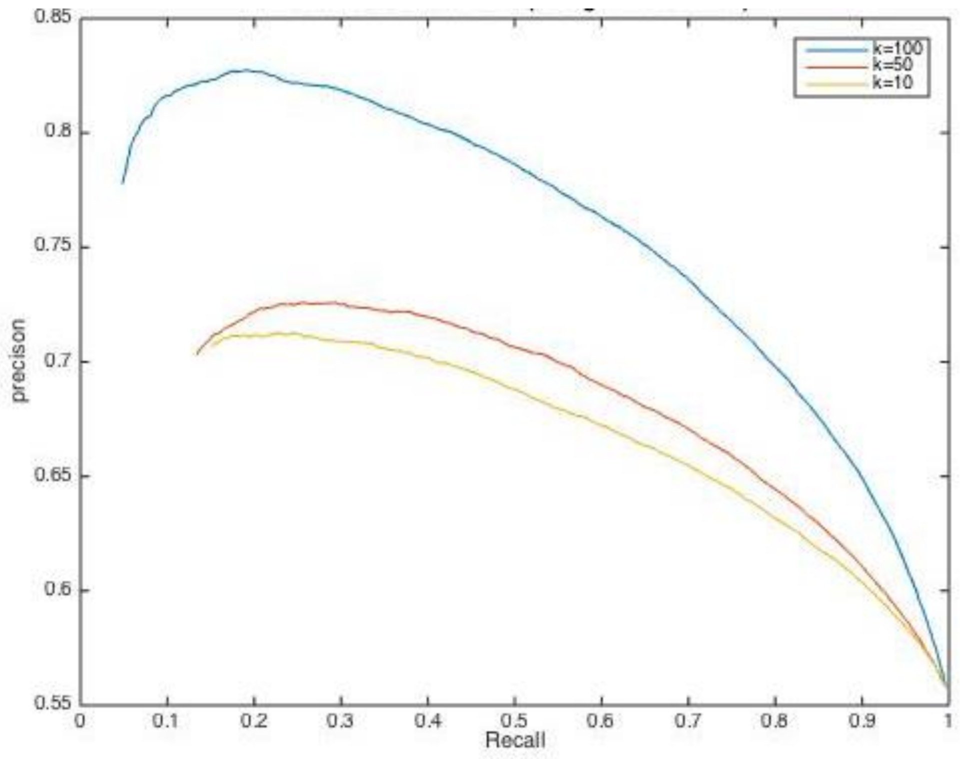

Figure 3.4 Precision vs Recall Average Curve

The curves above demonstrate that the precision decreases with the increase of recall. As k increases, the precision becomes larger with the same value of recall.

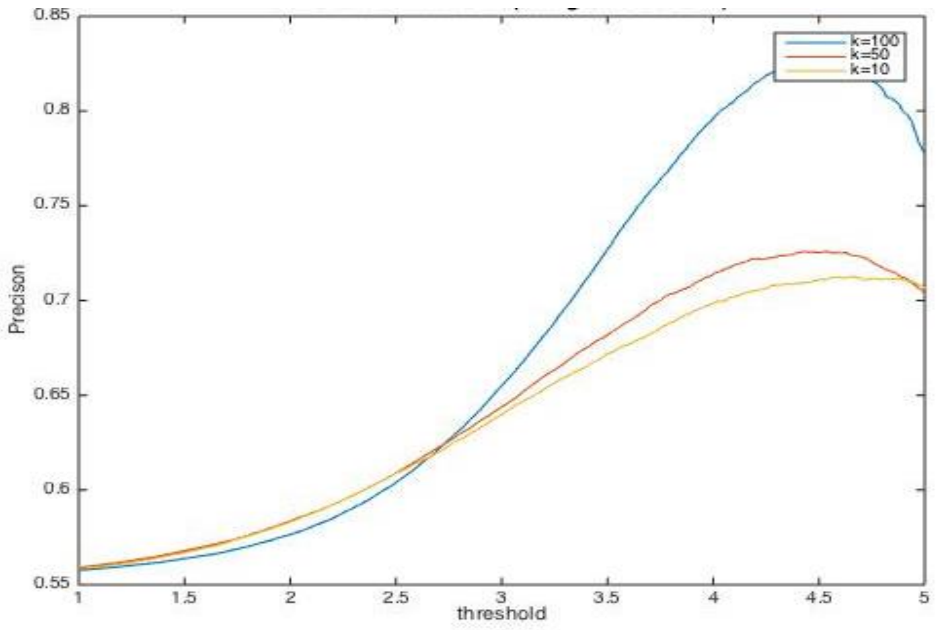

Figure 3.5 Precision vs Threshold Curve

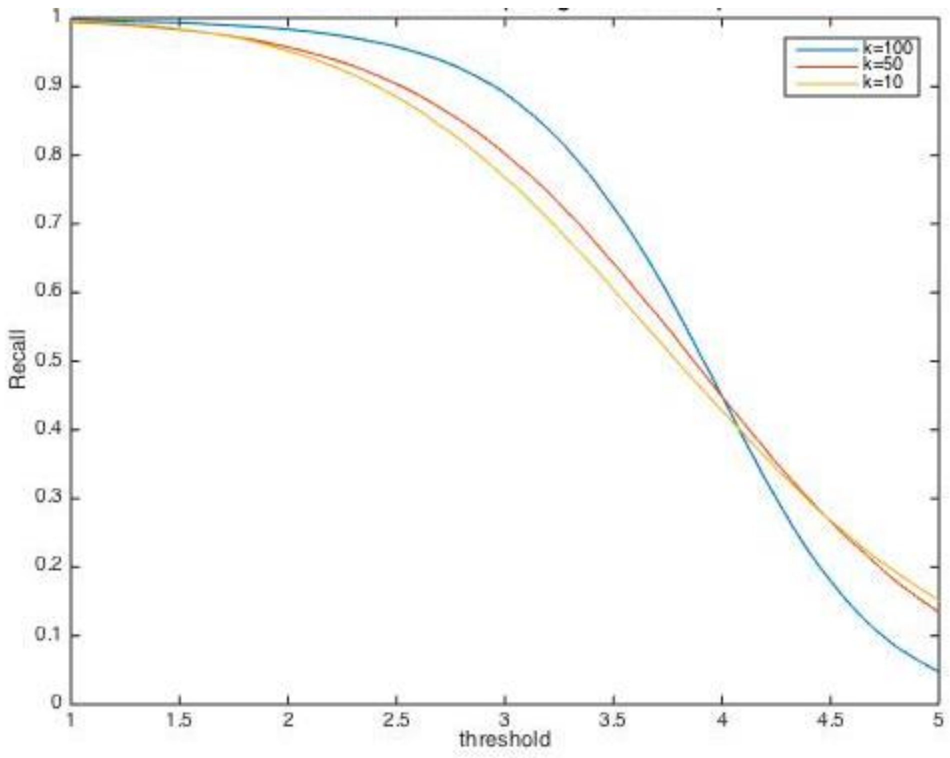

Figure 3.6 Recall vs Threshold Curve

From the curves above, we can find out that the precision increases as the threshold increases, while the recall decreases as the threshold increases.

IV. Part 4: ALS With Regularization

In this part, we firstly swapped weight and rating matrices, which means applying binary values in rating matrix and using rating values as weights for known data points. With the same factorization algorithm as in part 1, squared errors are shown in the following table.

Table 4.1 Squared Error and Residual (Swapped weight and rating matrices)

K	10	50	100
Sq.error	0.2774	1.6212	3.3294
Final Residual	1.4392	3.3317	4.8658

Compared with the squared error and residual of the original weight and rating matrices below,

Table 1.1 Squared Error and Residual (Original weight and rating matrices)

K	10	50	100
Sq.error	2.4621e+04	8.8567e+03	2.7648e+03
Final Residual	232.6866	138.9816	78.6874

We can see that, the final residual, squared error become extremely smaller than the original results, after making R a {0,1} matrix and W a rating matrix. However, we found that in this case, with the increasing of K, the error becomes larger. Therefore, we can see that making R a {0,1} matrix is a very good choice in building this recommendation system and we will use the same idea in the next part.

In order to avoid overfitting, we introduced a regularization term λ into the cost function of ALS. The cost function is changed to the following equation:

$$\min \sum_{i=1}^{m} \sum_{j=1}^{n} w_{ij} (r_{ij} - UV_{ij})^2 + \lambda \left(\sum_{i=1}^{m} \sum_{j=1}^{k} u_{ij}^2 + \sum_{i=1}^{k} \sum_{j=1}^{n} v_{ij}^2 \right) \quad \text{(equation 4-1)}$$

By applying the regularization, we can avoid singular solutions and make the computations more stable.

With the original matrices (before swapped), the squared errors and the final residuals for different values of λ and k are shown in the following table:

Tables 4.3 Squared Errors (Original Matrices with Different λ and k)

λ \ k	10	50	100
0.01	2.1758e+04	7.0071e+03	1.8066e+03
0.1	2.1736e+04	6.9961e+03	1.8139e+03
1	2.2386e+04	7.8319e+03	2.6975e+03

Tables 4.4 Final Residual (Original Matrices with Different λ and k)

λ \ k	10	50	100
0.01	218.7048	123.5582	63.7600
0.1	218.2897	123.3819	64.0543
1	219.0051	128.5529	75.8882

With the swapped matrices, the squared errors and the final residuals for different values of λ and k are shown in the following table:

Tables 4.5 Squared Errors (Swapped Matrices with Different λ and k)

λ \ k	10	50	100
0.01	0.2676	1.2846	2.8372
0.1	1.8079	2.3465	3.4786
1	69.2871	66.6076	63.3023

Tables 4.6 Final Residual (Swapped Matrices with Different λ and k)

λ \ k	10	50	100
0.01	1.1239	2.7126	4.2652
0.1	2.4312	3.1564	4.3141
1	15.1833	14.8087	14.3323

From the tables above, we can also see that both the squared errors and the final residual reduce to pretty smaller values when we swap the weight and the rating matrices, after introducing the regularization term λ into the cost function of ALS.

For the original matrices, when we fix a value of λ, with k increasing, squared error becomes smaller as well as the final residuals. When it comes to a fixed value of k, with λ increasing, there is an increase tendency for both squared error and the final residuals.

For the swapped matrices, when we fix a value of λ in 0.01 or 0.1, with k increasing, squared error becomes bigger as well as the final residuals. However when $\lambda = 1$, both the squared error and the final residuals decrease with the increase of k.When it comes to a fixed value of k, with λ increasing, squared error and the final residuals become bigger. And there in a much increment in the values of both squared error as well as the final residuals, when the value of λ turns into 1 from 0.1.

We also got the areas under corresponding precision and recall curves in the following table:

Table 4.7 Area under Precision and Recall curve (Swapped Matrices)

λ \ k	10	50	100
0.01	0.86121	0.8536	0.89478
0.1	0.8455	0.86598	0.89568
1	0.83354	0.85608	0.87187

From the tables above we can also see that, for a fixed value of λ, the area under curve has an increase tendency with the increasing of k. And for a fixed value of k, the area under curve has an increase tendency with the value of λ turns into 1 from 0.1.

The corresponding curves are shown as below.

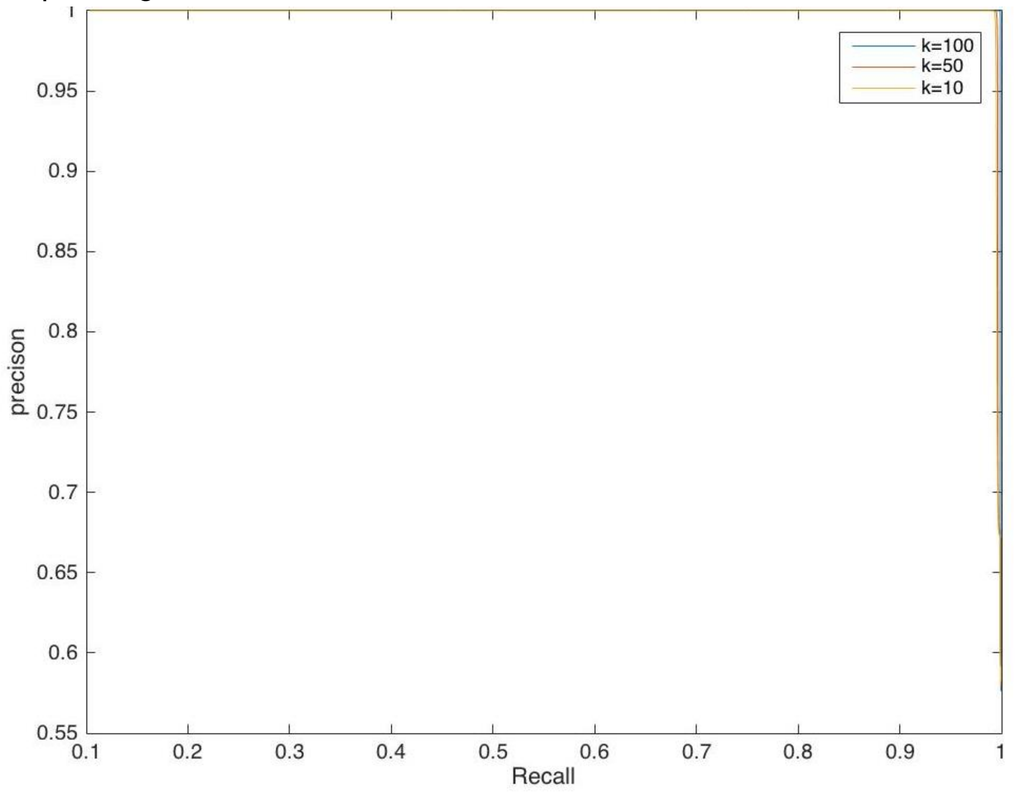

Figure 4.1 Precision vs Recall curve with $\lambda = 0.01$with different k

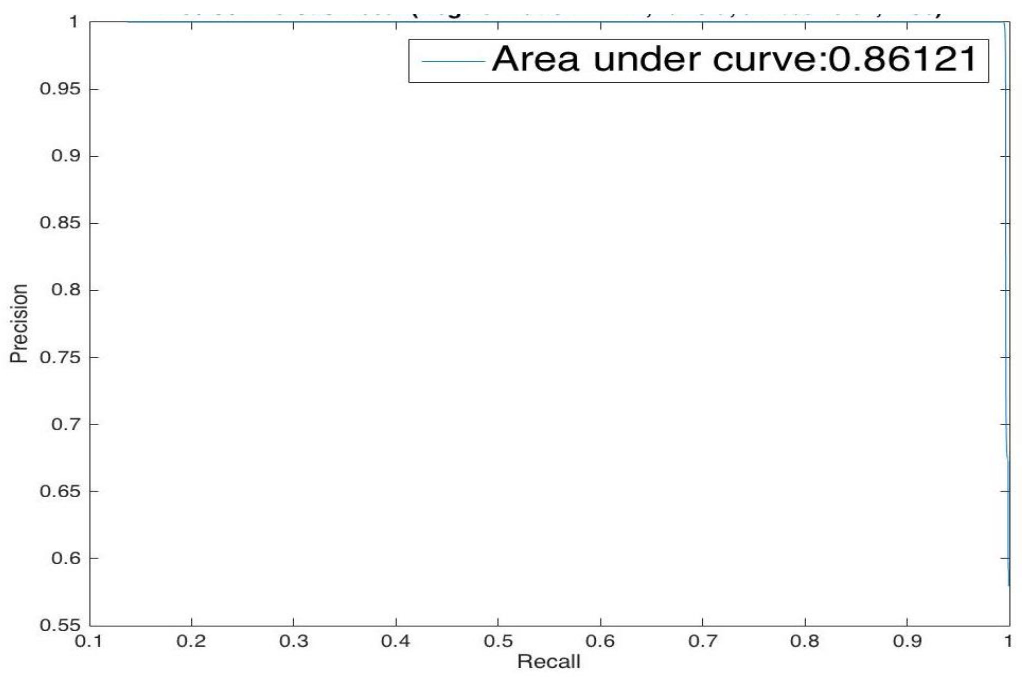

Figure 4.2 Area under precision recall curve with $\lambda = 0.01$, k = 10

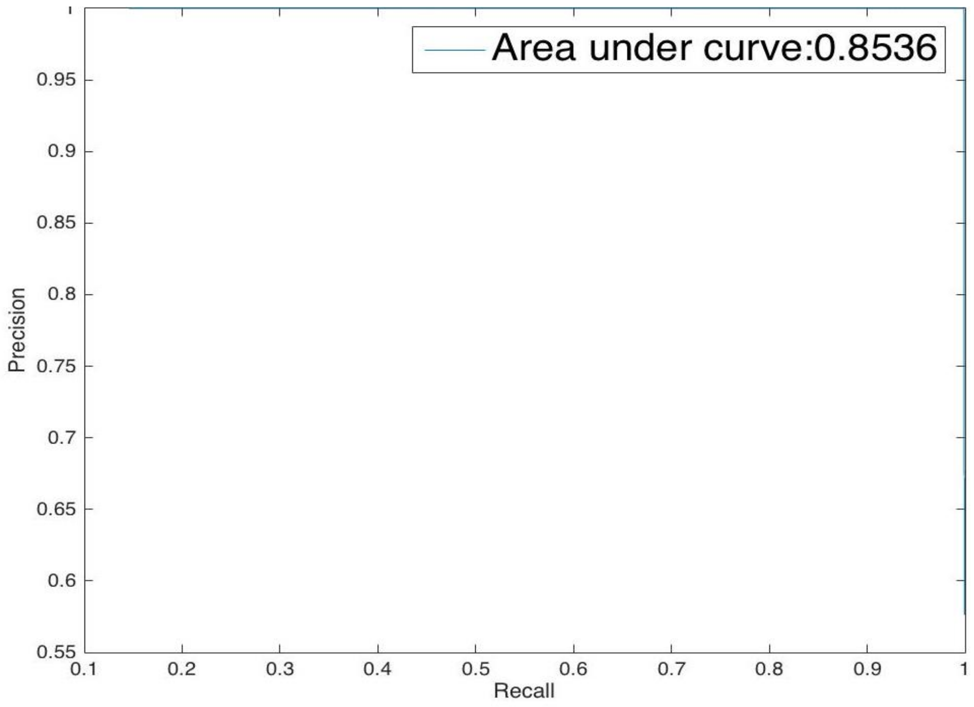

Figure 4.3 Area under precision recall curve with $\lambda = 0.01$, k = 50

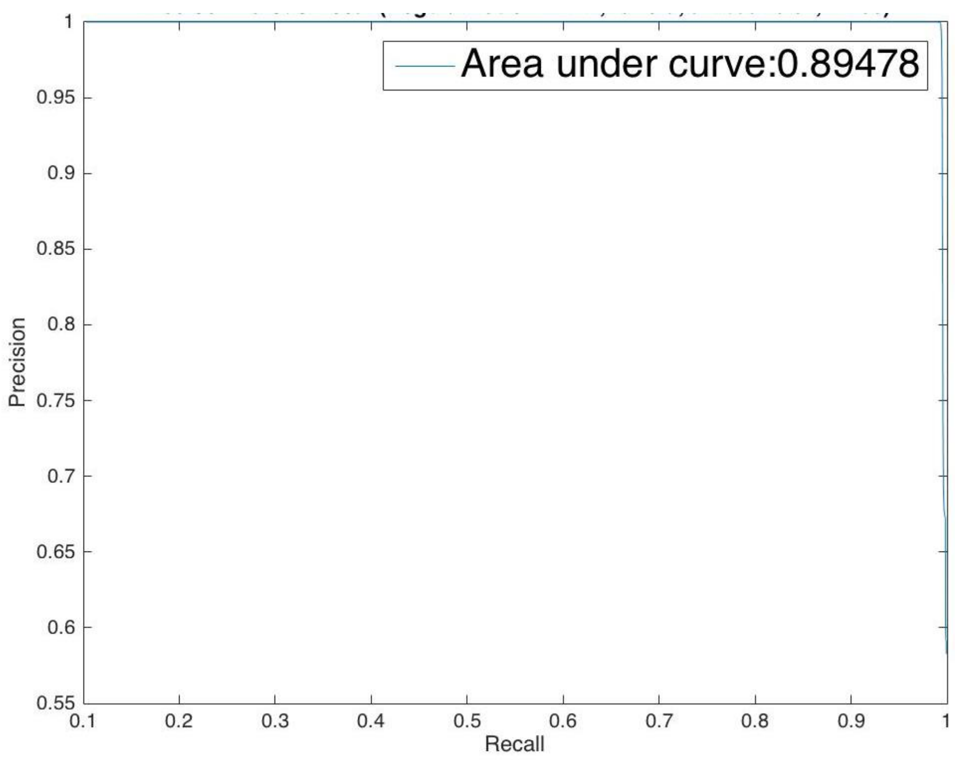

Figure 4.4 Area under precision recall curve with $\lambda = 0.01$, k = 100

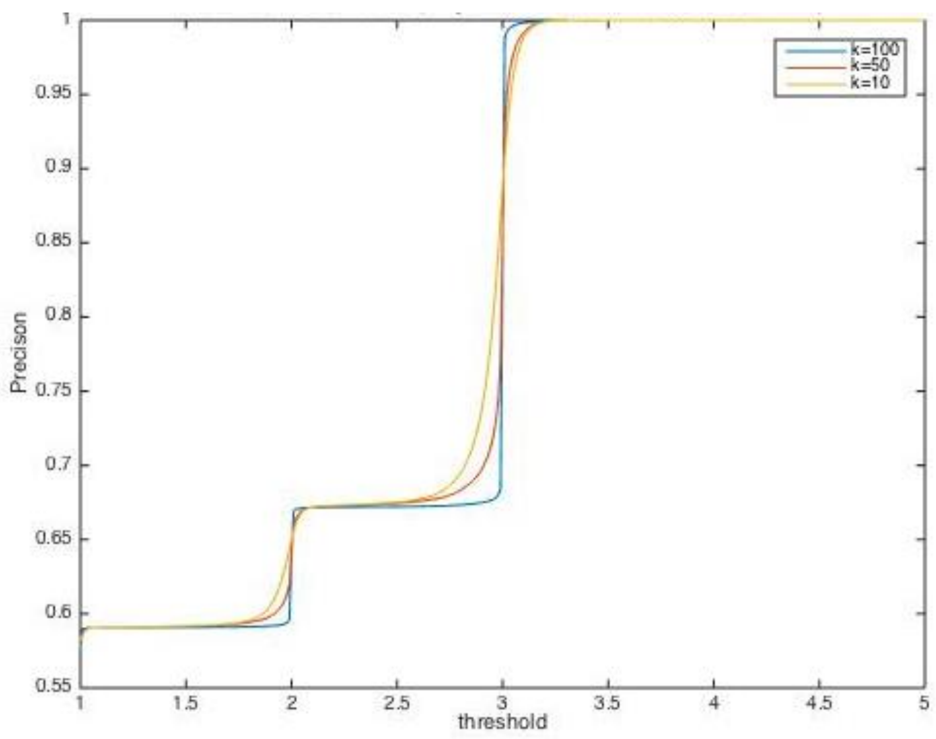

Figure 4.5 Precision vs Threshold with $\lambda = 0.01$

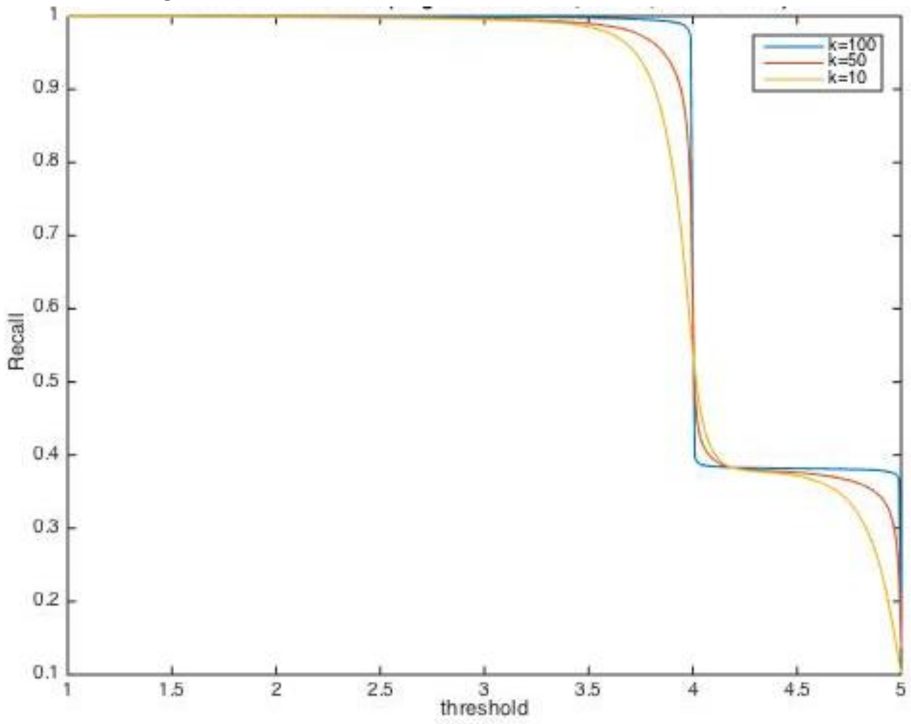

Figure 4.6 Recall vs Threshold with $\lambda = 0.01$

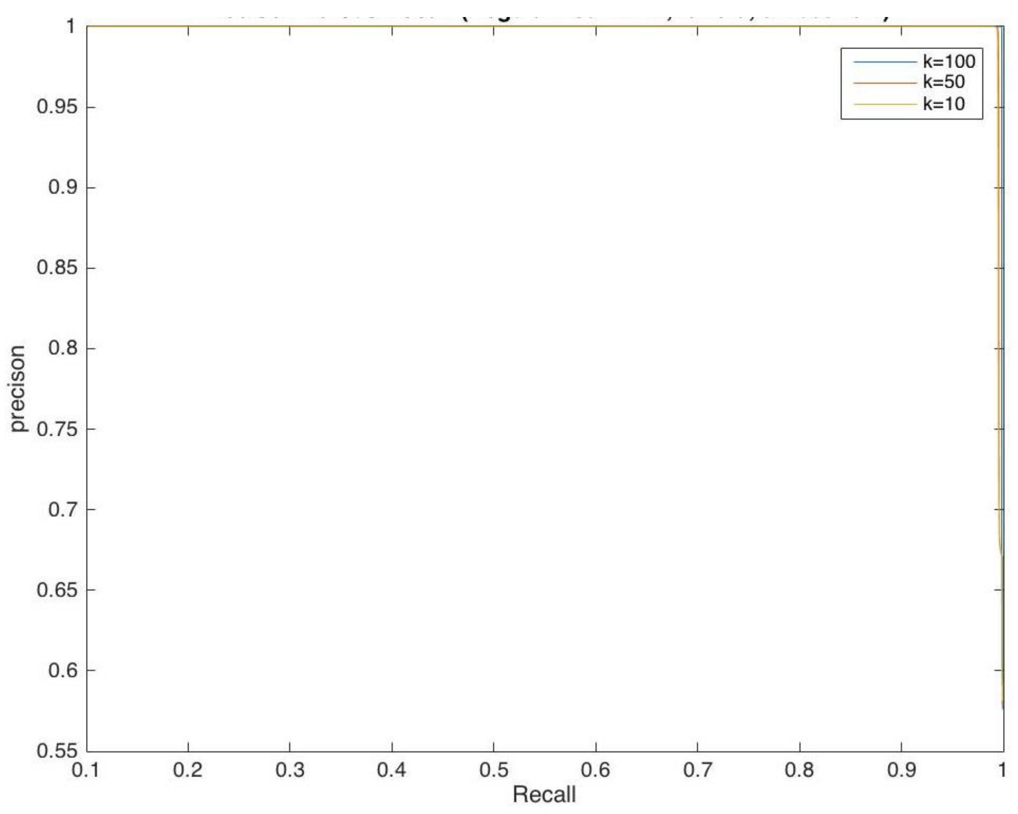

Figure 4.7 Precision vs Recall curve with $\lambda = 0.1$

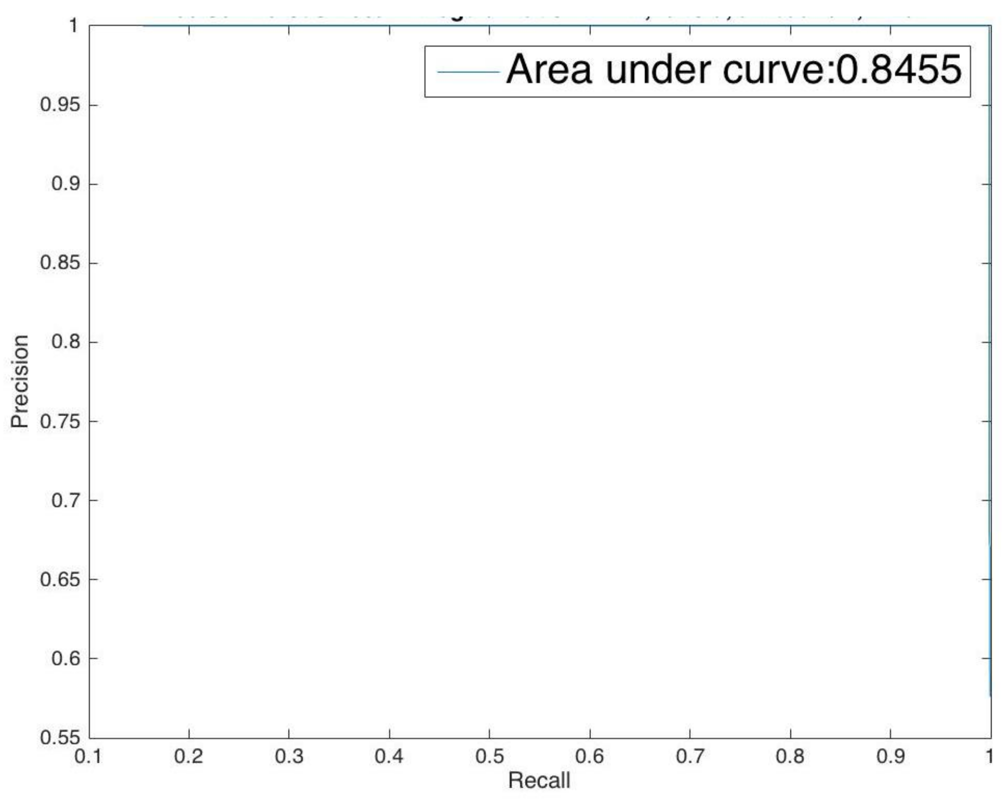

Figure 4.8 Area under precision recall curve with $\lambda = 0.1$, k = 10s

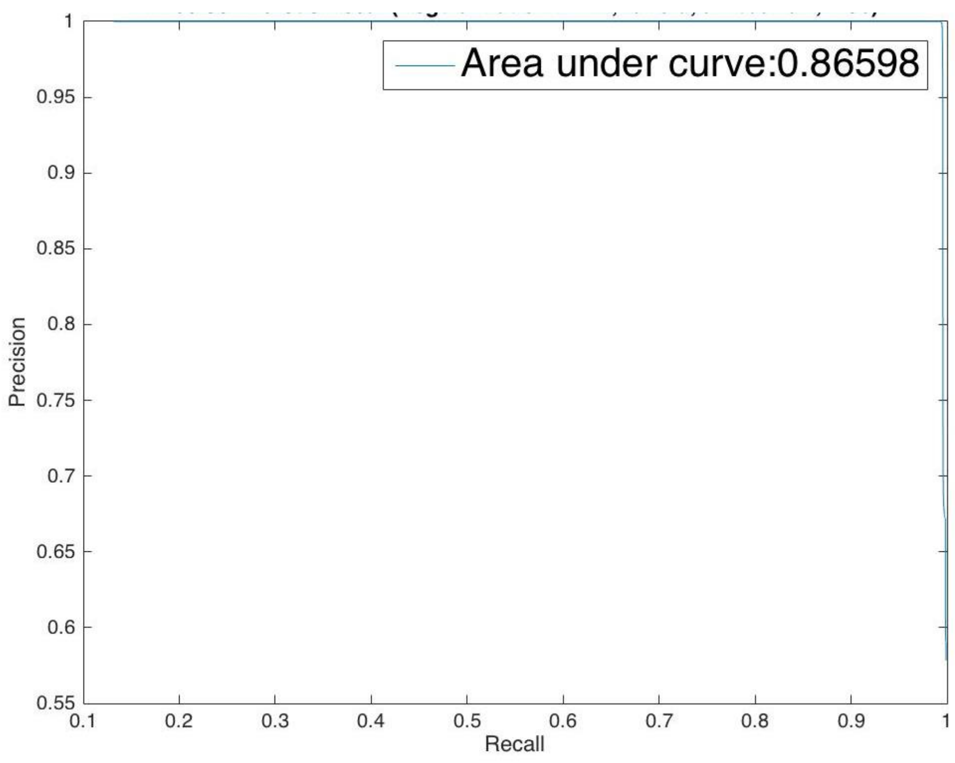

Figure 4.9 Area under precision recall curve with $\lambda = 0.1$, k = 50

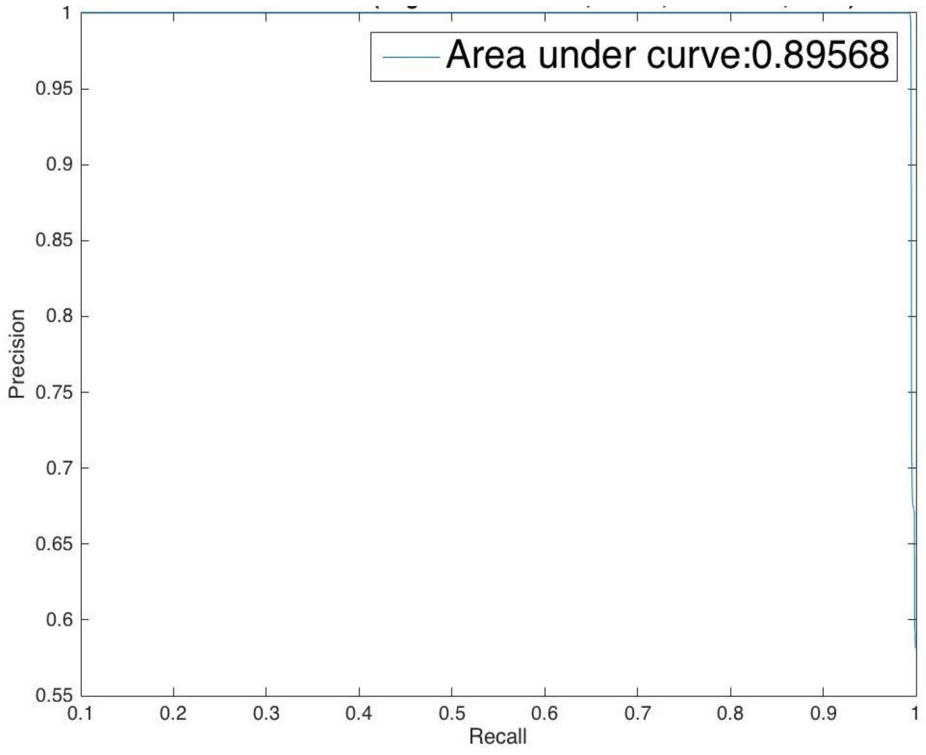

Figure 4.10 Area under precision recall curve with $\lambda = 0.1$, k = 100

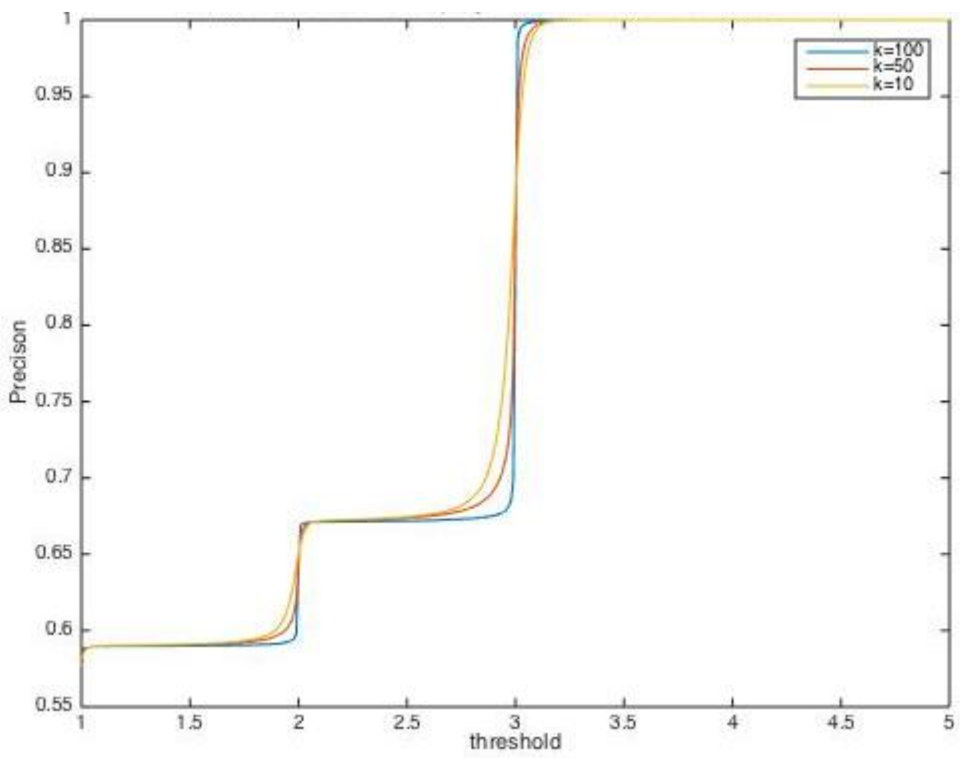

Figure 4.11 Precision vs Threshold with $\lambda = 0.1$

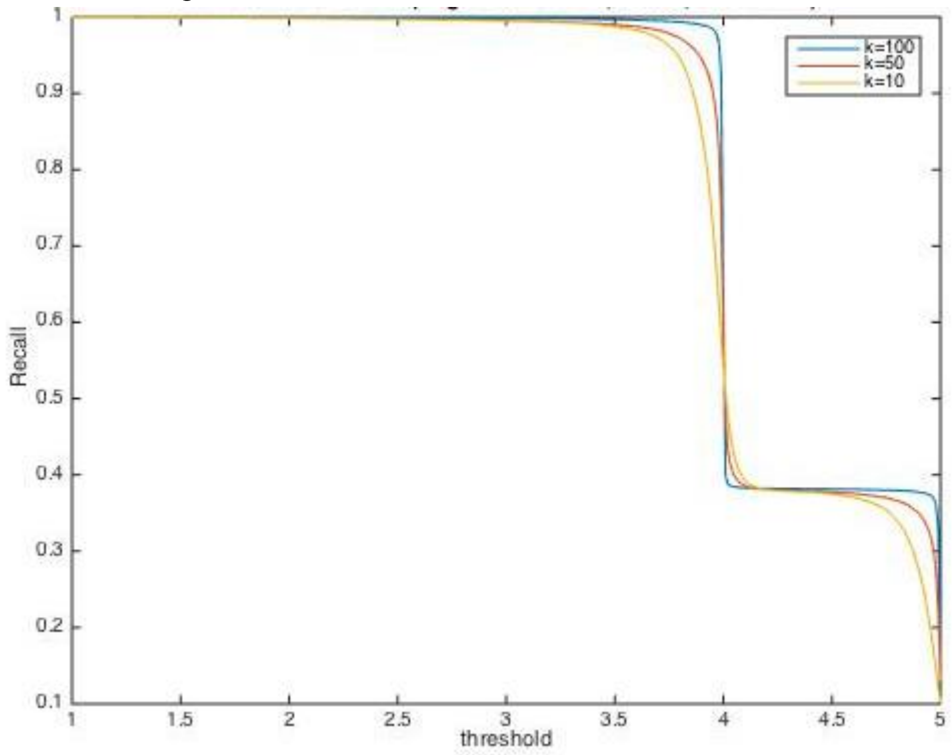

Figure 4.12 Recall vs Threshold with $\lambda = 0.1$

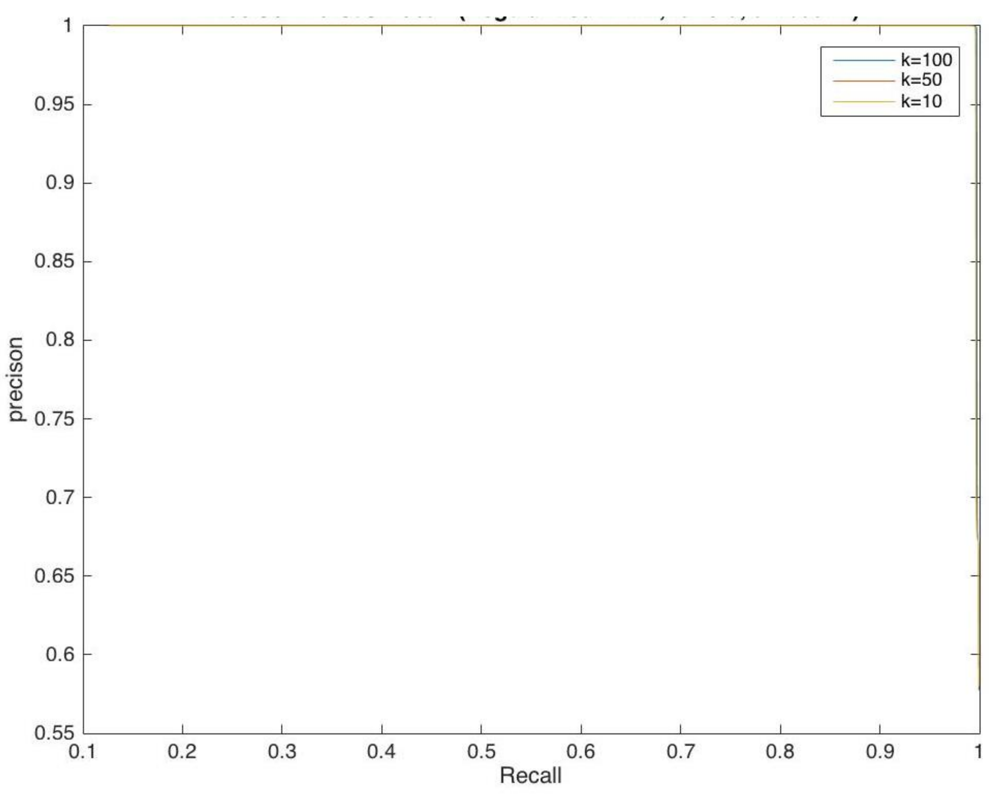

Figure 4.13 Precision versus Recall curve with $\lambda = 1$

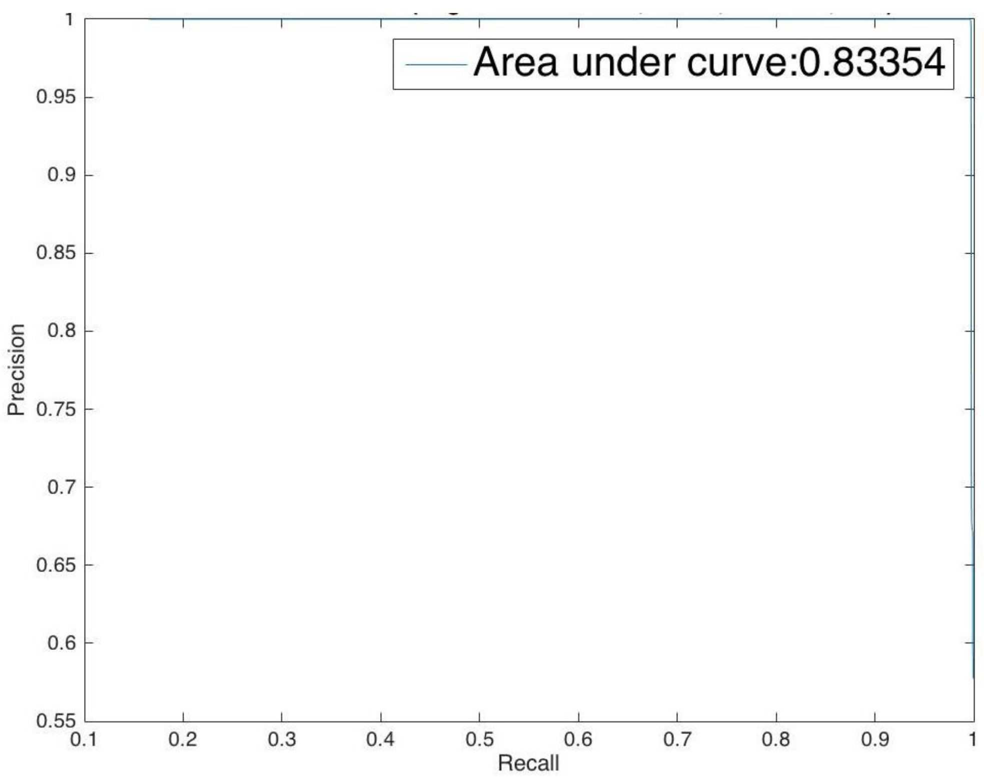

Figure 4.14 Area under precision recall curve with $\lambda = 1$, k = 10

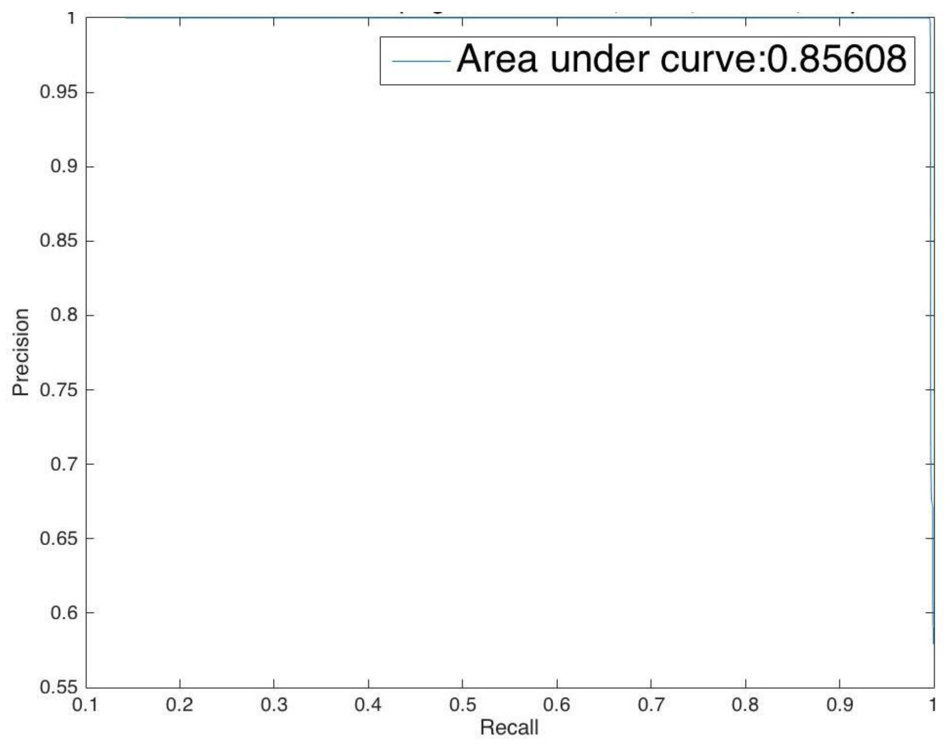

Figure 4.15 Area under precision recall curve with $\lambda = 1$, k = 50

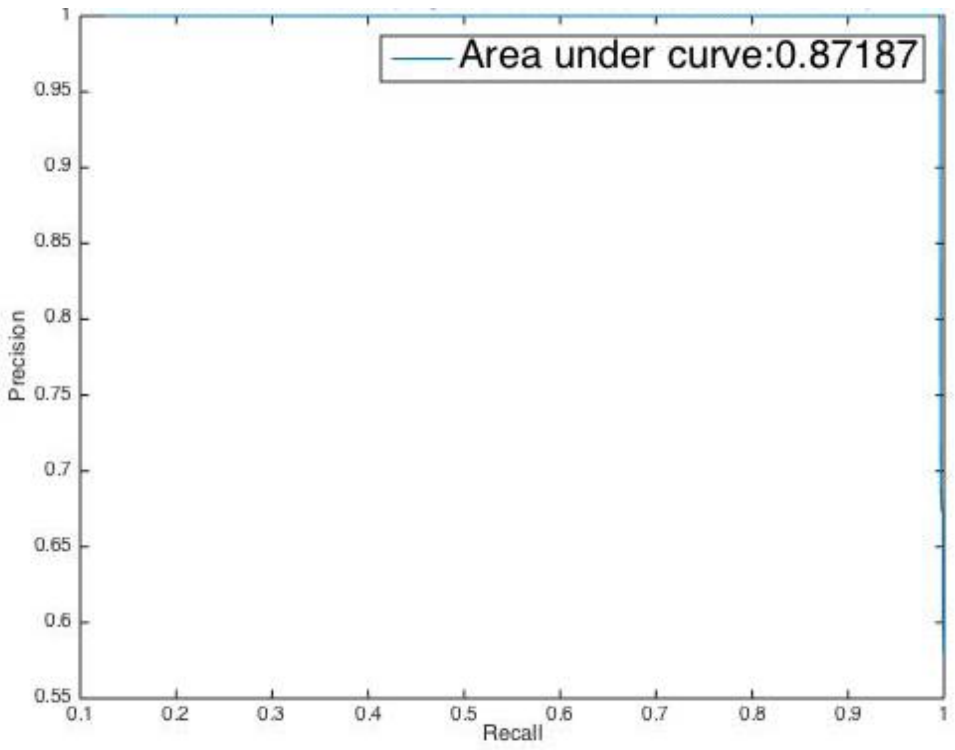

Figure 4.16 Area under precision recall curve with $\lambda = 1$, k = 100

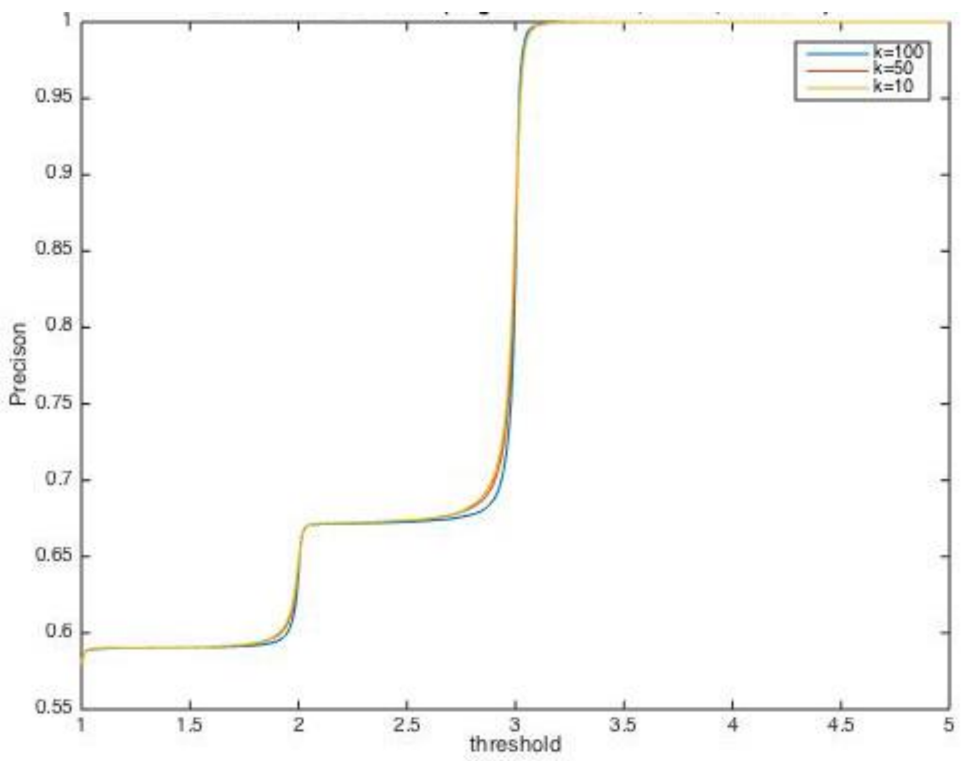

Figure 4.17 Precision vs Threshold with $\lambda = 1$

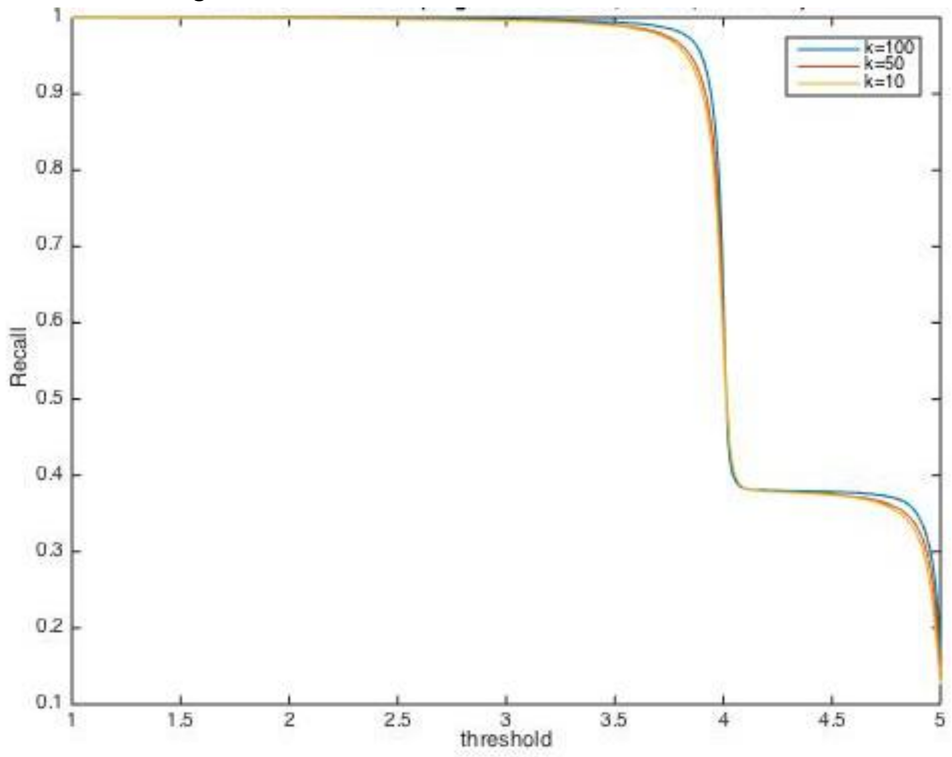

Figure 4.18 Recall vs Threshold with $\lambda = 1$

We can see from the above figures that for each λ, the lines of different k are nearly exact overlapped with each other. And we can see that after regularization, the precision, recall increases and the overall results of our algorithm improve are also improved as a result. From the Precision vs Threshold figure, we noticed that precision becomes larger when threshold increases. When threshold is set to be larger than 3, the precision is very close to 1. So when we set a higher threshold, we are able to give a more precise analysis of whether a user actually likes a movie. And there is a gap between each integer number of threshold which is reasonable because the samples only contain integer ratings. Besides, from the Recall vs Threshold figure we can see that recall decreases with the increment of K. When k is between 1 to 3, the recall is close to 1.

V. Part 5: Movie Recommendation

In this part, we have built a movie recommendation system by applying the same algorithms as before. Binary values are used in rating matrix, while rating values are used as weights for known data points. The number of recommendations is set by parameter L. We used the ALS algorithm with regularization as in part 4. 10-fold cross-validation method is applied for testing.

Same as in part 3, we set the rating threshold as 3 to predict whether the user like the movie or not.

In order to evaluate the system, we defined the related terms: true positive is the number of points that the system suggests which users actually like; false positive is the number of points that the system suggests which users actually do not like; false negative is the number of points that the system does not suggest which users actually like; true negative is the number of points that the system does not suggest which users actually do not like. The relationships can be indicated in the following table:

Table 5.1 Relationship Between Prediction and Reality

Like/Recommend or Not	True Positive	False Positive	False Negative	True Negative
Prediction	Yes	Yes	No	No
Reality	Yes	No	Yes	No

Precision, hit rate and false-alarm rate are computed as follow:

$$\text{Precision} = \text{True Positive} / (\text{True Positive} + \text{False Positive})$$
$$\text{Hit Rate} = \text{True Positive} / (\text{True Positive} + \text{False Negative})$$
$$\text{False} - \text{alarm Rate} = \text{False Positive} / (\text{False Positive} + \text{True Negative})$$

When L equals to 5, we would like to see the precision, hit-rate and false-alarm rate of our recommendation algorithm. After testing with different sets of Lambda and K, we choose $\mathrm{Lamda} = 0.1$, $\mathrm{K} = 100$ in this part and made use of 10-fold cross-validation to test our algorithm.

The precision, hit-rate, false-alarm rate of the corresponding 10 sets are as the following table:

Table 5.2 Precision, Hit-Rate and False-Alarm Rate

Fold	Precision	Hit-Rate	False-Alarm Rate
1	0.8099	0.5615	0.1627
2	0.8131	0.5590	0.1607
3	0.8063	0.5715	0.1672
4	0.8008	0.5544	0.1719
5	0.8139	0.5641	0.1609
6	0.8178	0.5669	0.1591
7	0.8054	0.5612	0.1664
8	0.8100	0.5639	0.1634
9	0.8174	0.5653	0.1555
10	0.8016	0.5574	0.1727

Then we calculated the average value as follow:

$$\mathrm{Average\ Precision} = 0.8097$$
$$\mathrm{Average\ Hit-rate} = 0.5626$$
$$\mathrm{Average\ false-alarm\ rate} = 0.1641$$

We can see from the above data that the average precision of our recommendation system is 0.8097 which is quite satisfying with the corresponding L. Also note that we set $\mathrm{threshold} = 3$ in this part and get the corresponding results. We can easily see that if we choose different threshold (say from 1 to 5), we will get different precision (the precision will decrease with the increment of L). And the average hit-rate is 0.5626, the average false-alarm rate is 0.1641. According to the definition of these two concepts, when L is set to be 5, we only recommend the top 5 movies to each user. So the fraction of test movies liked by the users are suggested by our algorithm will be affected by the number of L. Also, the fraction of test movies not liked by the users but suggested by our algorithm is also concerned with L. Next, we will plot the figures of Hit-Rate, False-Alarm Rate vs L and Hit-Rate vs False-Alarm Rate.

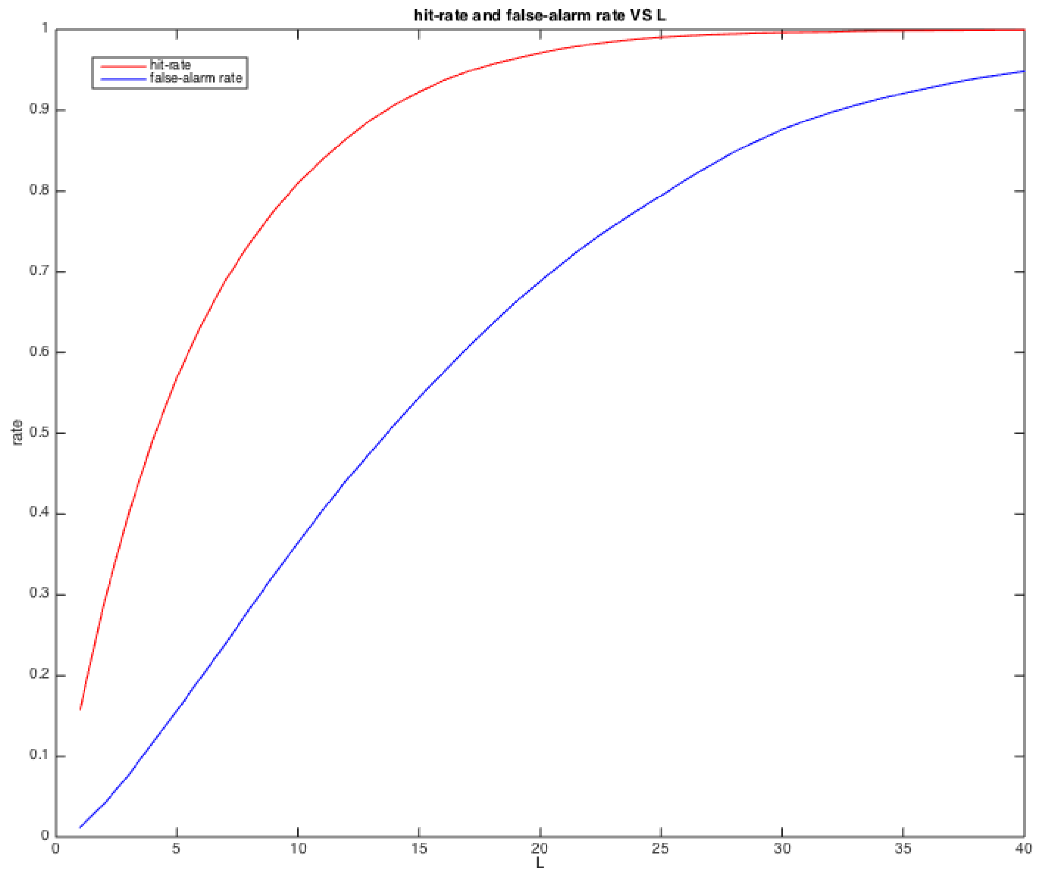

Figure 5.1 hit-rate and false-alarm rate vs L

We set L from 1 to 40 and while increasing L, we measure the hit-rate and false-alarm rate and plot the above figure. From the above figure, we can see that with the increasing of L, the hit-rate and false-alarm will increase as well. And when L is larger than 20, the hit-rate will be very close to 1. The result is pretty straightforward because with the increasing of L, we recommend more movies to each user and the number of true-positive and false-positive will then increase as a result. So we can get the two lines in the above figure.

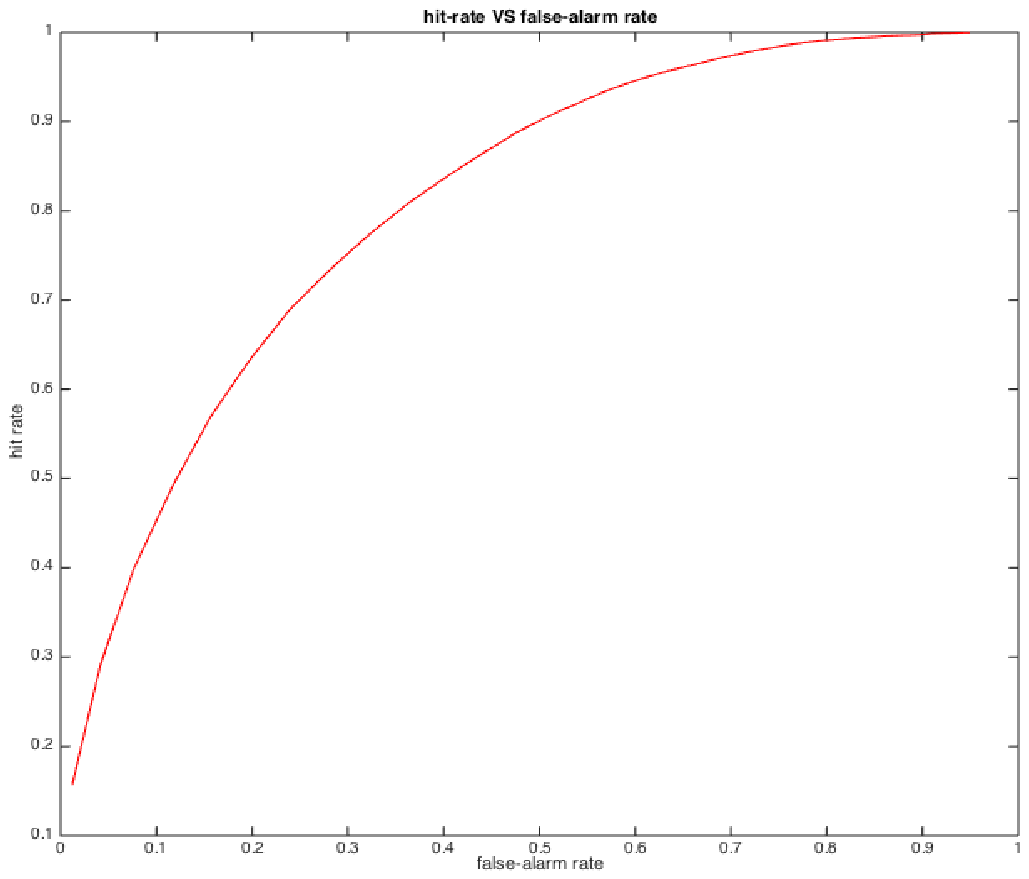

Figure 5.2 hit-rate vs false-alarm rate

The above figure displays a similar that with different setting of L, the hit-rate ranges from 0.15 to nearly 1 and the false-alarm rate ranges from 0 to 0.9. In the future work, we may choose different threshold and build a more considerable recommendation system.

VI. Conclusion

In this project, we have built a recommendation system to generate the inferences for missing data points in a user preference matrix. We applied Alternating Least Squares algorithm and modified it by adding a regularization factor. Also, we learned about using Non-negative Matrix Factorization to do matrix factorization and reconstruction. The metrics to measure the validity of our techniques used here is squared error which is quite simple but effective. We used cross-validation and test the quality of our recommendation system. We have also analyzed the threshold tradeoff between recall and precision. Eventually, we implemented a system that could recommend movies to users with a relatively high hit rate.