EE239AS - Project Report 1
Collaborative Filtering
Winter 2015

Ayush Minocha (104410979)
Nitish Mehta (404434455)

February 4, 2015

Collaborative Filtering is a method of making automatic predictions about the interests of a user by collecting taste information from many users. In this project, we are able to predict what a user will most probably like other than the movies he has rated.

1 Part 1

As the preliminary step, we constructed a matrix R denoted by users on the rows and movies on columns by extracting the userId, movie and rating columns of the given dataset. Each matrix location $R(i, j)$ represents the rating that a user i has given to movie j. The values that are missing in the matrix represent all those movies that a particular user has not rated yet.

In this part we try to decompose the R matrix into two matrices U and V such that:

$$R_{m\times n} \approx U_{m\times k} \times V_{k\times n} \tag{1}$$

Where,
Matrix U represents the movie features that each user likes
Matrix V represents the features of each movie
k represents the no of features chosen to represent a movie which takes the values: 10, 50, 100.

To effectively get the factorization as close to accurate as possible we have to minimize the squared error given by following equation:

$$min \sum_{\text{known } i,j} (r_{i,j} - (UV)_{i,j})^2 \tag{2}$$

We want to minimize the squared error cost function by using only the known data points in the R matrix. This can be achieved by creating a weight matrix $W_{m\times n}$ having 1 at known data points and 0 at unknown data points. The above equation now becomes:

$$min \sum_{i=1}^{m} \sum_{j=1}^{n} w_{i,j}(r_{i,j} - (UV)_{i,j})^2 \tag{3}$$

We implemented the above equation using "wnmfrule" from the matrix factorization toolbox. The values for Least Squared Error (LSE) for k = 10, 50, 100 are as follows:

k	Total LSE
10	232.8566
50	139.5878
100	79.1659

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

2 Part 2

For this part, we interchanged the weights and rating matrices with each other. The weight matrix now contains the rating values as weights for known data points and the rest as 0. R is converted into a 0-1 matrix just as W in the last part.

k	Residual Error	Total LSE
10	1.0557	0.5912
50	3.0697	1.6616
100	4.9126	2.5985

Table 2: Least Squared Errors (LSE) and Residual Error for different values of k

Here we have two different values of errors. This is because while calculating the error, the toolbox uses a formula which is not applicable when we swap the Rating and Weight matrices. So the formula used by the tool box is given by the equation:

$$min \sum_{i=1}^{m} \sum_{j=1}^{n} (w_{i,j}(r_{i,j} - (UV)_{i,j}))^2 \tag{4}$$

This equation is valid when our weight matrix is a 0-1 matrix, so the $w_{i,j}$ takes a value of 0 or 1 and thus it does not affect the square of the error. But in this case, the weight matrix is replaced by the rating matrix, and thus the residual error does not make much sense. For calculating the least squared error given in the table above, we use the equation 3.

3 Part 3

To test the recommendation system that we have designed, we used 10-fold cross validation technique. We assigned an index from 1 to N to all the known data points. Then we created a random ordering of these numbers and split them into 10 equal parts. For each fold we used one of these 10 parts for testing and the rest for training. This ensures that each data point is tested once. We found the absolute error over the testing data by calculating the difference between $R_{predicted}$ and R_{actual}. For each fold we take an average over all the test samples, and for each 'k' we take and average over all the folds. Thus the final mean error is the average over all the samples, where each is taken as test exactly once.

3.1 10 Fold Cross-Validation for Part 1

We perform a 10 fold cross-validation for the first part, where we take the R matrix as the rating matrix and W as a 0-1 matrix. The results for the experiment are given in the table 3

	k		
Fold No.	10	50	100
1	2.2275	2.2973	2.5185
2	2.2408	2.3046	2.4967
3	2.2514	2.3236	2.5333
4	2.2522	2.3237	2.5276
5	2.2330	2.2893	2.5063
6	2.2235	2.3060	2.5013
7	2.2458	2.3233	2.5321
8	2.2535	2.3287	2.5338
9	2.2450	2.3309	2.5337
10	2.2257	2.2954	2.5181

Table 3: Least Squared Errors (LSE) for different values of k and for each fold

The average absolute error, lowest average error and the highest average error for each value of k is given in the table 4.

k	Average Absolute Error	Lowest Average Error	Highest Average Error
10	2.2399	2.2235	2.2535
50	2.3123	2.2954	2.3309
100	2.5201	2.4967	2.5338

Table 4: Average Absolute Error , Lowest Average Error and Highest Average Error for different values of k

3.2 10 Fold Cross-Validation for Part 2

We perform a 10 fold cross-validation for the second part similar to the first, where we take the R matrix as the 0-1 matrix and the W matrix as the rating matrix. The results for the experiment are given in the table 5.

10 fold cross validation is a good measure, because we are considering all the samples as a test case exactly once, and also we train the model with 10 different sets of samples, thus eliminating any possibility of special cases.

	k		
Fold No.	10	50	100
1	0.6375	0.6603	0.7220
2	0.6368	0.6548	0.7209
3	0.6376	0.6599	0.7252
4	0.6403	0.6622	0.7230
5	0.6385	0.6574	0.7220
6	0.6369	0.6594	0.7216
7	0.6400	0.6577	0.7208
8	0.6388	0.6601	0.7245
9	0.6385	0.6647	0.7220
10	0.6348	0.6593	0.7237

Table 5: Least Squared Errors (LSE) for different values of k and for each fold

The average absolute error, lowest average error and the highest average error for each value of k is given in the table 6.

k	Average Absolute Error	Lowest Average Error	Highest Average Error
10	0.63797	0.6348	0.6403
50	0.65958	0.6548	0.6647
100	0.72257	0.7208	0.7252

Table 6: Average Absolute Error , Lowest Average Error and Highest Average Error for different values of k

4 Part 4

For the 4^{th} part we need to find the precision and recall values by varying the threshold over a certain range. As a ground truth we assume that if a rating is above the value 4, then the user has liked the movie, else not liked. We calculate the precision and recall using:

$$\text{Precision} = \frac{\text{Predicted movies actually liked by the user}}{\text{Total movies predicted as liked}} \tag{5}$$

$$\text{Recall} = \frac{\text{Predicted movies actually liked by the user}}{\text{Total movies liked by the users}} \tag{6}$$

We performed 10 fold cross validations for each k , and ranged the thresholds from 0.2 to 4.8 at an interval of 0.2 (i.e. 24 thresholds). Thus for each fold we get 24 precision-recall values, and for each k we take an average over all the folds, leaving us with 24 precision-recall values for each k. Here the trend in the precision and recall values is as expected. We see that the precision increases with threshold, because the number of positive detections decrease with threshold, and the recall decreases with threshold. Precision is low for small threshold because for small thresholds more movies are detected as liked and for the same reason the recall is high, because all the actually liked movies are detected when threshold is low. Hence the trend can be explained.

	k		
Threshold	10	50	100
0.2	57.4086	57.4088	57.2814
0.4	59.248	58.9806	58.8513
0.6	60.8904	60.5395	60.2556
0.8	62.7325	62.0336	61.7578
1	64.457	63.5515	63.1708
1.2	66.1168	65.0953	64.5059
1.4	67.4585	66.5557	65.7555
1.6	68.9776	68.1898	67.0164
1.8	70.5635	69.6688	68.2455
2	72.0568	70.9156	69.2332
2.2	73.5455	72.2974	70.3799
2.4	75.0185	73.5785	71.4964
2.6	76.536	74.9047	72.5145
2.8	77.9556	76.2789	73.4141
3	79.4752	77.4444	74.2758
3.2	80.9455	78.8661	75.0105
3.4	82.4172	79.986	75.8408
3.6	83.4447	81.0465	76.4076
3.8	84.7753	81.8031	76.6348
4	86.0765	82.2284	76.8158
4.2	87.5269	82.6036	76.5752
4.4	87.9396	82.704	76.3295
4.6	89.5457	82.6538	76.1923
4.8	90.2002	82.4709	75.8966

Table 7: Precision Values for different values of k and for different values of threshold (values in %)

	k		
Threshold	10	50	100
0.2	94.404	91.546	82.5245
0.4	87.6974	85.2539	75.5327
0.6	80.4468	79.026	69.3329
0.8	73.2321	72.7516	64.014
1	66.2153	66.9115	58.9526
1.2	59.5803	61.3532	54.3564
1.4	53.0414	56.1328	50.1046
1.6	47.0482	51.2775	46.2761
1.8	41.6303	46.4976	42.5958
2	36.5914	41.9471	39.1368
2.2	31.8305	37.7353	35.9423
2.4	27.685	33.7941	32.915
2.6	23.9275	30.3099	30.1492
2.8	20.4484	27.0297	27.5012
3	17.5518	23.9061	25.1188
3.2	14.8617	21.1054	22.8798
3.4	12.5137	18.6818	20.9333
3.6	10.4562	16.381	19.0431
3.8	8.6117	14.2204	17.3905
4	7.0738	12.3089	15.7909
4.2	5.6653	10.5717	14.3443
4.4	4.5027	9.0134	12.9848
4.6	3.6122	7.6993	11.935
4.8	2.8117	6.5741	10.9417

Table 8: Recall Values for different values of k and for different values of threshold (values in %)

Using these values we made 3 plots: Precision vs Recall, Precision vs Threshold and Recall vs Threshold. The plots are given in Figure 1.

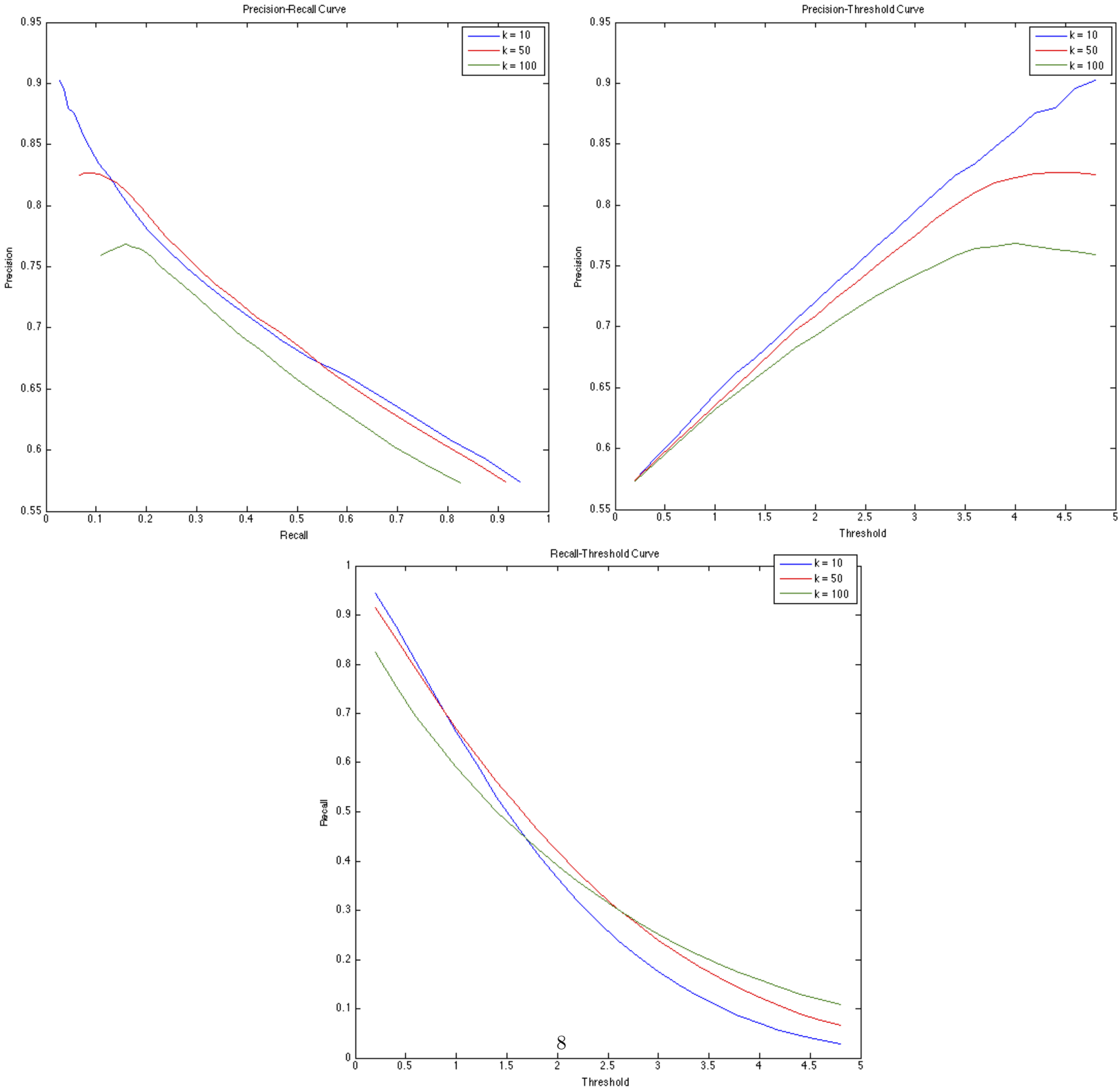


Figure 1: Plots for part 4

5 Part 5

In this system, if we have m users and n items, then we want to learn a matrix of factors which represent movies and users. In this problem, we have two unknown variables i.e. factor matrix for movies $Y \in R^{f \times n}$ and factor matrix for users $X \in R^{m \times f}$. We use an alternating least squares approach with regularization to first estimate Y using X and estimate X using Y. After enough number of iterations, we are aiming to reach a convergence point where X and Y are no longer changing or the change is quite small. We need regularization terms in order to avoid over-fitting the data. Ideally, regularization parameters need to be tuned using cross-validation for algorithm to perform better. The cost function with the regularization term λ is as follows:

$$min \sum_{i=1}^{m} \sum_{j=1}^{n} w_{ij}(r_{ij} - (UV)_{ij})^2 + \lambda(\sum_{i=1}^{m} \sum_{j=1}^{k} u_{ij}^2 + \sum_{i=1}^{k} \sum_{j=1}^{n} v_{ij}^2) \tag{7}$$

	λ		
k	0.01	0.1	1
10	235.7957	235.5736	236.6968
50	144.8724	145.6205	150.3508
100	88.1018	88.6675	98.0861

Table 9: Least Squared Errors (LSE) for different values of k and for different values of λ using the regularized function but R and W as mentioned in part 1

	λ		
k	0.01	0.1	1
10	1.3880	2.2406	8.8964
20	3.3690	3.3498	8.6409
50	4.3882	4.1594	8.3848

Table 10: Least Squared Errors (LSE) for different values of k and for different values of λ using the regularized cost but R and W as mentioned in part 2

The results for part 5 can be found in table 9 and table 10.

6 Part 6

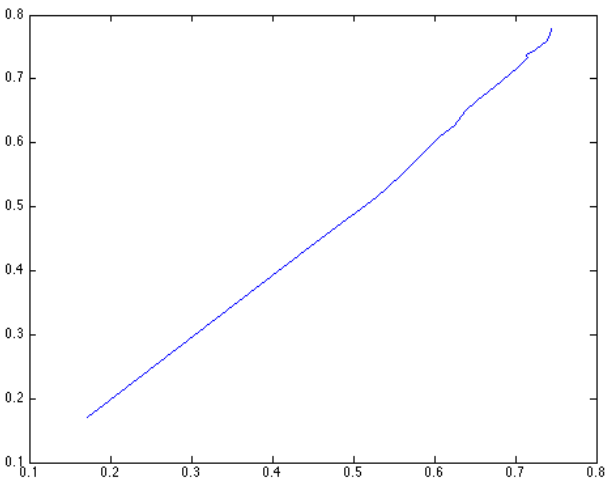
As done in Part 2, we have transformed R into a 0-1 matrix by placing 1 for available data points and 0 for missing data points. The W matrix contains rating which are used as weights. We then run the algorithm using the equation in Part 5 with the regularization term.

We first get a ground truth matrix for likes and dislikes of the movie based on threshold 3. The user likes a movie if he has rated a movie 4 or higher and disliked is he has rated it 3 or lower. For each value of lambda and each k we do a 10 fold cross validation. In each fold once we get the predicted matrix we use only the test values for the precision, hit rate and false rate calculation. Based on the values of the predicted matrix we sort and return the top L as the recommendations to the users and then calculate the precision using this. Similarly for hit-rate we find the number of movies liked by the user from the recommendations and take it as a fraction of the total number of movies liked by the user in the test set. Similarly for false alarm rate.

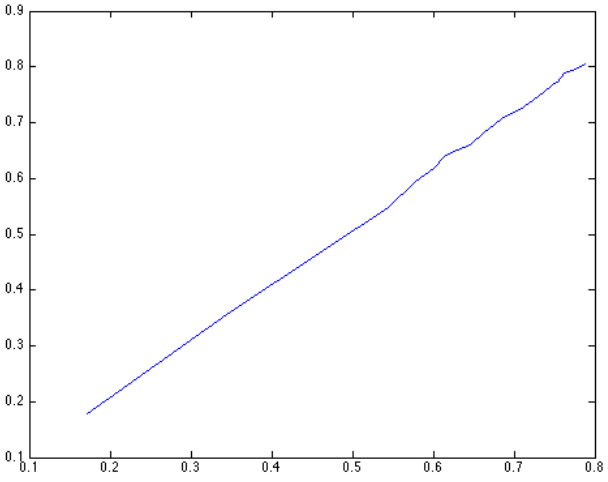
	λ		
k	0.01	0.1	1
10	56.83	57.2732	58.128
50	57.8862	57.8353	58.5543
100	58.6858	58.7218	59.0399

Table 11: Precision values for k=10,50,100 and λ=0.01,0.1,1 (all values in %)

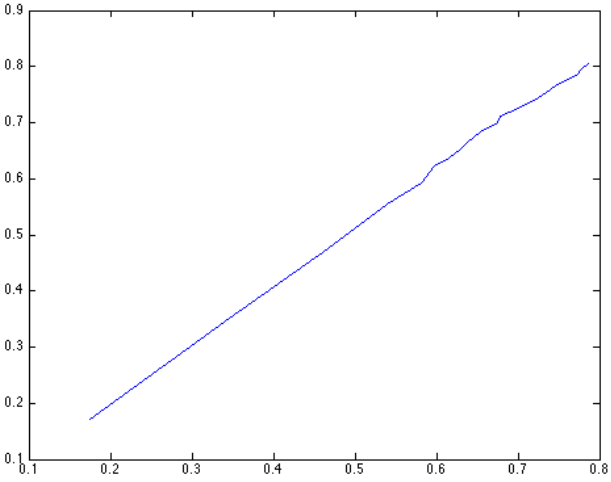
For each user we calculate the hit rate and the false alarm rate, and for each value of L we take an average over all the users, thus obtaining one hit rate and false-alarm rate for each L, corresponding to one value of k and one value of λ. We have k=10,50,100 and λ=0.01,0.1,1 and so we get nine combinations and thus we plot nine graphs for hit rate vs false alarm rate as shown in the Figures 2 and 3. We can see that as we increase L the hit rate increases as we are eventually covering all the test movies and hence all the liked movies are covered, similarly the false alarm also increases.



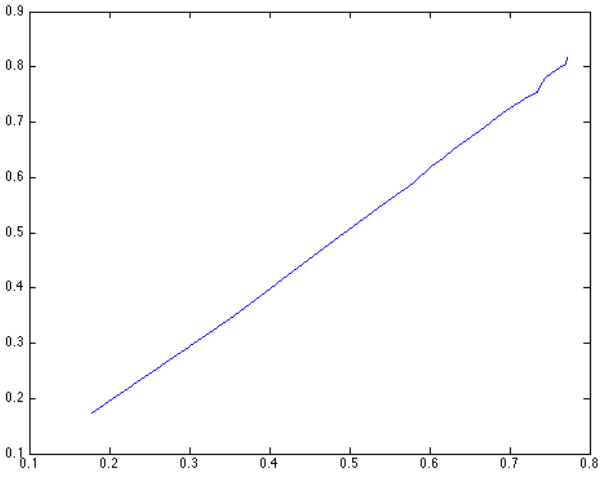
K=10 Lambda=0.01



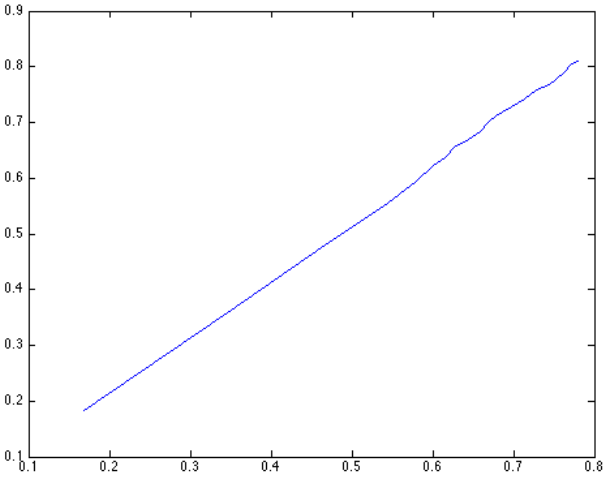
K=10 Lambda=0.1



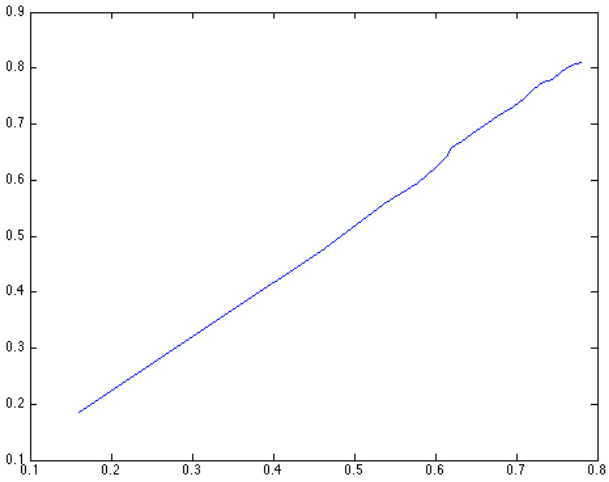
K=10 Lambda=1



K=50 Lambda=0.01

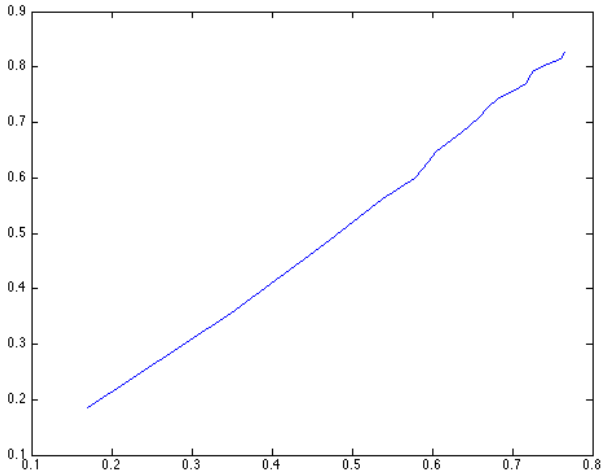


K=50 Lambda=0.1

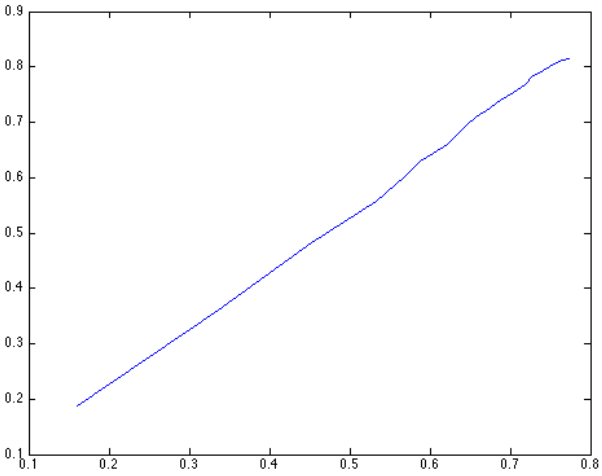


K=50 Lambda=1

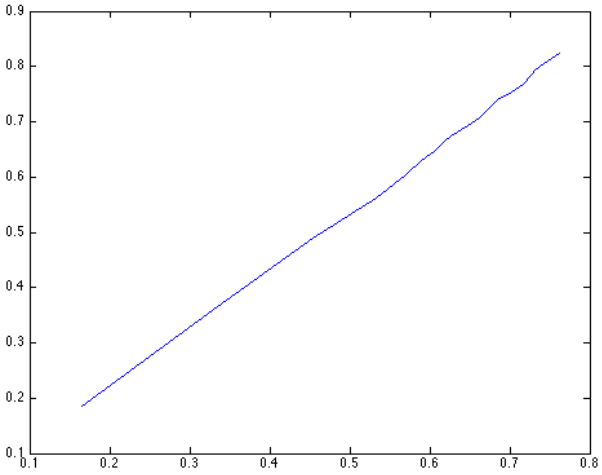
Figure 2: Plots for part 6



K=100 Lambda=0.01



K=100 Lambda=0.1



K=100 Lambda=1

Figure 3: Plots for part 6