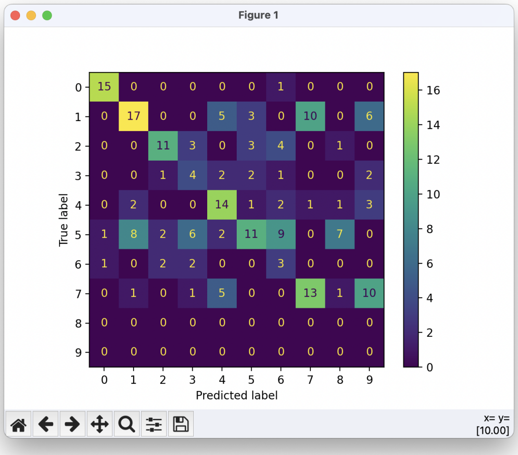
CS 349 Homework 2

1. We chose to map grayscale to binary. This was in order to treat distances the same between pixels that have at least one. Additionally, cosimilarity of two vectors does not depend on its magnitude. This also made our program run faster as we worked with smaller values. For the hyperparameter k, we decided on the value 10. This would help prevent overfitting. We found that any higher k would result in a lower-performing validation set. Here is our matrix:



	0	1	2	3	4	5	6	7	8	9
0	15	0	0	0	0	0	1	0	0	0
1	0	17	0	0	5	3	0	10	0	6
2	0	0	11	3	0	3	4	0	1	0
3	0	0	1	4	2	2	1	0	0	2
4	0	2	0	0	14	1	2	1	1	3
5	1	8	2	6	2	11	9	0	7	0
6	1	0	2	2	0	0	3	0	0	0
7	0	1	0	1	5	0	0	13	1	10
8	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0	0

This is for train size and query size 200. As we can see, our model is pretty accurate as the higher values are along the diagonal. We can see that our model gets confused with ‘1’ and ‘7’ and also ‘7’ and ‘9’.

2. For k-means, we decided to have a max iterations equal to 80. We found that any value higher than this does not change our performance. Additionally, we did not transform the data whatsoever. The results were okay. We found that our k-nearest neighbors function did much better. We found that the algorithm confused the digit ‘1’ and ‘7’ a lot as well as ‘3’ and ‘9’.

3. We should use cosimilarity because each user will most likely have a large attribute vector, meaning that the curse of dimensionality will make euclidean distances inaccurate. To estimate the rating of a movie, we should get the mean of the k nearest neighbors. This would provide a more accurate and “smooth” result as compared to getting just the mode. Firstly, we would define our k. Then, we would find the k-shortest neighbors by finding the distance between each user and the target user. We would then return the mean of the rating of these k neighbors.