Aamleen Ahmed
2020002

Assignment 2

Implemented the problem using Alternative Least Squares (ALS) Approach, with a learning rate or lambda to account for any deviations and adjust the pace accordingly, so that a correct minima can be reached fast.

Optimization to find the best value:

For finding the parameters that would give the best results, i.e the least NMAE, I observed the pattern while changing individual parameters while keeping the other constant. Also, before this I evaluated the best dataset or fold to work on.

After manually evaluating for a few parametric values to observe the range in which some significant and meaningful change happens, I extracted the NMAE values by varying individual parameters in their respective range using forloop for extracting the best parameteric combination. I then plotted the results for better pattern/data visualisation.

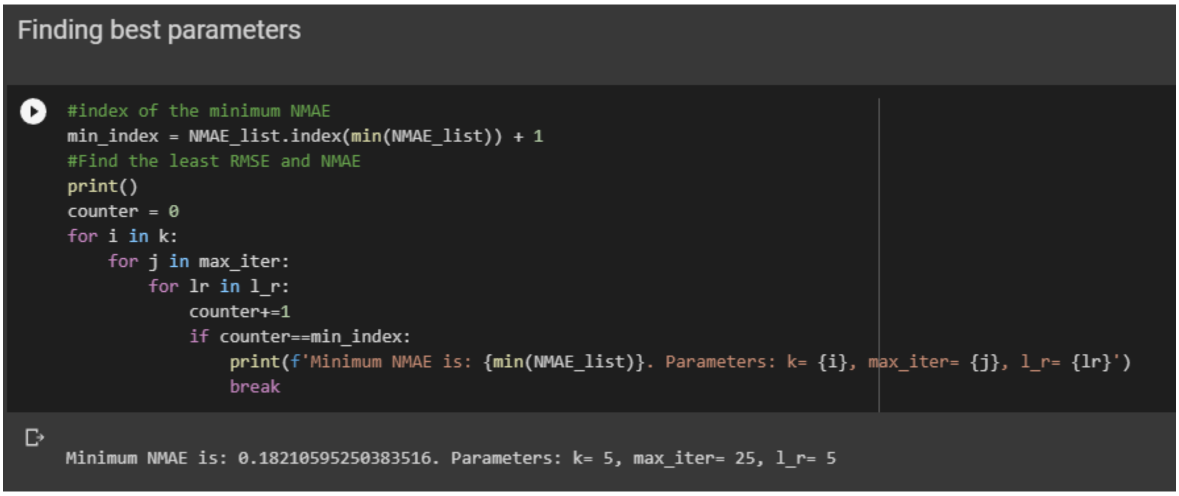
ALS Algorithm:

The SVD of the matrix would have resulted in negative ratings, so initialised U & V with random values between 0 & 1.

U is uxk matrix and V is ixk, while X is uxk matrix, where u is no. of users, I is no. of items, and k is the factors.

Then started the iterations by updating U & V. For this, obtained the indices where the ratings were present so that correct data can be used for prediction.

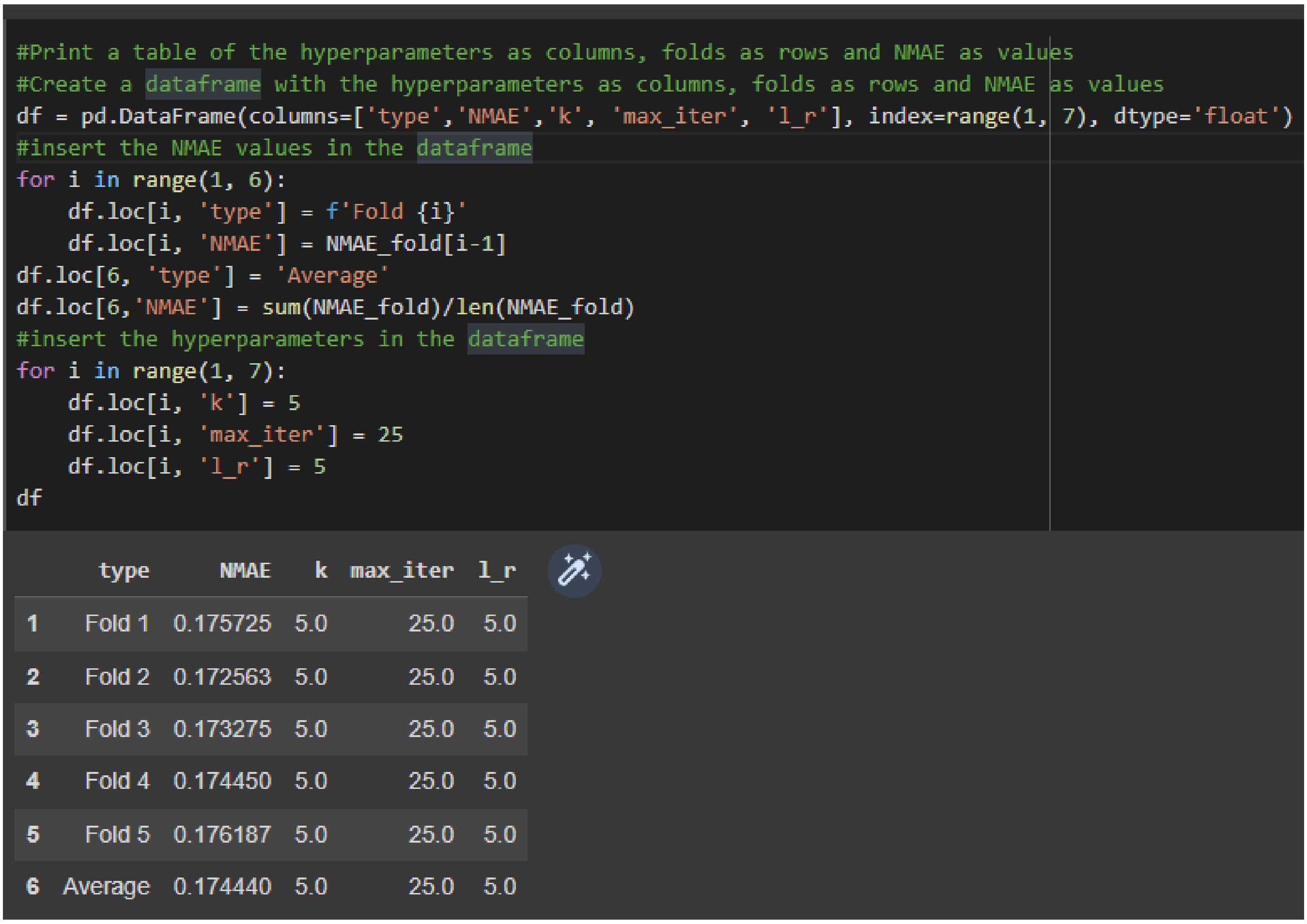
Then, computed the U first, and then V using np.linalg.solve(). The learning rate controlled the deviations and oscillations from the minima, so that the best value can be predicted without much oscillations.



Finding best parameters

```
#index of the minimum NMAE
min_index = NMAE_list.index(min(NMAE_list)) + 1
#Find the least RMSE and NMAE
print()
counter = 0
for i in k:
    for j in max_iter:
        for lr in l_r:
            counter+=1
            if counter==min_index:
                print(f'Minimum NMAE is: {min(NMAE_list)}. Parameters: k= {i}, max_iter= {j}, l_r= {lr}')
                break
```

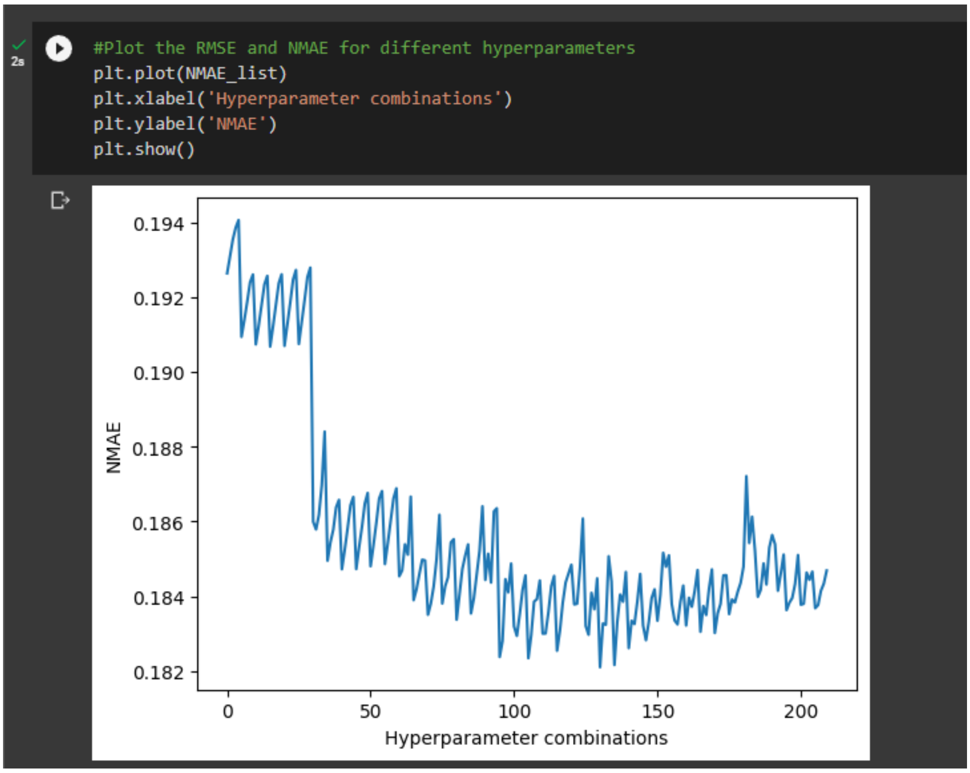
```
Minimum NMAE is: 0.18210595250383516. Parameters: k= 5, max_iter= 25, l_r= 5
```



```
#Print a table of the hyperparameters as columns, folds as rows and NMAE as values
#Create a dataframe with the hyperparameters as columns, folds as rows and NMAE as values
df = pd.DataFrame(columns=['type','NMAE','k', 'max_iter', 'l_r'], index=range(1, 7), dtype='float')
#insert the NMAE values in the dataframe
for i in range(1, 6):
    df.loc[i, 'type'] = f'Fold {i}'
    df.loc[i, 'NMAE'] = NMAE_fold[i-1]
df.loc[6, 'type'] = 'Average'
df.loc[6,'NMAE'] = sum(NMAE_fold)/len(NMAE_fold)
#insert the hyperparameters in the dataframe
for i in range(1, 7):
    df.loc[i, 'k'] = 5
    df.loc[i, 'max_iter'] = 25
    df.loc[i, 'l_r'] = 5
df
```

	type	NMAE	k	max_iter	l_r
1	Fold 1	0.175725	5.0	25.0	5.0
2	Fold 2	0.172563	5.0	25.0	5.0
3	Fold 3	0.173275	5.0	25.0	5.0
4	Fold 4	0.174450	5.0	25.0	5.0
5	Fold 5	0.176187	5.0	25.0	5.0
6	Average	0.174440	5.0	25.0	5.0

L_r is learning rate
K is the no. of factors
max_iter is the no. of iterations



```
#Plot the RMSE and NMAE for different hyperparameters
plt.plot(NMAE_list)
plt.xlabel('Hyperparameter combinations')
plt.ylabel('NMAE')
plt.show()
```