Question 3

a)

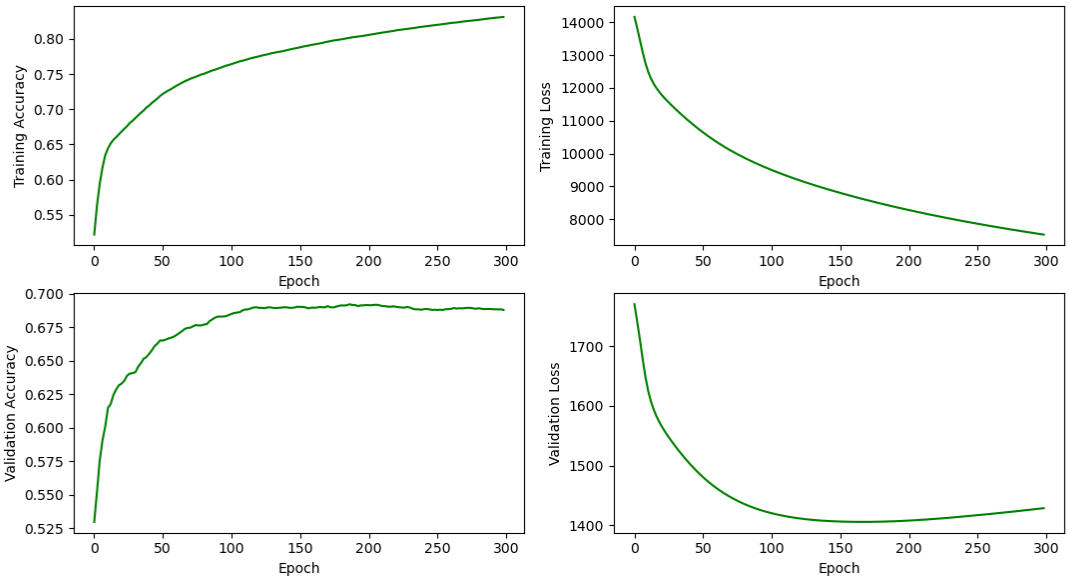
- ALS is used for matrix factorization for reconstruction - that is, factorize a matrix into two matrices U and Z, then combine them. On the other hand, the autoencoder first encodes the input by imposing a bottleneck that forces a compressed knowledge representation, then decodes it to get back the original data.
- The objective for ALS is $min_{U,Z} \frac{1}{2} \sum_{(n,m) \in O} (C_{nm} - u_n^T z_m)^2$, where C is the sparce matrix and $O = \{(n,m) \ : \ entry \ (n,m) \ of \ matrix \ C \ is \ observed\}$. The objective for autoencoder is $min_\theta \sum_{v \in S} |v - f(v;\theta)|_2^2$, where f is the reconstruction of the input v, $v \in R^{N_{questions}}$ from a set of users S.
- In ALS, we minimize the loss with respect to two factors, U and Z. However, we minimize with respect to only one factor, which is the weight matrix W, in autoencoder.
- We use gradient decent to update W at each iteration in autoencoder until convergence. In the case of ALS, the objective is non-convex in U and Z jointly, but is convex as a function of either U or Z individually. Therefore, we fix Z and optimize U, then fix U and optimize Z, so on until convergence.

b)

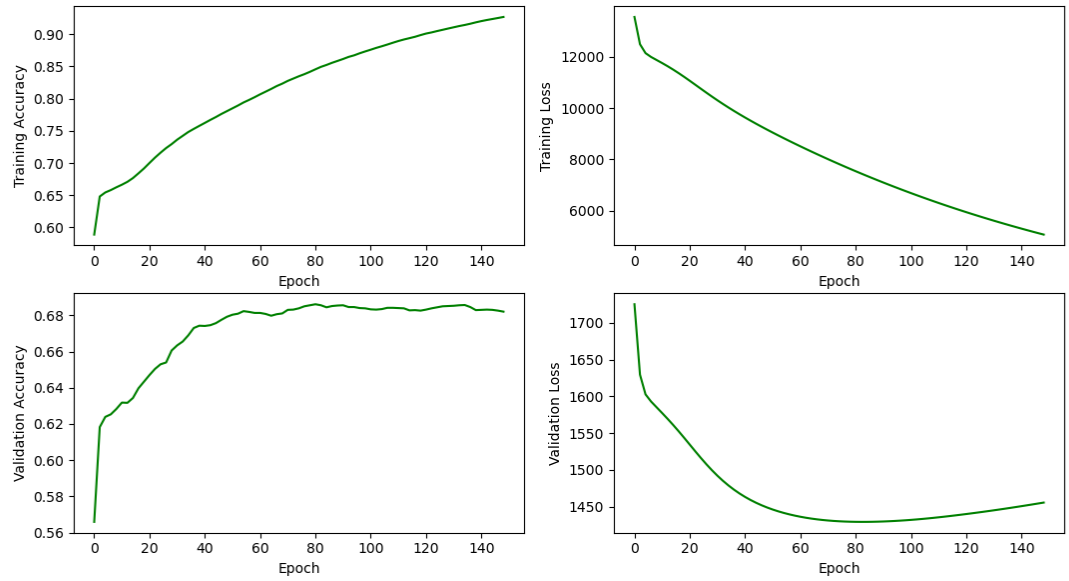
Code in neural_network.py.

c)

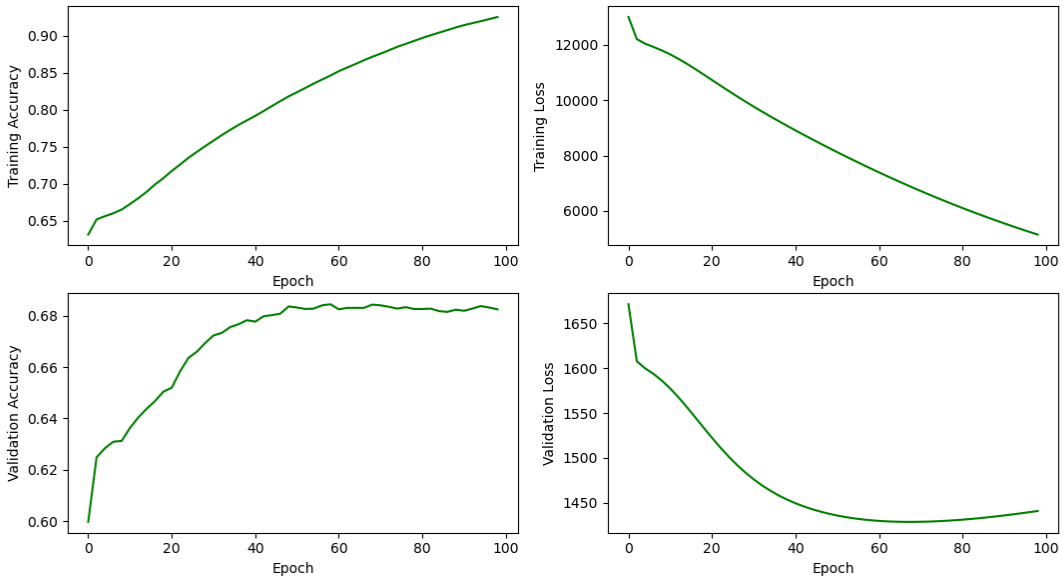
Plot for k = 10, learning rate = 0.005, num epoch = 300.



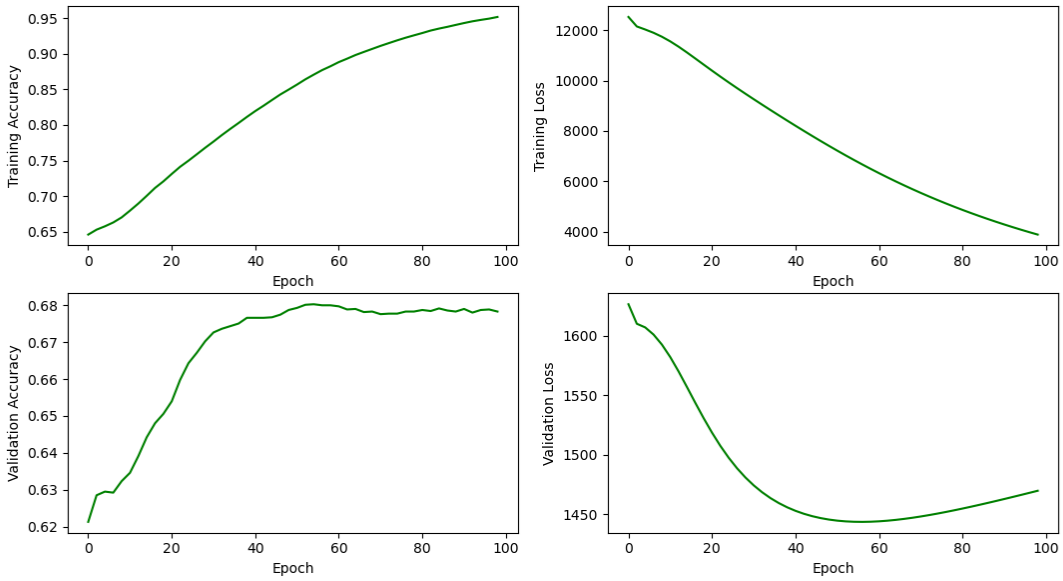
Plot for k = 50, learning rate = 0.005, num epoch = 150.



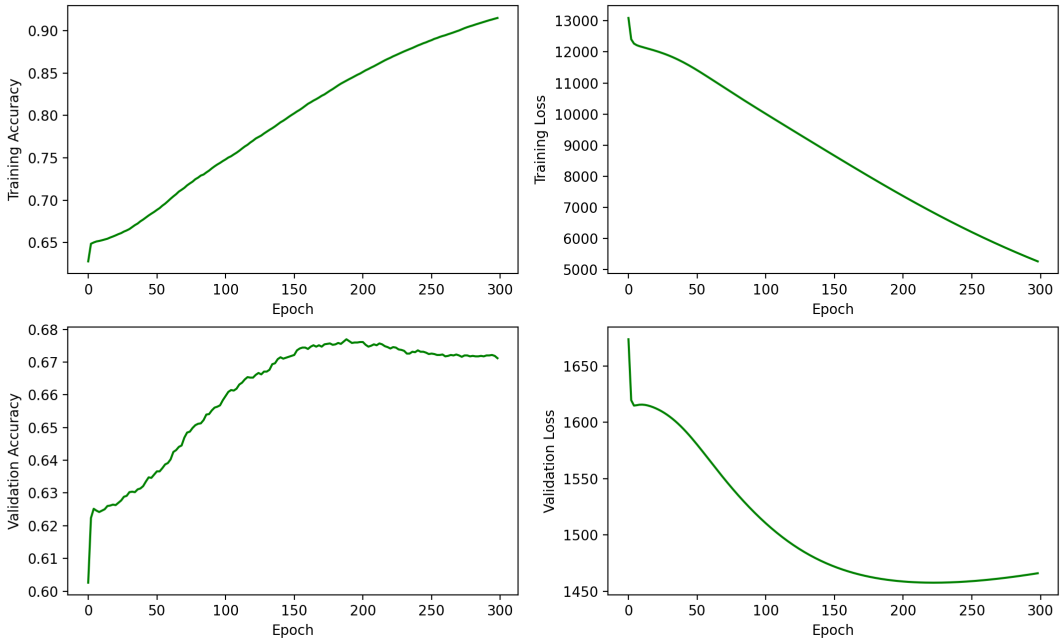
Plot for k = 100, learning rate = 0.005, num epoch = 100.



Plot for k = 200, learning rate = 0.005, num epoch = 100.



Plot for k = 500, learning rate = 0.001, num epoch = 300.

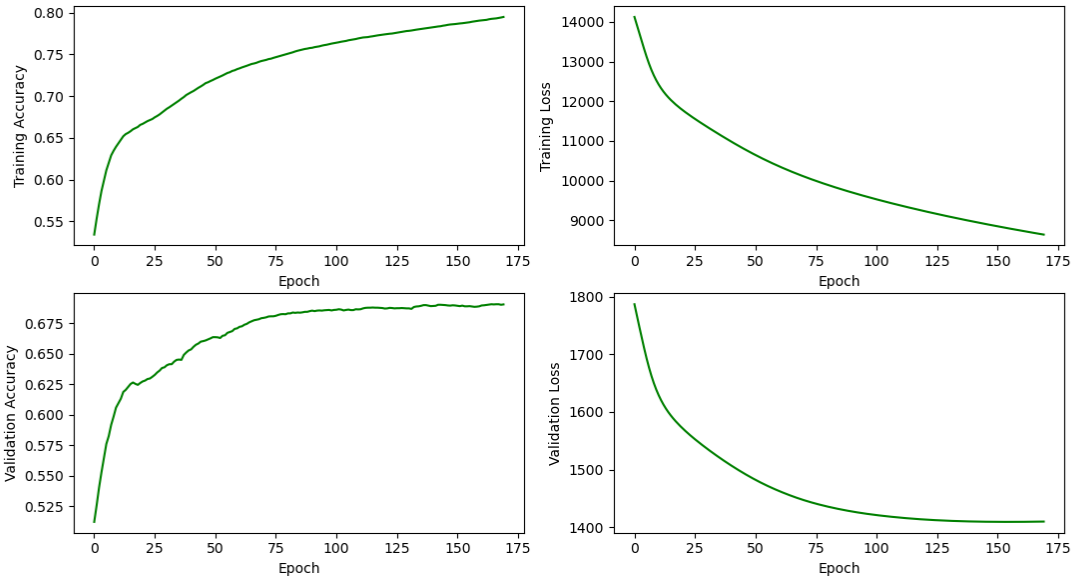


K	Highest Validation Accuracy	Learning Rate	Epoch
10	0.692210	0.005	300
50	0.686142	0.005	150
100	0.684448	0.005	100
200	0.680215	0.005	100
500	0.676969	0.001	300

The k that achieves the highest validation accracy (0.692210) is 10, so we choose it. Note the highest validaiton accuracy is calculated among the validaiton accuracy at all epoches.

d)

Plot for k = 10, learning rate = 0.005, num epoch = 170.

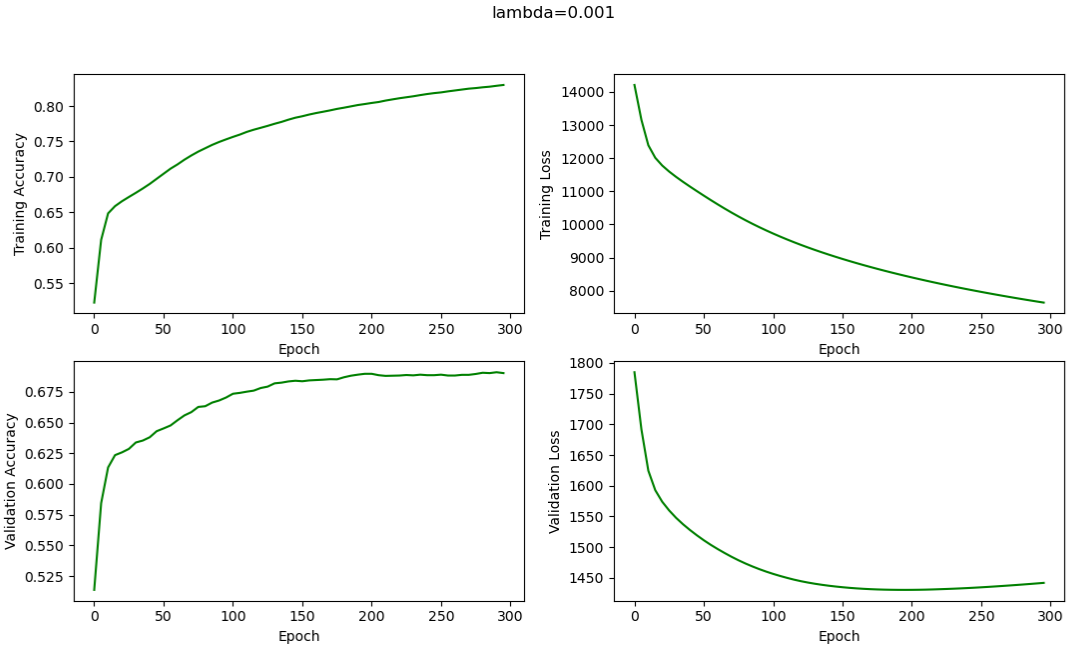


The final test accuracy is 0.691787.

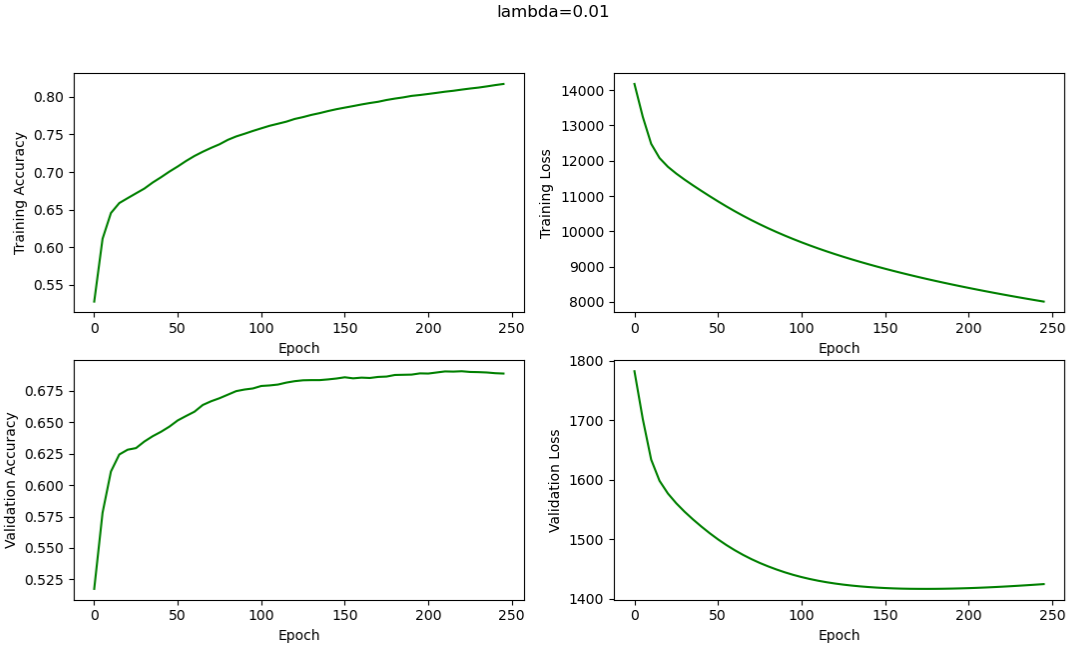
As epoch increases, the training and validation accuracies increase and the training and validation losses decrease, all with a diminishing rate. The training accuracy goes from 0.534117 to 0.794683; training loss from 14125.214311 to 8637.385720; validation accuracy from 0.512278 to 0.690375; validation loss from 1786.580161 to 1409.755266. The validation accuracy and loss eventually converge. The final training accuracy is higher than the validation and test accuracy; if more training is done, validation loss would increase, which indicates a subtle trace of overfitting.

e)

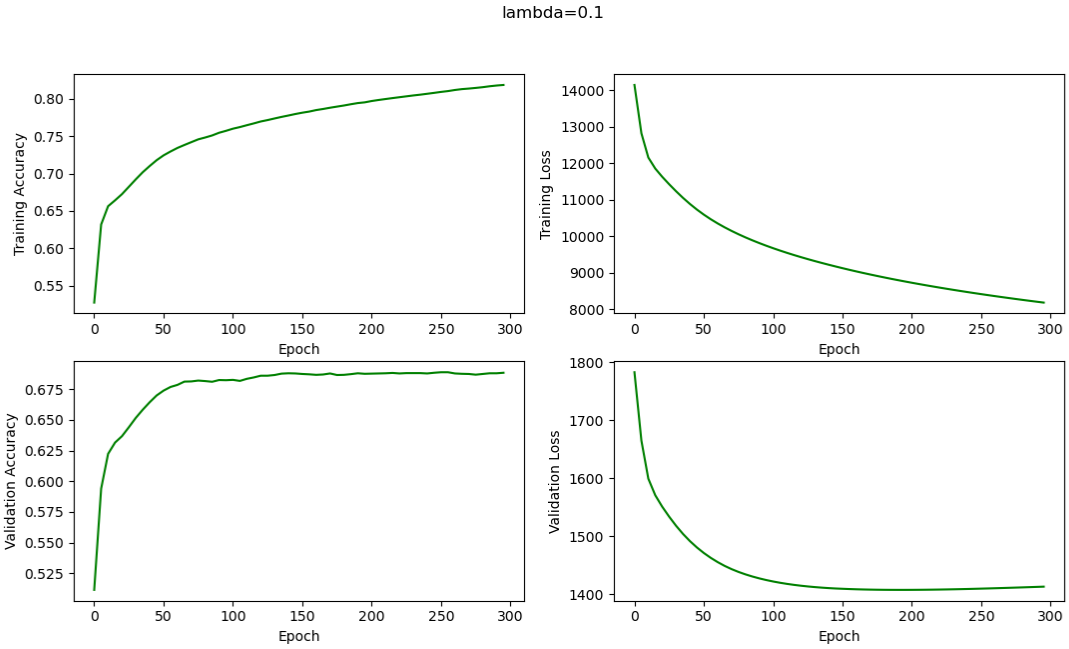
Plot for lambda = 0.001.



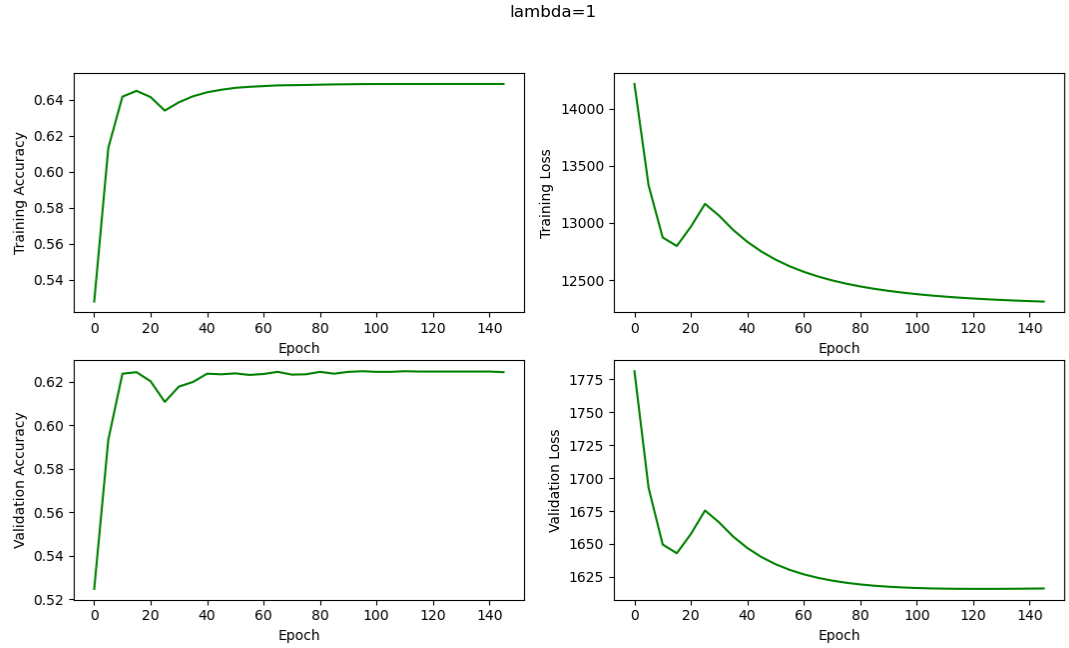
Plot for lambda = 0.01.



Plot for lambda = 0.1.



Plot for lambda = 1.0.



Lambda	Highest Validation Accuracy
0.0	0.692210
0.001	0.690940
0.01	0.690517
0.1	0.688823
1.0	0.624894

We can see model without regularization performs better, as it has a higher maximum validation accuracy. As lambda increases, the maximum validation accuracy decreases. The final validaiton accuracy is 0.690375 and test accuracy 0.691787.

In [ ]: