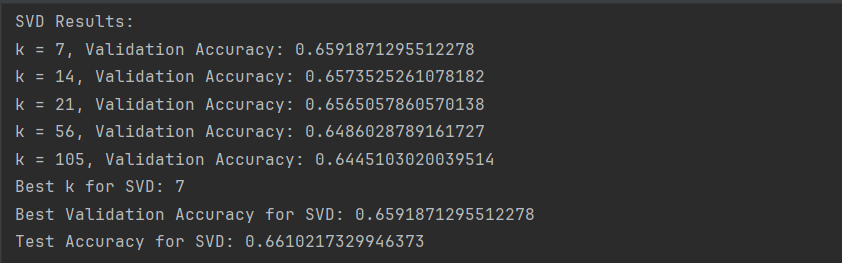
**Q4 Matrix Factorization**

(a) Singular value decomposition (SVD)

We implemented SVD to factorize the response matrix into two smaller matrices representing latent factors for students and questions. For the latent dimension k, we tried five distinct values: .

The validation accuracies for each k value are as follows:



The best k for SVD was found to be 7, and both the test and validation accuracy were approximately 0.66.

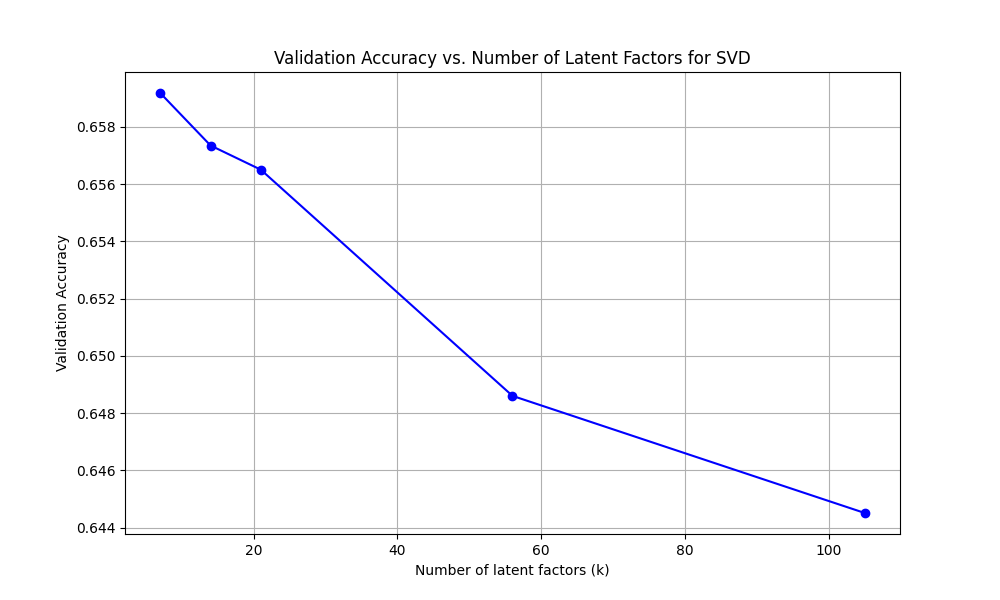
We also plot the results in the following figure:  


Figure 1. The validation accuracy versus the number of latent factors for the SVD method.

This graph shows how the validation accuracy changes with the number of latent factors for the SVD method. The accuracy is highest at the lowest number of factors (around ) and steadily decreases as the number of factors increases. This suggests that for SVD, simpler models (with fewer latent factors) perform better on the validation set, possibly due to better generalization.

(b) Alternating least square (ALS)

In this step, the ALS is implemented using stochastic gradient descent (SGD) by first initializing and iteratively updating the user (U) and question (Z) matrices randomly through the function *update\_u\_z().* Then we use this function in the *als()* function to run the entire process. We also use the same set of latent dimensions k in this step . We also use different values for the learning rate lr to look for promising convergence time and the vali dation accuracy decreased without any fluctuation (unlike some larger learning rates). The *lr* value we choose is 0.1.

The validation accuracies for each k value are as follows:

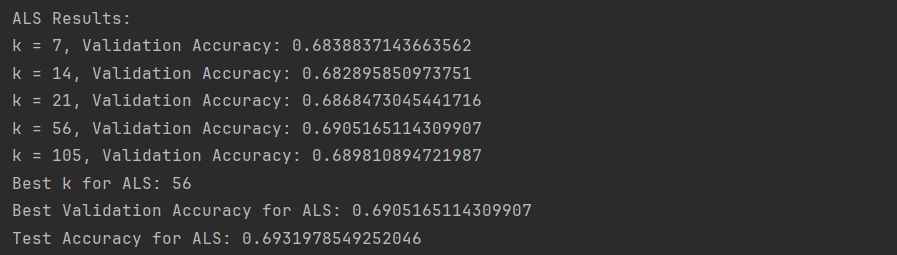


Figure 2. The ALS validation accuracies for each k values

The best k for ALS was found to be 56, with the validation accuracy to be nearly 0.69 and the test accuracy to be approximately 0.693.

The following figure shows the relationship between the validation accuracy and the number of latent factors (k) for the ALS method.

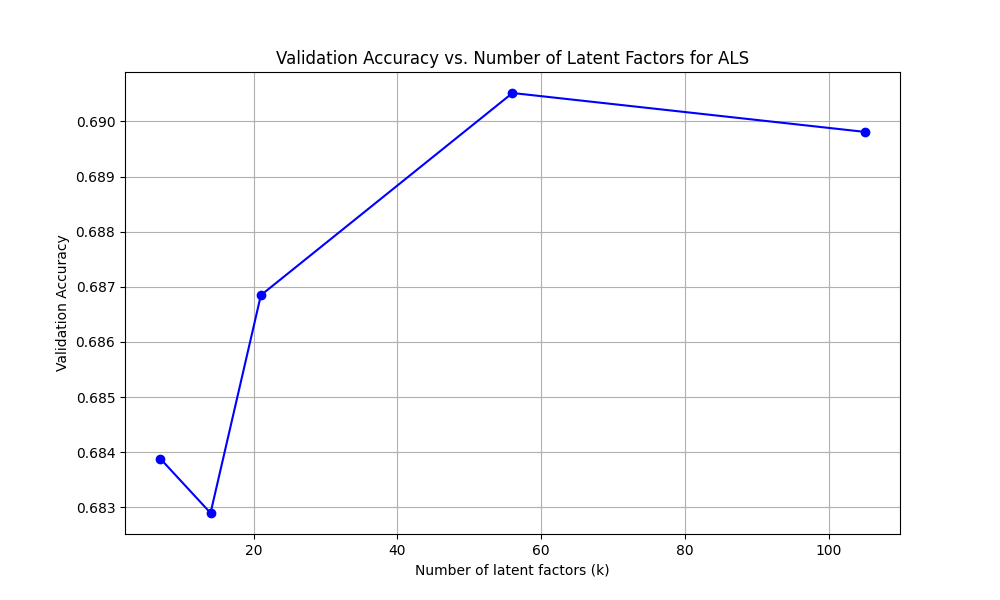


Figure 3. The validation accuracies versus the number of latent factors for the ALS.

The peak accuracy appears to occur around , with a validation accuracy of approximately 0.6905. After this peak, there's a slight decrease in accuracy as k increases to 100, but the decline is minimal.

We also have the following figure to present the training losses and validation losses in each interation of the best k:

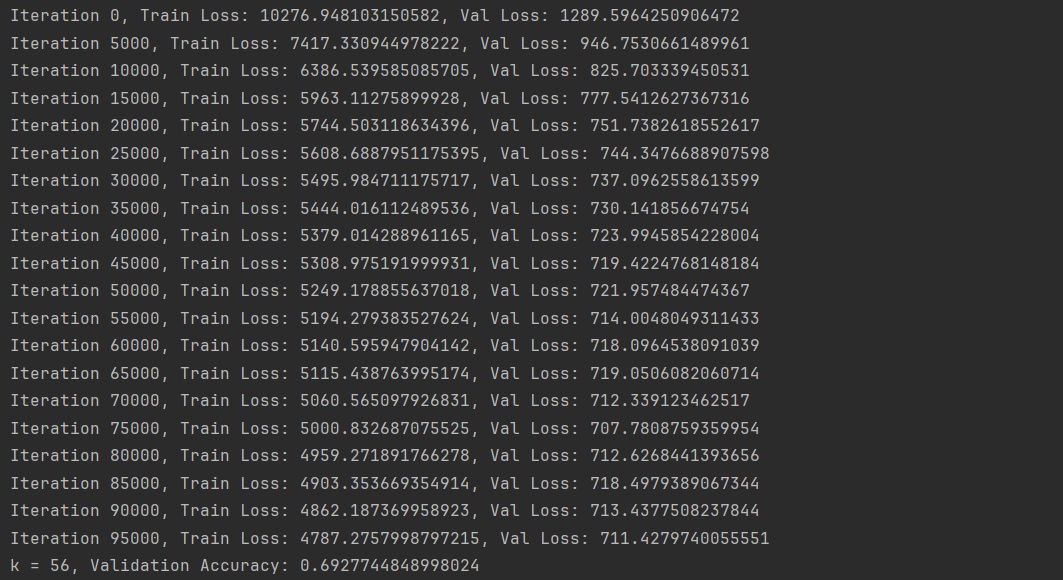


Figure 4. The training losses and validation losses in each iteration of .

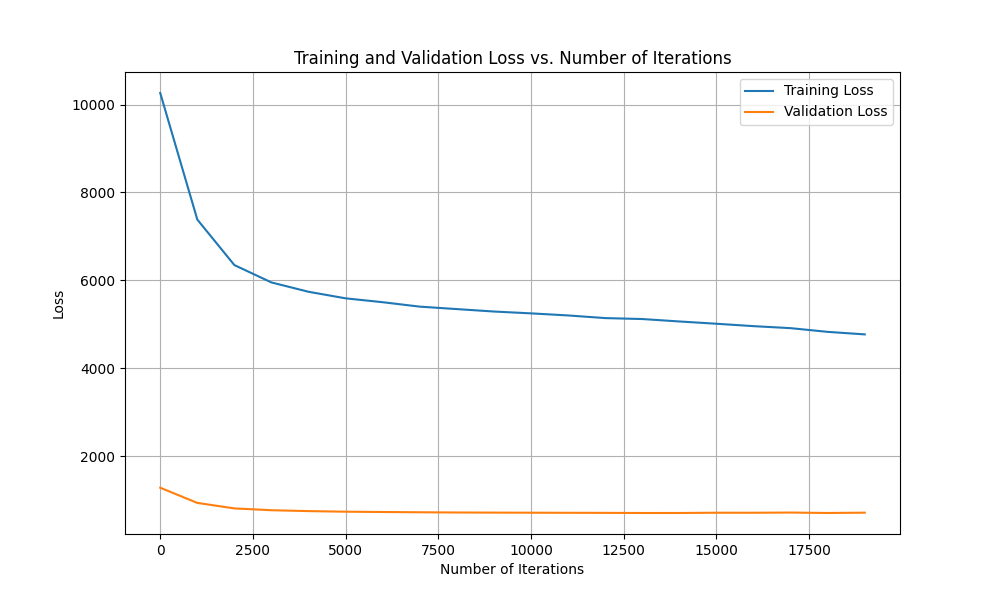


Figure 5. The training and validation loss versus the number of iterations for the ALS method.

This figure illustrates how the training and validation losses change with the number of iterations in the machine learning model. The training loss (blue line) starts very high, and consistently decreases as the number of iterations increases, indicating that the model is learning from the training data. The rate of decrease is rapid initially and then becomes more gradual.

The validation loss (orange line) starts much lower, around 1,500, and also decreases, but at a much slower rate compared to the training loss. It seems to remain unchanged after about 5,000 iterations, with only minimal changes afterwards. This suggests that the model's performance on unseen data stabilizes relatively early in the training process.

The continuing decrease in training loss, coupled with the relatively stable validation loss, indicates that the model is improving its fit to the training data without significant overfitting to the validation set.

(c)

1. Comparision of SVD and ALS

We compared the results of SVD and ALS by plotting the validation accuracies for both methods against the number of latent factors (k). This comparison is visualized in the following figure:

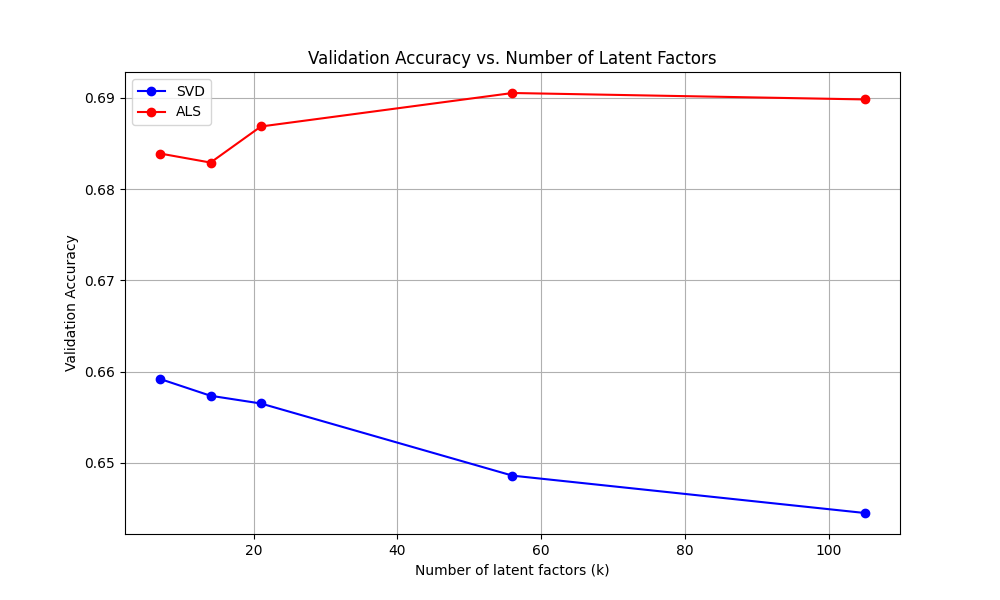


Figure 6. The validation accuracy versus the number of latent factors for both SVD (Singular Value Decomposition) and ALS methods.

2. Limitation of Matrix Factorization

* Handling of Missing Data: In the SVD implementation, we filled in missing values using the average of the current item. This approach may introduce bias, especially if the missing data is not randomly distributed.
* Cold Start Problem: Matrix factorization methods may struggle with new users or items that have no or very few ratings, as there isn't enough information to accurately place them in the latent space.
* Interpretability: The latent factors discovered by these methods are not always easily interpretable, making it challenging to explain why certain predictions are made.
* Scalability: As the number of users and items grows, the computational cost of these methods can increase significantly, especially for larger values of k.
* Assumption of Linearity: These methods assume that the interaction between user and item factors is linear, which may not always hold true in real-world scenarios.
* Temporal Dynamics: Standard matrix factorization doesn't account for changes in user preferences or item characteristics over time, which could be relevant in an educational context where student knowledge evolves.