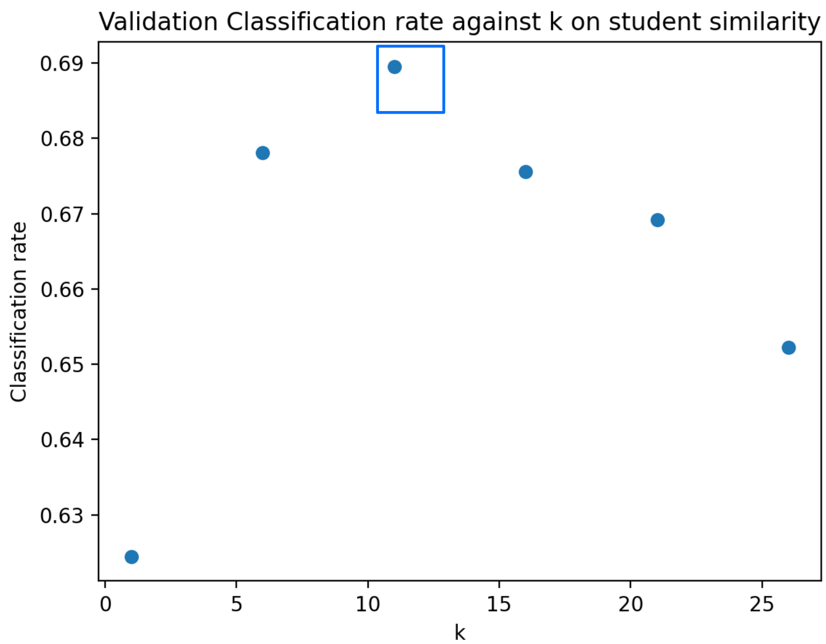
Q1 a)



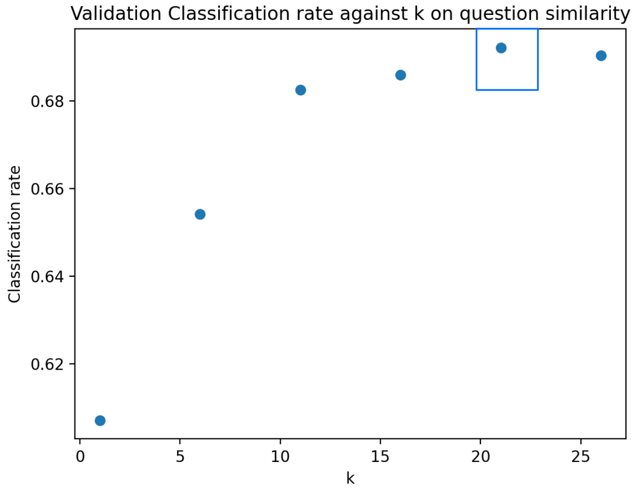
b)

```
Validation Accuracy: 0.6244707874682472
Validation Accuracy: 0.6780976573525261
Validation Accuracy: 0.6895286480383855
Validation Accuracy: 0.6755574372001129
Validation Accuracy: 0.6692068868190799
Validation Accuracy: 0.6522720858029918
Best user k: 11
Best user accuracy: 0.6895286480383855
```

$k^* = 11$

Accuracy 0.689

c)



Assumption: If question A has the same correct and incorrect answers by students as question B, A's correctness matches that of question B

From the validation test, k* = 21. Final test accuracy: 68.16%

d) Based on test data user-based collaborative filter performs better.

e) KNN problem: One potential limitation of kNN for the given task is the curse of dimensionality. In this task, there are 542 students and 1774 diagnostic questions meaning that the dimension of input student A is 1774. As the dimension gets higher, most points on the dimension get farther from each other and are approximately the same distance. This means that it is harder to distinguish between neighbors based on distance.
Another potential limitation is that kNN is sensitive to missing data, which means that it must be imputed before performing and thus might miss correlation between features.

Q2 a)

$$
\begin{aligned}
\log p(\mathbf{C} \mid \boldsymbol{\theta}, \boldsymbol{\beta}) &= \log \prod_{i=1}^{n} \prod_{j=1}^{m} \left(\frac{\exp(\theta_i - \beta_j)}{1 + \exp(\theta_i - \beta_j)} \right)^{c_{ij}} \left(\frac{1}{1 + \exp(\theta_i - \beta_j)} \right)^{(1 - c_{ij})} \\
&= \sum_{i=1}^{n} \sum_{j=1}^{m} \left(c_{ij} \log \left(\frac{\exp(\theta_i - \beta_j)}{1 + \exp(\theta_i - \beta_j)} \right) + (1 - c_{ij}) \log \left(\frac{1}{1 + \exp(\theta_i - \beta_j)} \right) \right) \\
&= \sum_{i=1}^{n} \sum_{j=1}^{m} \left(c_{ij} \left(\log(\exp(\theta_i - \beta_j)) - \log(1 + \exp(\theta_i - \beta_j)) \right) - (1 - c_{ij}) \log(1 + \exp(\theta_i - \beta_j)) \right) \\
&= \sum_{i=1}^{n} \sum_{j=1}^{m} \left(c_{ij} (\theta_i - \beta_j) - c_{ij} \log(1 + \exp(\theta_i - \beta_j)) - \log(1 + \exp(\theta_i - \beta_j)) + c_{ij} \log(1 + \exp(\theta_i - \beta_j)) \right) \\
&= \sum_{i=1}^{n} \sum_{j=1}^{m} c_{ij} (\theta_i - \beta_j) - \log(1 + \exp(\theta_i - \beta_j))
\end{aligned}
$$

Derivatives

By above, $log(p(C|\theta,\beta)) = \sum_{i=1}^{n} c_{ij}(\theta_i - \beta_j) - log(1 + exp(\theta_i - \beta_j))$

Since $\frac{\partial(log(1+exp(\theta_i-\beta_j)))}{\partial\theta_i} = \frac{1}{log(1+exp(\theta_i-\beta_j))} \frac{\partial(1+exp(\theta_i-\beta_j))}{\partial\theta_i}$

$= \frac{1}{log(1+exp(\theta_i-\beta_j))} exp(\theta_i - \beta_j) = \frac{exp(\theta_i-\beta_j)}{log(1+exp(\theta_i-\beta_j))}$

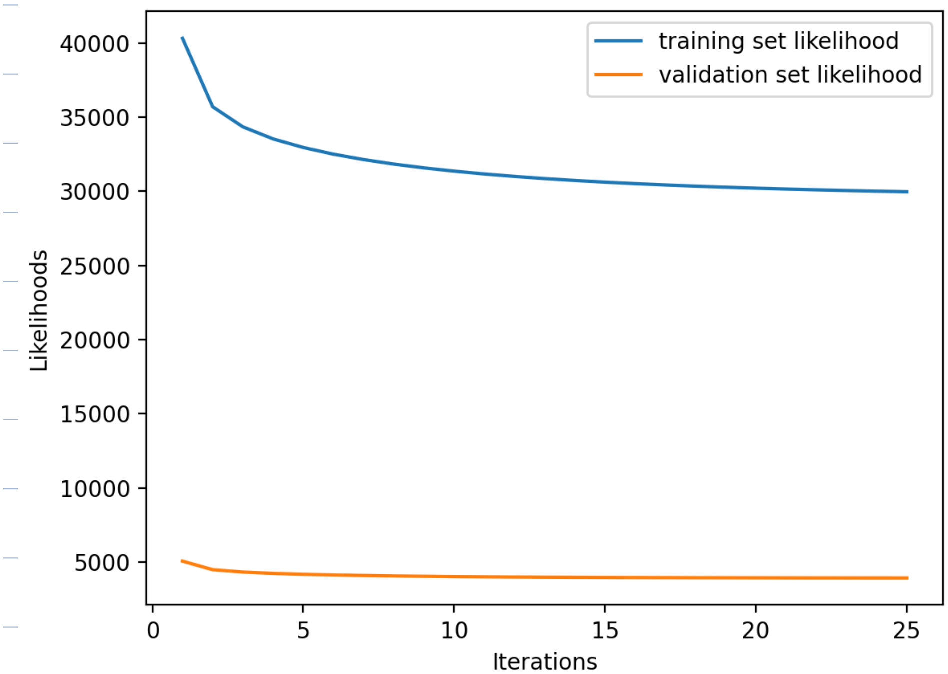
Thus, $\frac{\partial log(p(C|\theta,\beta))}{\theta_i} = \sum_{i=1}^{n} c_{ij} - \frac{exp(\theta_i-\beta_j)}{log(1+exp(\theta_i-\beta_j))}$

Similarly, we have $\frac{\partial log(p(C|\theta,\beta))}{\beta_j} = \sum_{i=1}^{n} - c_{ij} + \frac{exp(\theta_i-\beta_j)}{log(1+exp(\theta_i-\beta_j))}$

b) hyperparameter

```
Learning rate: 0.01
Number of iterations: 25
```

Likelihood curve



b) From the curve, both training and validation loss decrease, especially for the first 5 iterations. However, the loss for both curves doesn't change a lot for the following iterations.

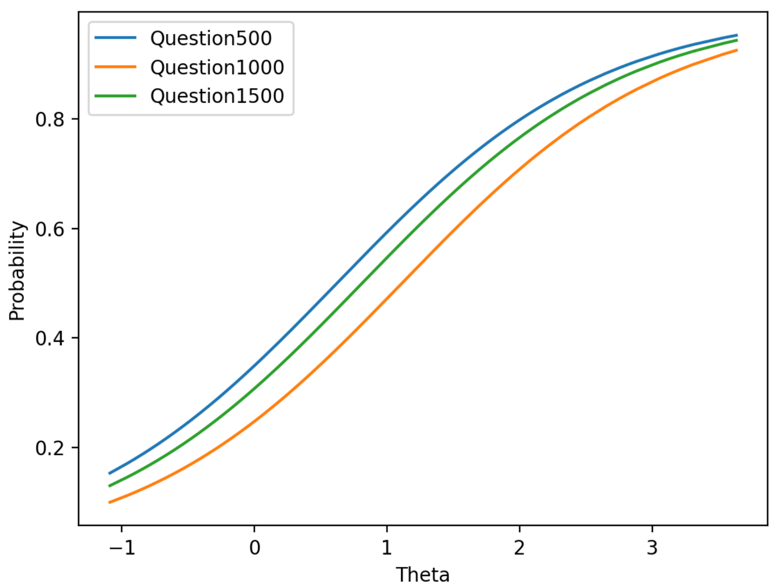
Also, we have tried a higher learning rate, eg LR = 0.1, the loss decrease dramatically, but the accuracy is relatively low.

c) Accuracy

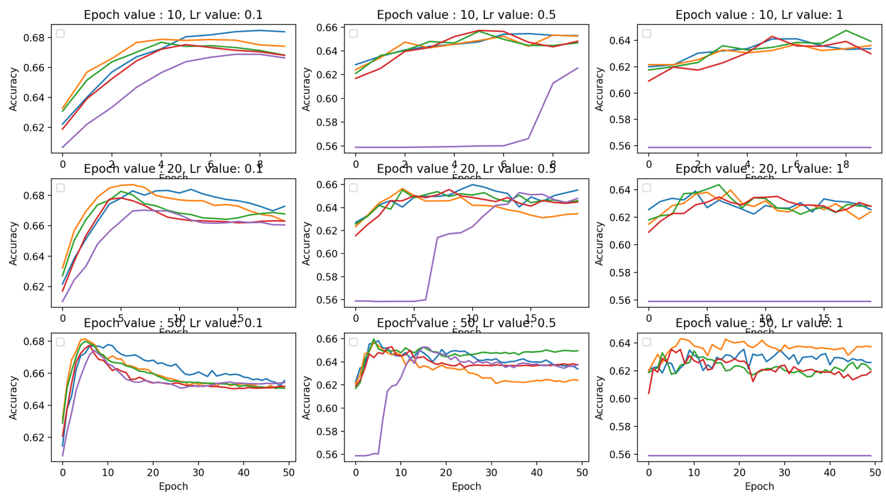
```
Final accuracy on the validation data is:0.7067456957380751
Final accuracy on the training data is:0.7047699689528648
```

d) For this plot, x axis repersents student's abilities and y axis repersents student's probability of answering selected question correctly. We know with the increase of theta, the probabilities also increase. This indicates a positive relationship between two variables.

Shape of curve: For these three curves, they are concave within interval [-1, 1] and they are convex within interval [1,3]



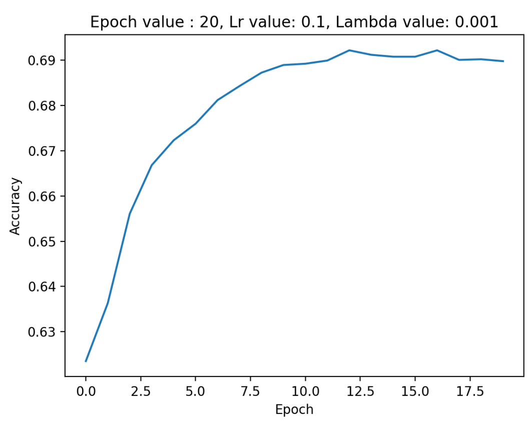
c)

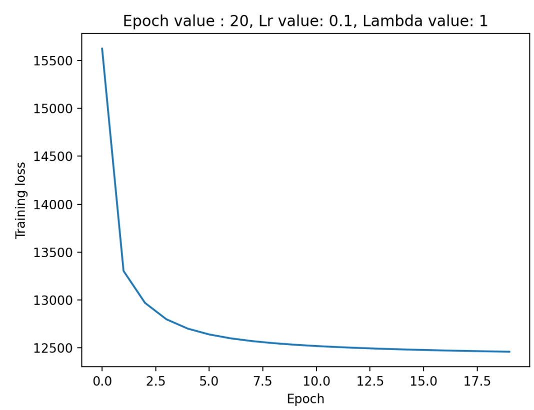


Run the model again $k^* = 10$

```
Highest validation accuracy:0.6831780976573525
```

d)

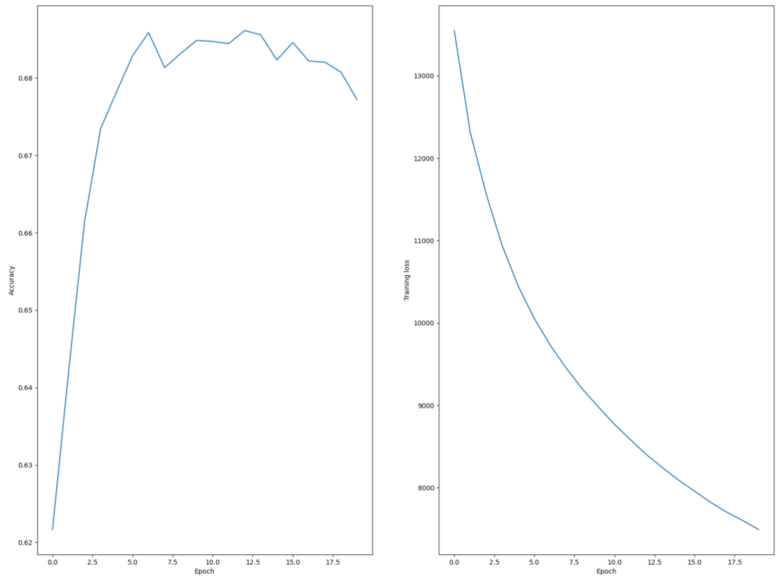





```
Highest validation accuracy:0.6891052780129834
Final test accuracy: 0.6799322607959356
```

e)

```
The best accuracy: 0.6922099915325995 for lambda = 0.001
```




```
Final test accuracy with regularizer: 0.6260231442280553
```

By comparing performance of different model, lambda = 0.001 is good, best accuracy is 0.69220. Comparing to c), the accuracy increase a little bit, so the model with regularizer performs better.