CSC311 Project Writeup

Jacob Yoke Hong Si, Jay Jongwoo Lim, Evence YiFan Wang

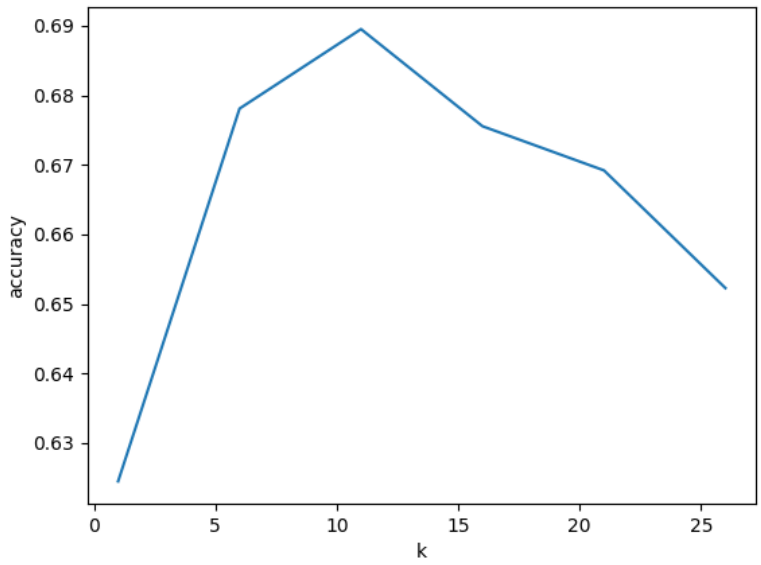
December 2021

Part A

1. k-Nearest Neighbor

a

Here are the accuracy with $k = 1, 6, 11, 16, 21, 26$ with using kNN, and imputing by users.




```
Validation Accuracy: 0.6244707874682472
Validation Accuracy: 0.6780976573525261
Validation Accuracy: 0.6895286480383855
Validation Accuracy: 0.6755574372001129
Validation Accuracy: 0.6692068868190799
Validation Accuracy: 0.6522720858029918
```

b

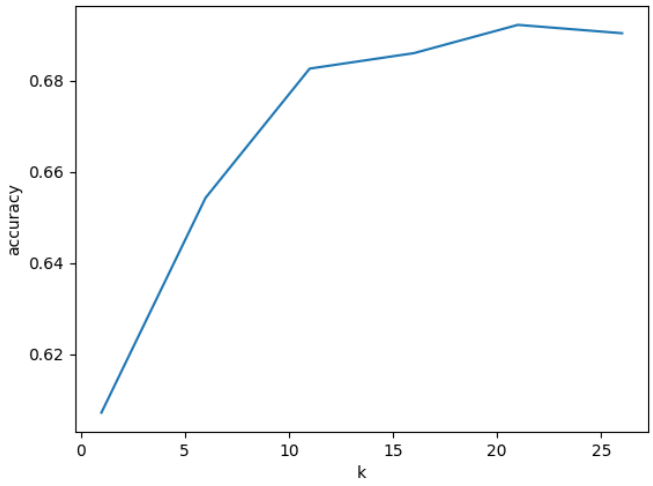
We can tell in section a, that the k with highest accuracy was $k = 11$ with a validation accuracy of 0.6895, thus, $k^* = 11$ on the testing dataset. here is the accuracy we were able to achieve:

```
0.6841659610499576
```

c

We know that if we are imputing by questions, then we are making the assumption that if a student gets most of the questions wrong about a certain subject, then that student knows nothing about that subject and will thereby get every single question with the subject wrong.

With item based collaborative filtering, we achieved such results:




```
Validation Accuracy: 0.607112616426757
Validation Accuracy: 0.6542478125882021
Validation Accuracy: 0.6826136042901496
Validation Accuracy: 0.6860005644933672
Validation Accuracy: 0.6922099915325995
Validation Accuracy: 0.69037538808919
```

We can tell that the highest validation accuracy occurred with $k = 21$ at 0.6922, and with this value of k we were able to achieve a testing accuracy of:

```
0.6683601467682755
```

d

We can tell from the result in part a as well as part c that although item based collaborative filtering gave us a better overall result in the validation accuracy, user based collaborative filtering method achieved a higher testing accuracy in the end with 0.6842, while the item based collaborative filtering achieved 0.6684.

e

1. The accuracy is very much based on the data given to us, and how sparse it may be, there could be potential underfitting if the data is too sparse.

2. There might be questions that only a few students answered so the correctness of those questions will yield a low accuracy. If the user only answered a handful of questions, it could limit the algorithm to correctly guess the answers with only a few k neighbors to compare to. With a big sparse matrix, there may be students who answered a question that no other students answered.

3. In our dataset, we are tasked with applying the KNN algorithm on the answers of 542 students to 1774 diagnostic questions. Therefore, we may encounter issues with the curse of dimensionality since there would be a high input dimension. Hence, most points would be approximately the same distance and no meaningful clusters can be formed for us to classify the points.

2. Item Response Theory

a

We will try to obtain the log-likelihoods first.
From the formula given, we can get this:

$$\begin{aligned} p(c_{ij} = 1|\theta_i, \beta_j) &= \frac{exp(\theta_i - \beta_j)}{1 + exp(\theta_i - \beta_j)} \\ p(c_{ij} = 0|\theta_i, \beta_j) &= 1 - \frac{exp(\theta_i - \beta_j)}{1 + exp(\theta_i - \beta_j)} \\ &= \frac{1}{1 + exp(\theta_i - \beta_j)} \end{aligned}$$

And so, we can make the generalization like this:

$$p(c|\theta, \beta) = \prod_{i=1}^{N} \prod_{j=1}^{M} \left(\frac{exp(\theta_i - \beta_j)}{1 + exp(\theta_i - \beta_j)} \right)^{c} \left(\frac{1}{1 + exp(\theta_i - \beta_j)} \right)^{(1-c)}$$

And hence:

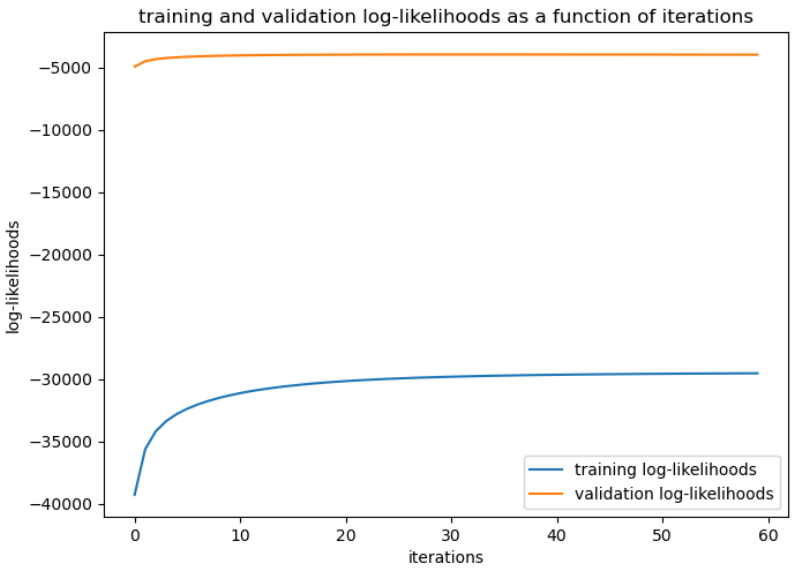
$$\begin{aligned} \log p(c|\theta, \beta) &= \sum_{i=1}^{N} \sum_{j=1}^{M} \left(c \ \log \frac{exp(\theta_i - \beta_j)}{1 + exp(\theta_i - \beta_j)} + (1 - c) \ \log \left(\frac{1}{1 + exp(\theta_i - \beta_j)} \right) \right) \\ &= c \ (\log exp(\theta_i - \beta_j) - \log(1 + exp(\theta_i - \beta_j))) + (1 - c) \ (\log(1) - \log(1 + exp(\theta_i - \beta_j))) \\ &= c \ (\theta_i - \beta_j) - \log(1 + exp(\theta_i - \beta_j)) \end{aligned}$$

And now we can calculate for the derivatives with respect to θ and β:

$$\begin{aligned} \frac{dp}{d\theta_i} &= c - \frac{exp(\theta_i - \beta_j)}{1 + exp(\theta_i - \beta_j)} \\ \frac{dp}{d\beta_j} &= \frac{exp(\theta_i - \beta_j)}{1 + exp(\theta_i - \beta_j)} - c \end{aligned}$$

b

Implementation of the alternating gradient descent is in the code submitted.
We decided to go with a learning rate of 0.1, as well as 60 iterations.
Below is the training curve for training and validation log-likelihoods as a function of iteration.



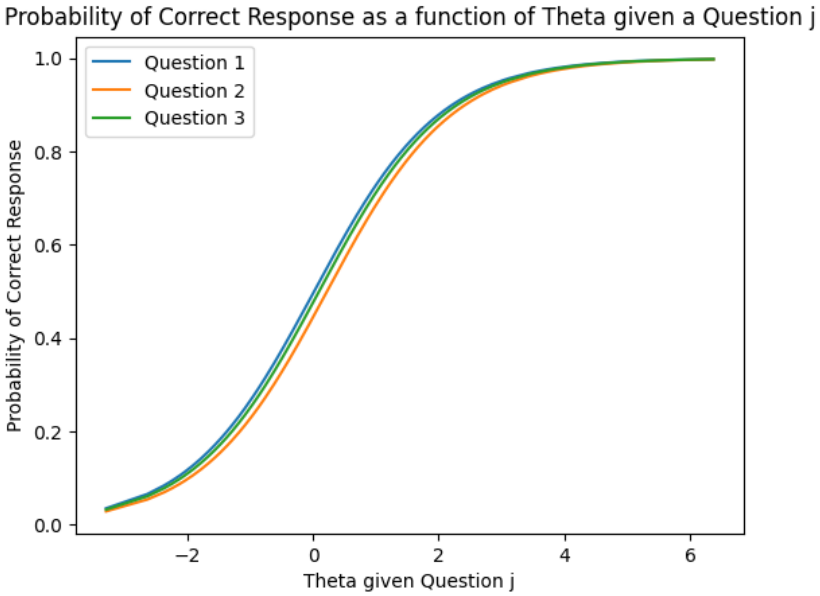
c

Below are the final validation and testing accuracy.

```
Validation Accuracy: 0.7050522156364663
Test Accuracy: 0.7129551227773073
```

d

Below are the three curves on the same plot.



We can tell that all three curves fit very closely together, and they all represent a shape that resembles the sigmoid curve. The curves indicate the probability that a student will answer a question correctly, as a function to the student's own ability, θ. And the higher the student's ability, the better chance the student will have in answering the question correctly. And due to the difference in difficulty of each question, the curves are slightly different.

3. Option 2: Neural Networks

a

1. ALS minimizes two loss functions alternatively, while the traditional neural networks minimizes only one loss function.
2. ALS requires different hyperparameters compared to the traditional neural networks. Some of the ALS hyperparameters are the maximum number of iterations and the number of latent factors in the model. Some of the neural network hyperparameters could be the number of hidden layers in the model, the activation function, etc.
3. Traditional neural networks can learn using multiple epochs with batches to lower the loss while ALS has no way of learning using epochs.

b

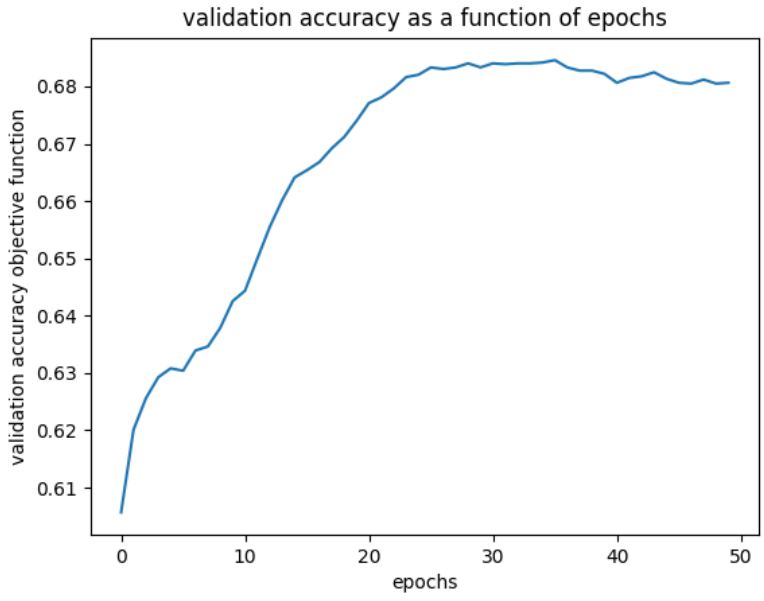
The completed code is in the submission.

c

The completed code is in the submission.
With our code, we achieved the highest validation accuracy with $k = 50$, a learning rate of 0.01 and 50 iterations. Therefore, $k^* = 50$

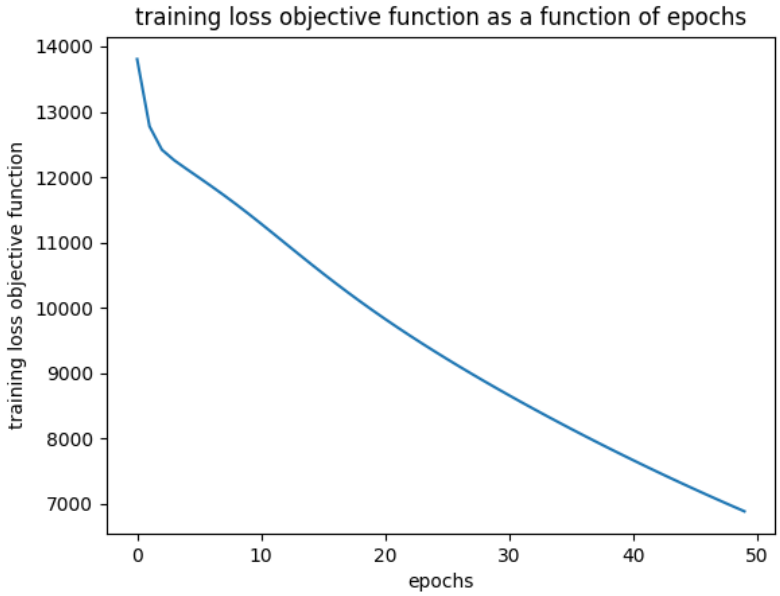
d

Below is the validation objective changes as a function of epoch:



Continued on the next page

Below is the training loss changes as a function of epoch:



Below is the final test accuracy with $k = 50$

```
Final Test Accuracy: 0.6816257408975445
```

e

After running each λ, we realized that $\lambda = 0.01$ yielded the best results.
Here is the final validation and testing accuracy:

```
Epoch: 49   Training Cost: 11291.535563  Valid Acc: 0.6820491109229466
Final Test Accuracy: 0.6776742873271239
```

So the model, in fact, did not perform better with the regularization penalty, as we can see, the testing accuracy for the model without the penalty was 0.6816, and the accuracy for the model with the penalty was 0.6777.

4. Ensemble

We made 3 ensemble sparse matrices that each takes the same number of users as the original data set, but it picks random users from the data set with repetition. For each sparse matrix, we fill in the NaN values with kNN using kNNImputer. We then create three original sparse matrices and put the filled in values to the corresponding user in the original data set sparse matrix. For users that is in the original data set but not the ensemble sparse matrix, we fill in the NaN values using the data from the sparse matrix using our own kNN by user algorithm that we created.

We had the best result when $k = 6$, here are the validation and testing accuracy, with validation on the top and testing on the bottom:

```
0.6391476150155235
0.6415467118261361
```

We did not obtain better results by using the ensemble. The limitation to the kNN algorithm is that it requires good data set to begin with in order to achieve better accuracy. The repetition of some users may have generated us with a bad data set which in turn, lowered our accuracy.

Part B

1. Formal Description

We are proposing to use the metadata provided to us, particularly the columns that contain information about premium students as well as the subject information related to each question.
We will modify the Item Response Theory (IRT) model to fit the said extension, and we will provide the new mathematical equation down below.

$$\begin{aligned} p(c_{ij} = 1|\alpha_p, \theta_i, \beta_j, \gamma_s) &= \frac{exp(\alpha_p + \theta_i - \beta_j - \gamma_s)}{1 + exp(\alpha_p + \theta_i - \beta_j - \gamma_s)} \\ p(c_{ij} = 0|\alpha_p, \theta_i, \beta_j, \gamma_s) &= 1 - \frac{exp(\alpha_p + \theta_i - \beta_j - \gamma_s)}{1 + exp(\alpha_p + \theta_i - \beta_j - \gamma_s)} \\ &= \frac{1}{1 + exp(\alpha_p + \theta_i - \beta_j - \gamma_s)} \end{aligned}$$

And so, we can make the generalization like this:

$$p(c|\alpha_p, \theta_i, \beta_j, \gamma_s) = \prod_{i=1}^{N} \prod_{j=1}^{M} \left(\frac{exp(\alpha_p + \theta_i - \beta_j - \gamma_s)}{1 + exp(\alpha_p + \theta_i - \beta_j - \gamma_s)} \right)^c \left(\frac{1}{1 + exp(\alpha_p + \theta_i - \beta_j - \gamma_s)} \right)^{(1-c)}$$

And hence:

$$\begin{aligned} \log p(c|\alpha_p, \theta_i, \beta_j, \gamma_s) &= \sum_{i=1}^{N} \sum_{j=1}^{M} \left(c \ \log \frac{exp(\alpha_p + \theta_i - \beta_j - \gamma_s)}{1 + exp(\alpha_p + \theta_i - \beta_j - \gamma_s)} + (1 - c) \ \log \left(\frac{1}{1 + exp(\alpha_p + \theta_i - \beta_j - \gamma_s)} \right) \right) \\ &= c \ (\log exp(\alpha_p + \theta_i - \beta_j - \gamma_s) - \log(1 + exp(\alpha_p + \theta_i - \beta_j - \gamma_s))) \\ &+ (1 - c) \ (\log(1) - \log(1 + exp(\alpha_p + \theta_i - \beta_j - \gamma_s))) \\ &= c \ (\alpha_p + \theta_i - \beta_j - \gamma_s) - \log(1 + exp(\alpha_p + \theta_i - \beta_j - \gamma_s)) \end{aligned}$$

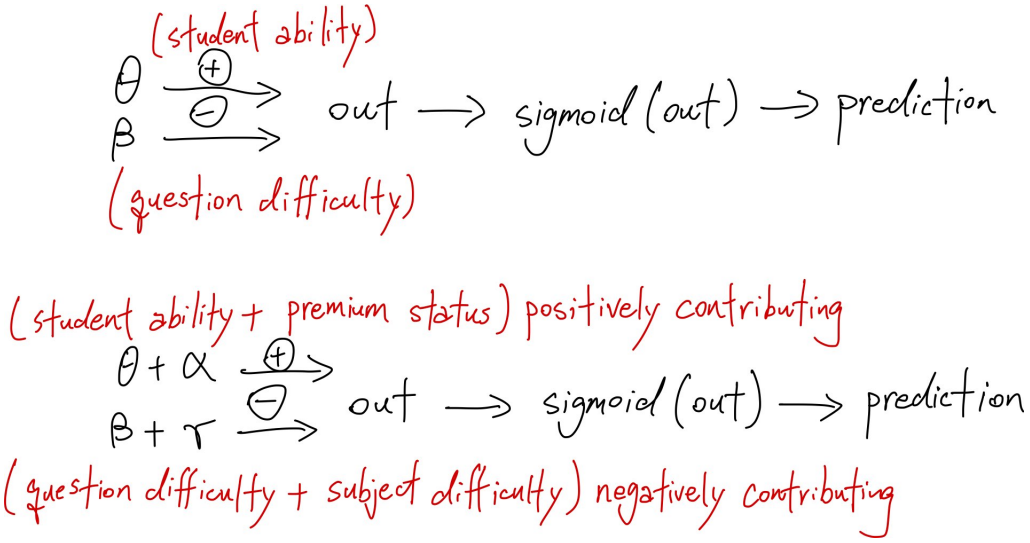
And now we can calculate for the derivatives with respect to θ, β, α, γ:

$$\begin{aligned} \frac{dp}{d\theta_i} &= c - \frac{exp(\alpha_p + \theta_i - \beta_j - \gamma_s)}{1 + exp(\alpha_p + \theta_i - \beta_j - \gamma_s)} \\ \frac{dp}{d\beta_j} &= \frac{exp(\alpha_p + \theta_i - \beta_j - \gamma_s)}{1 + exp(\alpha_p + \theta_i - \beta_j - \gamma_s)} - c \\ \frac{dp}{d\alpha_p} &= c - \frac{exp(\alpha_p + \theta_i - \beta_j - \gamma_s)}{1 + exp(\alpha_p + \theta_i - \beta_j - \gamma_s)} \\ \frac{dp}{d\gamma_s} &= \frac{exp(\alpha_p + \theta_i - \beta_j - \gamma_s)}{1 + exp(\alpha_p + \theta_i - \beta_j - \gamma_s)} - c \end{aligned}$$

So rather than having two parameters in our IRT model (θ and β) we will add two more: α, and γ. Where α_p represents the premium pupil status of pupil p, and γ_s represents the subject of subject s.
With this proposed model, we are hoping to achieve a higher accuracy and improved optimization by involving the work ethics of students from different financial backgrounds, as well as the overall difficulty of subjects.

2. Figure or Diagram

Here is our hand-drawn figure showing the basic idea of how the new model will work compared to the IRT given to us.



The image at the top shows the IRT model provided to us in Question 2 where we use $\theta - \beta$ and passed the result into the sigmoid function to obtain the resulting prediction.

Since we wanted to create a model that would yield better results, we needed more polarizing results going into the sigmoid function, and the premium student status, as well as the subject difficulty will do just that. The student ability plus the premium status (denoted by $\theta + \alpha$) positively contributes to the result, while the question difficulty plus subject difficulty (denoted by $\beta + \gamma$) negatively contributes to the result.

3. Comparison or Demonstration

After running our implemented algorithm, we observe that the accuracy has been improved. We used a learning rate of 0.1 and ran it for 60 iterations for both algorithms.

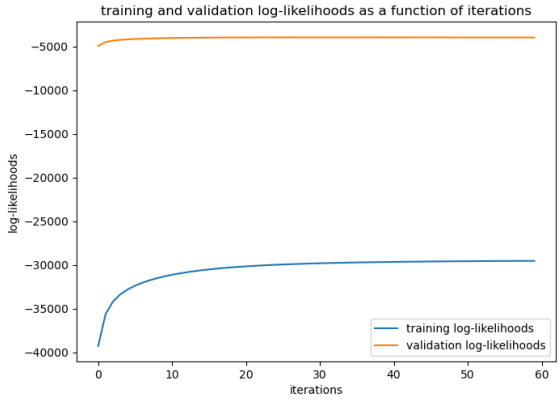
The following is the validation and testing accuracy for the original algorithm in Part A:

```
Validation Accuracy: 0.7050522156364663
Test Accuracy: 0.7129551227773073
```

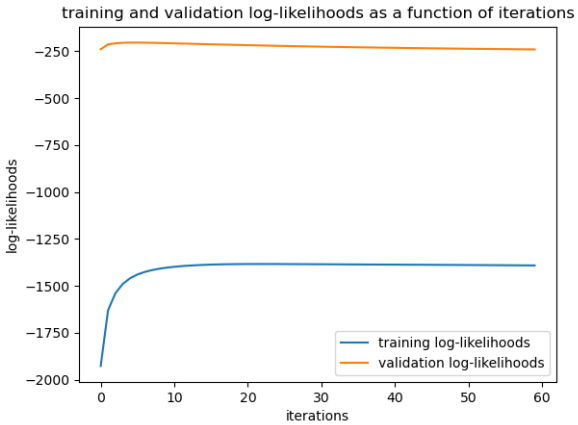
And here is the validation and testing accuracy for Part B, the modified version:

```
Validation Accuracy: 0.7217391304347827
Test Accuracy: 0.7546762589928058
```

The following is Part A, Q2's log likelihood graph:



And this is Part B's log likelihood graph:



We observe that the validation accuracy increase by around 2% and the test accuracy increased by around 4% when comparing our modified IRT in part B. to the original algorithm in part A. which is not a trivial improvement. Furthermore the negative log-likelihood is much lower for our algorithm which indicates an improvement in our predictions. Since we are able to incorporate more features in the model, we will be able to train the model more accurately and optimize with a higher accuracy.

4. Limitations

Since we are using the metadata provided, there is assurance that the algorithm works as long as there is consistent data inside that we can work with. However, we think that if the premium student data was a little more sparse, then it is possible that we don't get enough information we need to make the optimization happen. And of course, if the metadata was not provided at all, the entire model would fail. We think that the sole reason the model is working right now is because there is a direct correlation between the student's ability and their economic status, and so if the metadata were to be a lot more sparse or even empty, then our modification would not make a difference at all. A possible extension to this issue is to collect more data without missing values of premium students. Hence, this would improve the statistical power of our predictions as well as reducing any potential bias of our algorithm.

Another possible extension to improve our algorithm is that we can include more independent variables in our model. For instance, if we were to include a parameter that denotes the time spent by the student studying, it could further increase the predictability of our model, since we would expect a student who studies more is more likely to achieve a correct answer. However, if we were to include variables that could be uncorrelated with a students' correctness to diagnostic questions, it may lead to data dredging where we would over fit our data, leading to misinterpretation of our results. Furthermore, if we were to include more variables, it would require the machine to compute more gradients which would potentially increase the time complexity of the algorithm, leading to longer computation time.

Lastly, during our analysis, we found that high number of iterations would lead to a decrease in the negative log-likelihoods hence, we should ensure that we implement early stopping to prevent any overfitting of the data.

Contributions

Jacob Yoke Hong Si: Completed questions 2, 3 and part B as well as edited the final write-up.

Jay Jongwoo Lim: Completed questions 1 and 4 as well as question 3 (a).

Evence YiFan Wang: Assisted Jacob to complete questions 2, 3 and part B as well as wrote the final write-up.