CSC311 Summer 2023 Final Report

A machine learning model analysis on predicting students' correctness of a given diagnostic question.

Edited By

Ian Quan

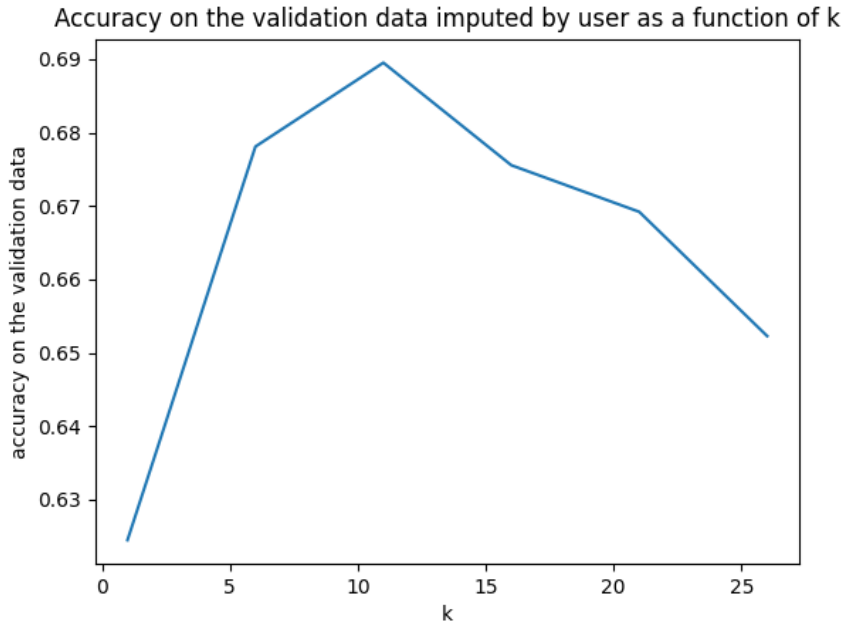
Douglas Quan

University of Toronto

1 [5pts] k-Nearest Neighbor

Solution.

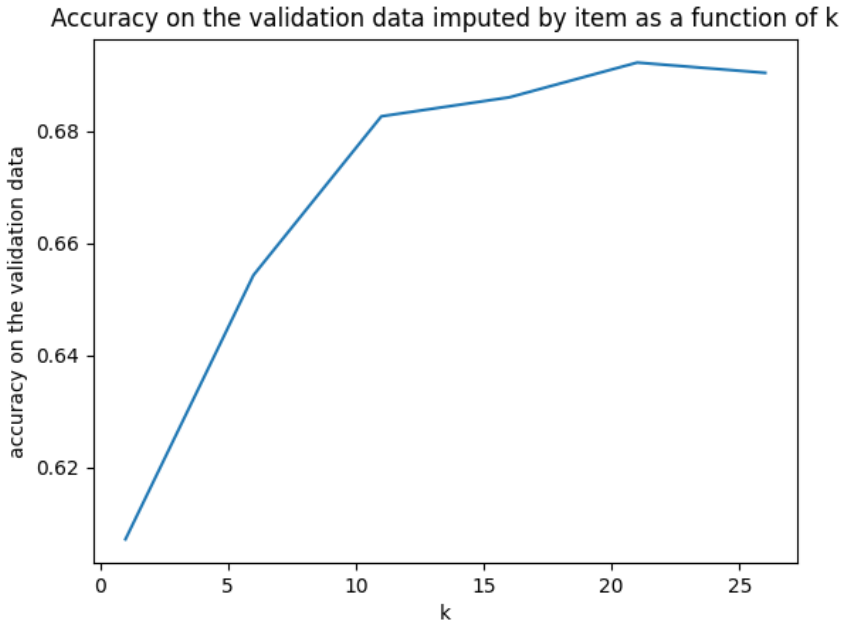
(a) The accuracy on the validation data with $k \in [1, 6, 11, 16, 21, 26]$ implemented by the user-based collaborative filtering:




```
For k = 1, Validation Accuracy on user-based collaborative filtering: 0.6244707874682472
For k = 6, Validation Accuracy on user-based collaborative filtering: 0.6780976573525261
For k = 11, Validation Accuracy on user-based collaborative filtering: 0.6895286480383855
For k = 16, Validation Accuracy on user-based collaborative filtering: 0.6755574372001129
For k = 21, Validation Accuracy on user-based collaborative filtering: 0.6692068868190799
For k = 26, Validation Accuracy on user-based collaborative filtering: 0.6522720858029918
For k = 11, Validation Accuracy on user-based collaborative filtering: 0.6841659610499576
The final test accuracy when k_star = 11 on user-based collaborative filtering is 0.6841659610499576
```

(b) Since $k = 11$ has the highest performance on validation data, $k^* = 11$ is chosen and the test accuracy is 0.6841659610499576.

(c) The accuracy on the validation data with $k \in [1, 6, 11, 16, 21, 26]$ implemented by the item-based collaborative filtering:




```
For k = 1, Validation Accuracy on item-based collaborative filtering: 0.607112616426757
For k = 6, Validation Accuracy on item-based collaborative filtering: 0.6542478125882021
For k = 11, Validation Accuracy on item-based collaborative filtering: 0.6826136042901496
For k = 16, Validation Accuracy on item-based collaborative filtering: 0.6860005644933672
For k = 21, Validation Accuracy on item-based collaborative filtering: 0.6922099915325995
For k = 26, Validation Accuracy on item-based collaborative filtering: 0.69037538808919
For k = 21, Validation Accuracy on item-based collaborative filtering: 0.6816257408975445
The final test accuracy when k_star = 21 on item-based collaborative filtering is 0.6816257408975445
```

Since $k = 21$ has the highest performance on validation data, $k^* = 21$ is chosen and the test accuracy is 0.6816257408975445.

(d) Based on the test accuracy, user-based collaborative filtering is slightly better than item-based collaborative filtering. Also, the computational time of user-based is much lower than that of item based. Hence, user-based performed better.

(e)
- Curse of Dimensionality: In high dimension, most points are approximately the same distance. KNN will not be feasible as we are not able to compare the distances between each data.
- Computational Cost: KNN is computationally expensive on large data set and large k value.

■

2 [15pts] Item Response Theory

Solution.

(a) The log-likelihood for all students and questions is derrived as follows:

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

where n is the number of students, m is the number of questions, and $\boldsymbol{C}$ is the sparse matrix which indicates question j is correctly answered by student i.

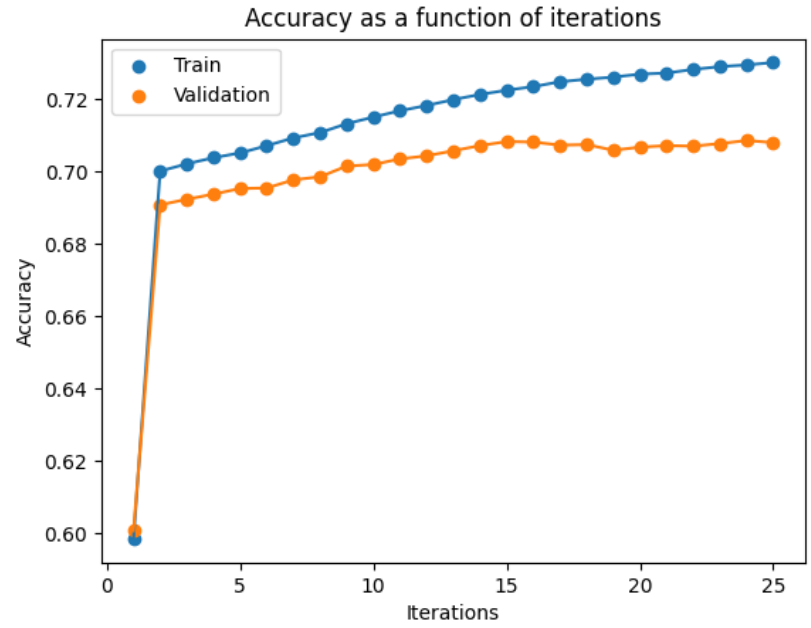
Derivative of the log-likelihood with respect to θ_i :

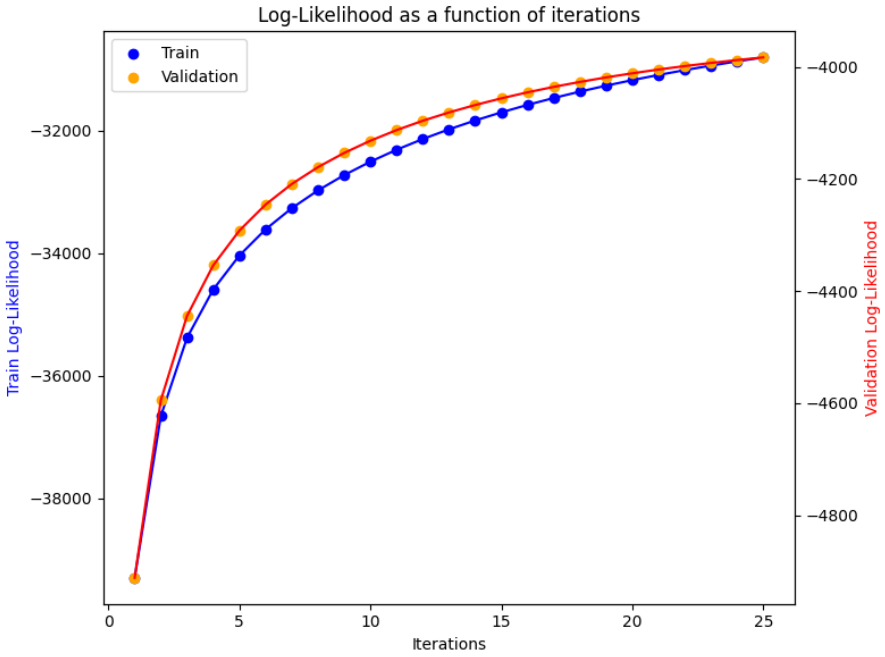
$$\begin{aligned}
\frac{\partial \log p(\boldsymbol{C}|\boldsymbol{\theta}, \boldsymbol{\beta})}{\partial \theta_i} &= \frac{\partial}{\partial \theta_i} \sum_{i=1}^{n} \sum_{j=1}^{m} c_{ij}(\theta_i - \beta_j) - \log \left(1 + \exp(\theta_i - \beta_j) \right) \\
&= \sum_{j=1}^{m} c_{ij} - \frac{\exp(\theta_i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

Derivative of the log-likelihood with respect to β_j :

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

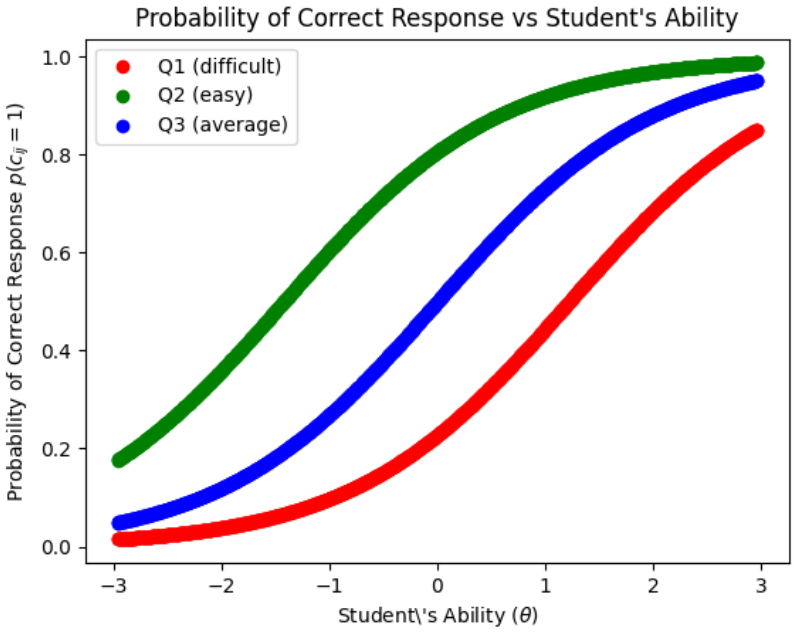
(b) To optimize our model with gradient decent, we choose learning rate = 0.005 and iteration = 25 as our hyperparameters.





(c) The final validation accuracy is 0.7082980524978831
The final test accuracy is 0.7011007620660458

(d) Let j_i be the question with the highest difficulty, j_2 be the question with the lowest difficulty, and j_3 be the question with an average difficulty. The plot of the probability of correct response vs student's ability is shown as follows:



The shape of the curves approximately follows a sigmoid curve. The curve of the easy question has a steepest slope and reaches to a higher probability of correct response the fastest as θ increases. Conversely, the curve of the difficult question has the less steep slope and is the slowest the reach a higher probability of correct response as θ increases. The result could be concluded that the probability of correct response increases has the students ability increases, while the rate of increase in probability is affected by the difficulty of the question, the easier the question, the higher the probability of receiving a correct response with the same ability.

■

3 [15pts] Neural Networks

Solution.

(a) Three differences between ALS and nueral networks:
- Purpose
ALS: Recommending items based on user preferences.
Neural Networks: Solving diverse tasks like image recognition and language processing.

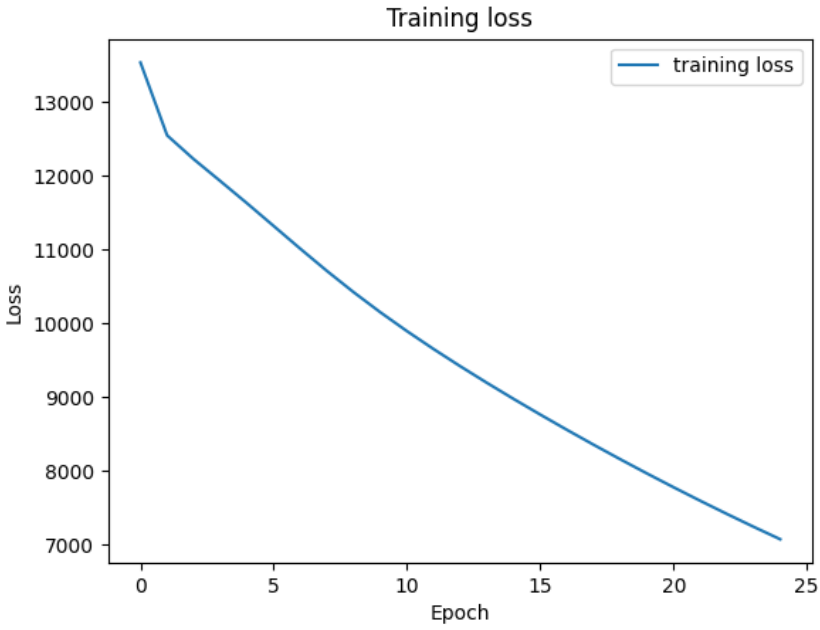
- Complexity
ALS: Simplified matrix factorization
Neural Networks: Multilayered, complex pattern learner.

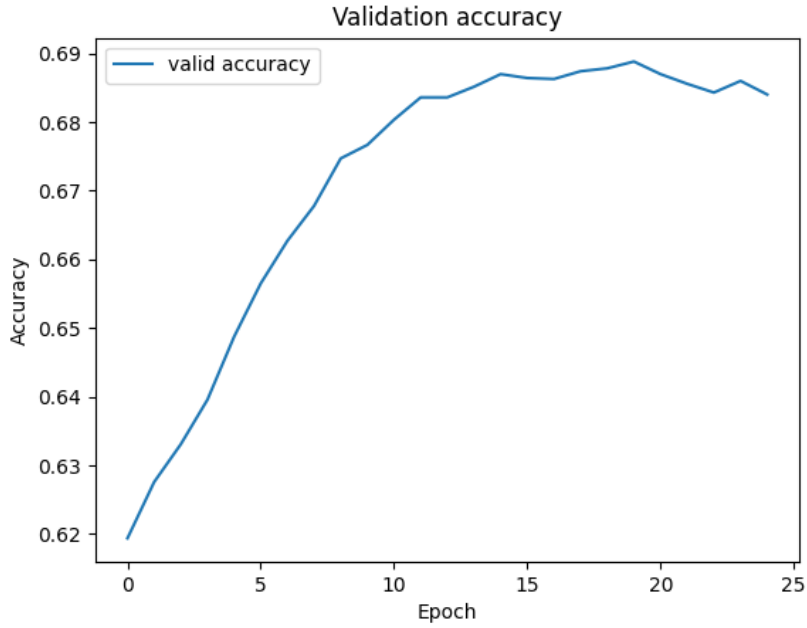
- Training Process
ALS: Update parameters alternately by fixing one and updating another.
Neural Networks: Update all parameters at once through back propagation.

(c) After testing all the combinations of hyperparameters, we discovered that the model starts to overfit when epoch is greater than 50. We also notice that different value of k cause the validation accuracy to converge at different rates, typically a larger value of k requires a larger number of epoch to reach convergence in validation accuracy, Thus, the model is optimized when learning rate = 0.02 and number of epoch = 25. Also, $k^* = 50$ is selected as it returns the highest validation accuracy

```
k: 10, learning rate: 0.005, epoch: 25, Validation Accuracy: 0.6364662715213096
k: 10, learning rate: 0.005, epoch: 50, Validation Accuracy: 0.6518487157775896
k: 10, learning rate: 0.001, epoch: 25, Validation Accuracy: 0.5750776178379904
k: 10, learning rate: 0.001, epoch: 50, Validation Accuracy: 0.6089472198701665
k: 10, learning rate: 0.02, epoch: 25, Validation Accuracy: 0.6799322607959356
k: 10, learning rate: 0.02, epoch: 50, Validation Accuracy: 0.6827547276319503
k: 50, learning rate: 0.005, epoch: 25, Validation Accuracy: 0.6521309624611911
k: 50, learning rate: 0.005, epoch: 50, Validation Accuracy: 0.6827547276319503
k: 50, learning rate: 0.001, epoch: 25, Validation Accuracy: 0.6233418007338414
...
k: 500, learning rate: 0.001, epoch: 50, Validation Accuracy: 0.6323736946090883
k: 500, learning rate: 0.02, epoch: 25, Validation Accuracy: 0.683460344340954
k: 500, learning rate: 0.02, epoch: 50, Validation Accuracy: 0.6646909398814564
Optimal k*: 50
Optimal learning rate: 0.02
Optimal epoch: 25
Optimal validation accuracy: 0.6883996613039797
```

(d) Selecting $k^* = 50$ and the model previously trained, the changes in training loss and validation accuracy as a function of epoch are as follows:





Final Test Accuracy: 0.6816257408975445

(e) After tuning the regularization penalty λ, $\lambda = 0.01$ is selected.

center The test accuracy after applying the regularization penalty on the model decreased slightly, thus, the model actually preforms better when the regularization penalty is not applied.

■

4 [15pts] Ensemble

Solution.

We choose Item Response Theory to implement the bagging ensemble and trained 3 base models with bootstrapping on the training set, where each new generated dataset has the same number of samples randomly chosen from the training set with replacements. Then we compute the average prediction of the 3 models with the validation set, the threshold of the prediction is set to be 0.5 as we are dealing with a binary classification problem. At last, we compute the final validation accuracy and test accuracy with the trained model.

The training result with hyperparameter: iteration $= 20$ and lr $= 0.005$ is shown as follows:

```
IRT Model 1 training:
Iteration: 1     Training log-likelihood: 39293.127371531205     Validation Accuracy: 0.6007620660457239
Iteration: 2     Training log-likelihood: 36534.33454883913      Validation Accuracy: 0.6883996613039797
...
Iteration: 10    Training log-likelihood: 32012.189844347264     Validation Accuracy: 0.6974315551792266

IRT Model 2 training:
Iteration: 1     Training log-likelihood: 39293.127371531205     Validation Accuracy: 0.6007620660457239
Iteration: 2     Training log-likelihood: 36472.53363359899      Validation Accuracy: 0.6840248377081569
...
Iteration: 10    Training log-likelihood: 31533.035245115694     Validation Accuracy: 0.6910810047981937

IRT Model 3 training:
Iteration: 1     Training log-likelihood: 39293.127371531205     Validation Accuracy: 0.6007620660457239
Iteration: 2     Training log-likelihood: 36343.966640679406     Validation Accuracy: 0.6836014676827548
...
Iteration: 10    Training log-likelihood: 31057.123763730055     Validation Accuracy: 0.6879762912785775

Final Validation Accuracy after ensemble: 0.6939034716342083
Test Accuracy after ensemble: 0.6943268416596104
```

The final validation and test accuracy of the ensemble model are both lower than that of the original model. Based on the result, it is suggested that the decrease in performance may be attributed to the insufficiency of models as only 3 base models are used for ensemble. However, for the ensemble model, the difference between the validation and test set is smaller than that of the original model, this suggest that the stability on prediction is improved after ensemble. ■

5 Part B:

5.1 Students' Ability Feature Enrichment

Motivation: Given the task done in Part A Question 2, a vector of students' ability, θ, is provided. In reality, if a student has higher ability, they should have a higher probability of answering a question correctly, vice versa

Method Description: We achieve the enrichment in the feature space of our input data by concatenating θ to the input vector. This can provide the neural network with more detailed and nuanced information about the input, the network can learn to leverage this information to make more informed predictions, allowing it to capture more complex patterns in the data, resulting a higher accuracy in predicting the correctness of students' answers.

Hypothesis: As students' ability and the correctness of students' answer might be related, we suspect that including the information of students' ability in the input data might be beneficial to the overall performance, and both the training accuracy and validation accuracy will increase compared with the base model.

Algorithm Box:

1. Concatenate the original input data with the students' ability tensor according to the user ID.
2. Increase the dimension of the linear function g.
3. Train the model with the new input data.
4. Tune the hyper-parameters and evaluate the model.

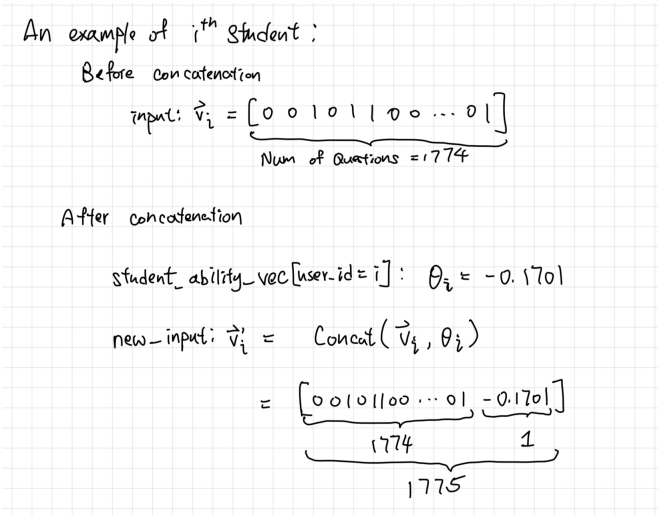
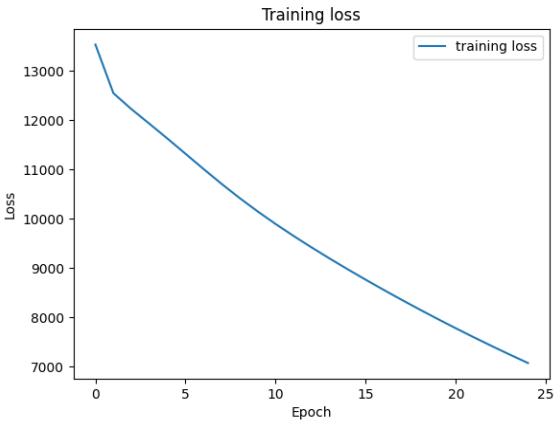
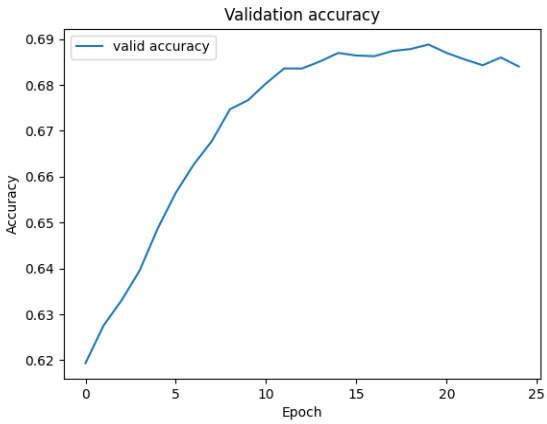


Figure 1: Illustration of Students' Ability Feature Enrichment

Performance Comparison: After tunning the hyperparameters, the comparison of model accuracy between the base model and the model with student ability feature enrichment are showed in table 1(p.12).

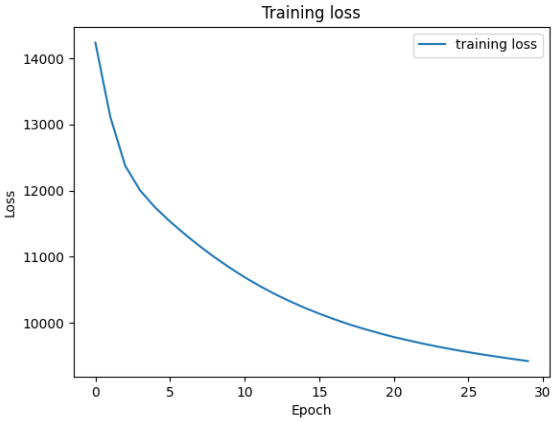


(a) Base Model training loss for each epoch

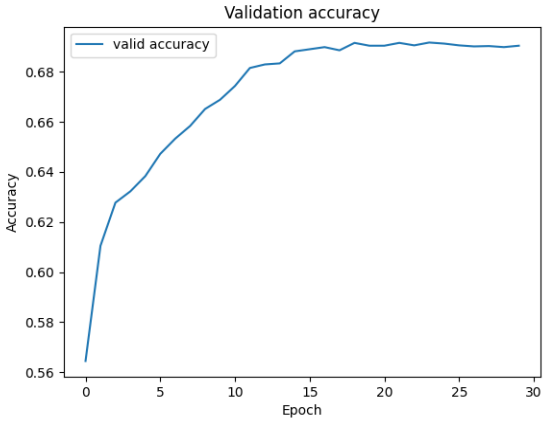


(b) Base Model validation accuracy for each epoch

Figure 2: performance of Base Model



(a) Student Ability Model training loss for each epoch



(b) Student Ability Model validation accuracy for each epoch

Figure 3: performance of Student Ability Model

We can observe that there is increase in both validation accuracy and test accuracy for the Student Ability Model, which supports our original hypothesis.
Comparing the graphs of the performance for both models(figure 2 for Base Model, figure 3 for Student Ability Model), we can see the training loss for the Student Ability Model decreases at a higher rate than that for the Base Model. The validation accuracy for the Student Ability Model also has a slight increase per epoch. Therefore, we can conclude the Student Ability Model is better in terms of both training speed and accuracy.

5.2 Binary Cross Entropy Loss

Motivation: We attempted to further improve the Student Ability Model using Binary Cross Entropy Loss. Binary Cross Entropy Loss is specifically designed for binary classification problems where the output is a probability score between 0 and 1, indicating the likelihood of belonging to one class. It's well-suited this problem where we are predicting the correctness of a single question with discrete class labels 0 and 1. Besides, Binary Cross Entropy Loss handles class imbalance better than the original Mean Squared Error Loss. It penalizes misclassification more heavily, making the model more sensitive to the minority class.

Method Description: We shift our minimization problem from the Mean Squared Error

$$\min_{\theta} \sum_{\mathbf{v} \in S} \|\mathbf{v} - f(\mathbf{v}; \theta)\|_2^2$$

to Binary Cross Entropy

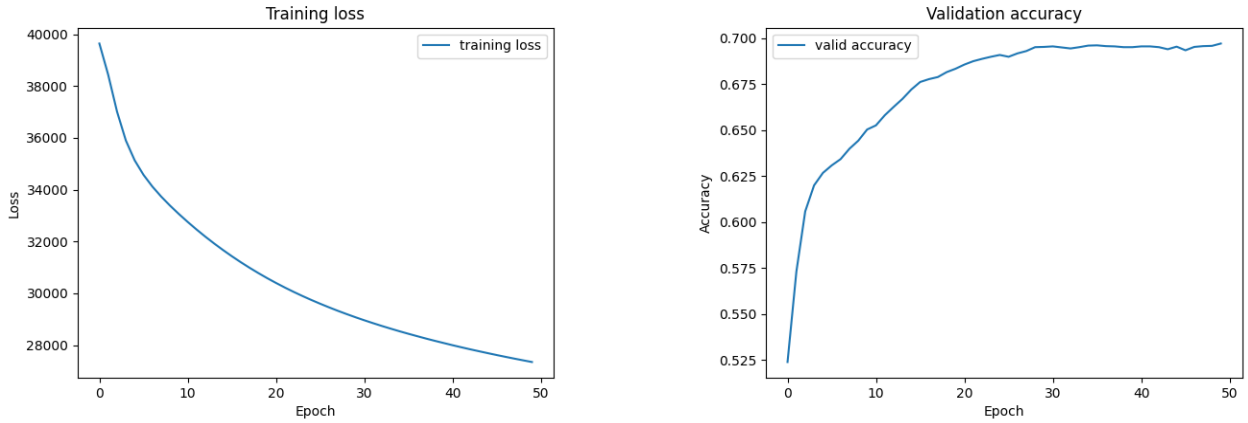
$$\min_{\theta} \sum_{\mathbf{v} \in S} -\mathbf{v} log(f(\mathbf{v}; \theta)) + (1 - \mathbf{v}) log(1 - f(\mathbf{v}; \theta))$$

Hypothesis: As Binary Cross Entropy is a better loss function for this problem, both the training accuracy and validation accuracy will increase compared with the base model.

Algorithm Box:

1. Change the loss function from MSE to BCE
3. Train the model with the new loss function.
4. Tune the hyper-parameters and evaluate the model.

Performance Comparison:



(a) Student Ability Model with BCE training loss for each epoch

(b) Student Ability Model with BCE validation accuracy for each epoch

Figure 4: performance of Student Ability Model with BCE

From table 1, We can observe that there is a slight increase in both validation accuracy and test accuracy for the Student Ability Model with BCE compared to the Student Ability Model with MSE, which supports our original hypothesis.
Comparing the graphs of the performance for both models(figure 3 for Student Ability Model with MSE, figure 4 for Student Ability Model with BCE), we can see the training loss for the Student Ability Model with BCE is higher than that with MSE. The validation accuracy for the Student Ability Model with BCE has a slight increase per epoch. Therefore, we can conclude the Student Ability Model with BCE is better in terms of accuracy but MSE is better in terms of training cost.

Table 1: Models Comparison

Model	**Hyperparameters**	**Validation Accuracy**	**Test Accuracy**
Base Model	$k^* = 50$ learning rate = 0.02 number of epoch = 25 lambda = 0.01	0.68163	0.68078
Base Model with Student Ability Feature Enrichment	$k^* = 10$ learning rate = 0.001 number of epoch = 30 lambda = 0.01	0.69250	0.69207
Base Model with Student Ability Feature Enrichment and Binary Cross Entropy	$k^* = 10$ learning rate = 0.0005 number of epoch = 50 lambda = 0.01	0.69870	0.69489

Limitations:

(a) Our models' predictions are only based on students' ability, whereas the meta data related to the questions, such as the subject of the questions, are not incorporated in the models. The introduction of the information of questions might possibly improve the accuracy of the neural network.

(b) Our models' prediction are only based on a single student, in other words, our models do not know other students' performance on the same or related question. Allowing the model to analysis other students' performance might improve our models if this can be implemented.

(c) The θ in the Student Ability Model is given by the IRT model, which might be a potential problem when there are dependencies between models.

(d) The computation cost is high which require long time for parameter tuning, which likely resulted in poorly tuned parameters.

6 Contributions:

Ian Quan: Part A
Douglas Quan: Part B