CSC311 Project

Yilin Li 996886473
Zheng Duan 1005727812
Yanbin Huang 1006345616

December 2023

1 Part A

1.1 k-Nearest Neighbor:

(a). Table 1 and Figure 1 shows the validation accuracy for $k \in \{1, 6, 11, 16, 21, 26\}$ by User-based KNN.

Table 1: Validation Accuracy by User

k	Validation(%)
1	62.45
6	67.81
11	68.95
16	67.56
21	66.92
26	65.23

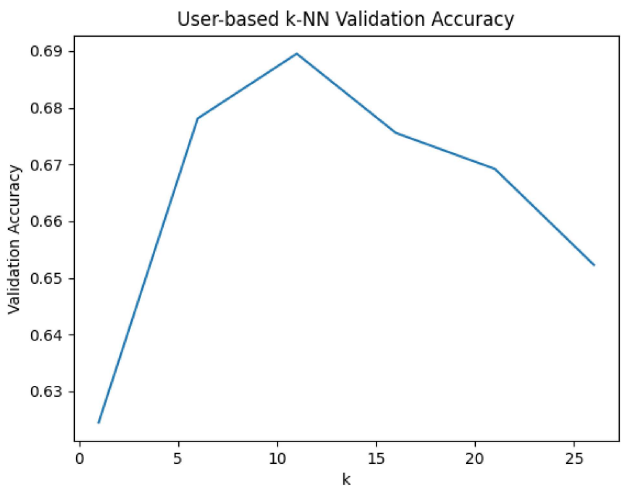

Figure 1: User-based k-NN Validation Accuracy

(b). **User-Based:** The k^* was 11, with a validation accuracy of approximately 68.95%. The test accuracy with this k^* was approximately 68.42%.

(c). Table 2 and Figure 2 shows the validation accuracy for $k \in \{1, 6, 11, 16, 21, 26\}$ by Item-based KNN.
Item-Based: The k^* was 21, with a validation accuracy of approximately 69.22%. The test accuracy with this k^* was approximately 68.16%.

Assumption: If two questions (items) have been answered similarly by various students, then they are similar to each other. This similarity is used to predict responses to one question based on the responses to its similar questions.

Table 2: Validation Accuracy by Item

k	Validation(%)
1	60.71
6	65.42
11	68.26
16	68.60
21	69.22
26	69.03

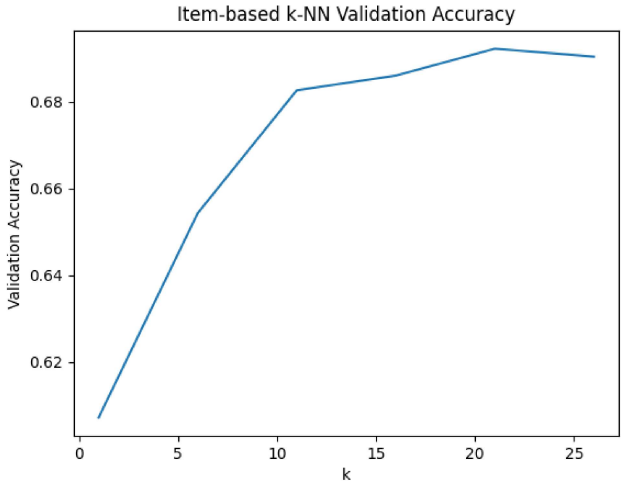

Figure 2: Item-based k-NN Validation Accuracy

(d). Comparison:
User-based collaborative filtering at k = 11 yields a test accuracy of approximately 68.42%.
Item-based collaborative filtering at k = 21 yields a test accuracy of approximately 68.16%.
From these results, we can conclude that the user-based collaborative filtering method performs slightly better than the item-based method for this particular dataset and set of k^* values, based on the test accuracy.

(e). Potential limitations:
Sensitivity to Noisy Data: kNN can be sensitive to noisy data or outliers. If some students' answers are not representative or contain errors, it can affect the accuracy of the predictions for other students.
Sparsity: Many students may not have answered all questions, leading to a sparse matrix. kNN might not perform well in such cases, as it relies on finding neighbors with similar response patterns.

1.2 Item Response Theory:

(a). Assuming that $\{c_{11}, ..., c_{nm}\}$ are $i.i.d.$, where n is the number of rows of the sparse matrix $\mathbf{C}$ and m is the number of columns of $\mathbf{C}$.
Given the formula of the probability that the question j is correctly answered by student i, we know that:

$$p(c_{ij}|\theta_i, \beta_j) = (\frac{exp(\theta_i - \beta_j)}{1 + exp(\theta_i - \beta_j)})^{c_{ij}} (\frac{1}{1 + exp(\theta_i - \beta_j)})^{1-c_{ij}}$$

We then have the probability of $\mathbf{C}$ given $\boldsymbol{\theta}, \boldsymbol{\beta}$ as follow:

$$p(\mathbf{C}|\boldsymbol{\theta}, \boldsymbol{\beta}) = \prod_{i=1}^{n} \prod_{j=1}^{m} p(c_{ij}|\theta_i, \beta_j) = \prod_{i=1}^{n} \prod_{j=1}^{m} (\frac{exp(\theta_i - \beta_j)}{1 + exp(\theta_i - \beta_j)})^{c_{ij}} (\frac{1}{1 + exp(\theta_i - \beta_j)})^{1-c_{ij}}$$

Then we can derive the log-likelihood $\log p(\mathbf{C}|\boldsymbol{\theta}, \boldsymbol{\beta})$:

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

The derivatives of the log-likelihood with respect to θ_i, β_j are then:

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

(b). We select the learning rate as 0.01 and the number of iterations as 20. Figure 3 shows the training and validation negative log-likelihoods as a function of iteration:

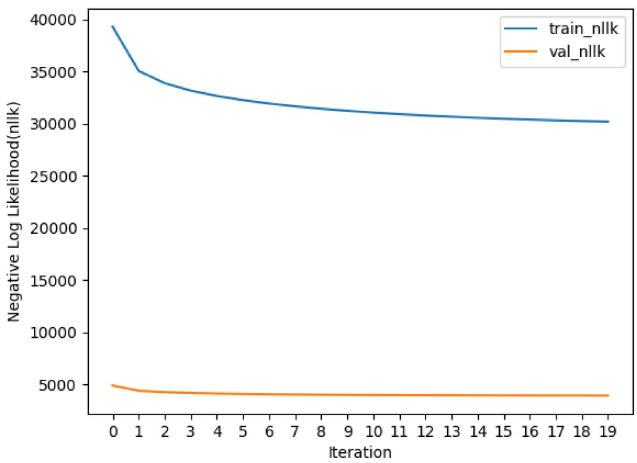

Figure 3: NLLK as a function of iteration

(c). Final validation accuracy is: 70.65%
Final test accuracy is: 70.51%

(d). For this question, we chose question 1, 10 and 100 as j_1, j_2 and j_3. The curves in Figure 4 show us how does the probability of correctly answer a question changes as the student's ability changes. Each curve implies that the probability of correctly answer a question is positively correlated with the ability of the student. Comparing these curves we can infer that there is a negative correlation between probability of correctly answer a question and the difficulty of the question.

1.3 Matrix Factorization:

(a). The set of k values we picked are [1, 2, 3, 4, 5, 10, 15, 20, 25, 30, 50, 100, 150, 200].
Table 3 records the SVD validation accuracy for each k.
Best SVD Performance: k=25, Test Accuracy = 65.57%

(b). SVD handles missing values by filling them with the average of each item. This can introduce bias into the data. The average may not accurately represent the preferences of users who didn't rate that item. This could lead to distorted recommendations or predictions.

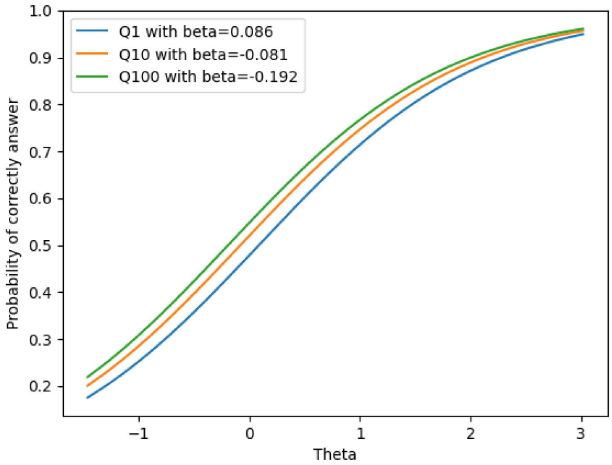

Figure 4: $P(c_{ij} = 1)$ as a function of abilities given question $j = \{1, 10, 100\}$

Table 3: SVD Validation Accuracy

k	Validation(%)	k	Validation(%)
1	64.28	20	65.40
2	65.79	25	65.95
3	65.83	30	65.48
4	65.78	50	64.84
5	65.90	100	64.71
10	65.86	150	64.30
15	65.65	200	63.27

In addition, filling in all missing values increases the density of the matrix, which can be problematic for large datasets. This can lead to increased computational complexity and memory requirements, making the SVD less scalable and efficient.

(d). The learning rate we used is 0.01, the number of iterations we used is 20, the set of k values we picked are [1, 2, 3, 4, 5, 10, 15, 20, 25, 30, 50, 100, 150, 200].
Table 4 records the ALS validation accuracy for each k.

Table 4: ALS Validation Accuracy

k	Validation(%)	k	Validation(%)
1	71.14	20	70.25
2	71.03	25	70.41
3	70.65	30	70.92
4	70.14	50	70.84
5	70.24	100	70.55
10	70.62	150	70.91
15	70.41	200	70.80

$k^* = 1$ achieves the highest validation accuracy, which is 71.14%.
$k^* = 4$ achieves the lowest validation accuracy, which is 70.14%.

(e). Let $k^* = 1$,
Best ALS Performance: Val Accuracy = 71.15%
Best ALS Performance: Test Accuracy = 70.45%
Figure 5 shows the Squared Error Loss Train-Validation Accuracy, and Figure 6 illustrates the Average Squared Error Loss Train-Validation Accuracy. Both figures are used 40 iterations for better illustration.

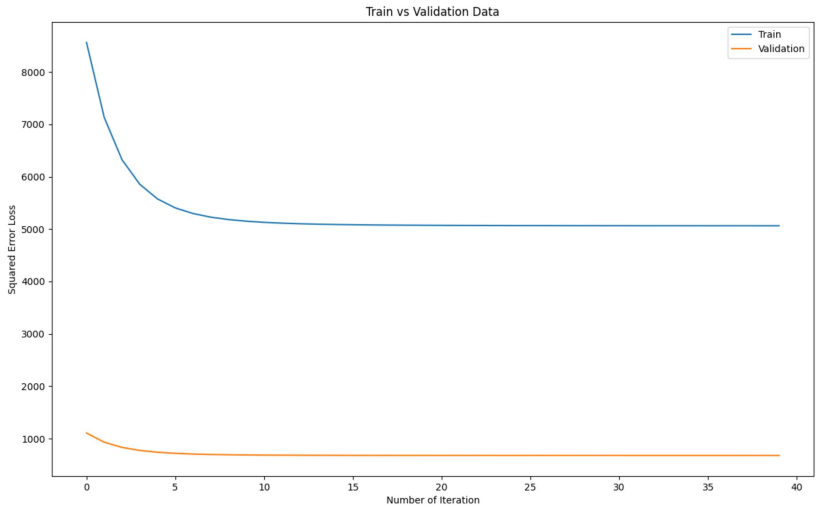

Figure 5: Squared Error Loss Train-Validation Accuracy

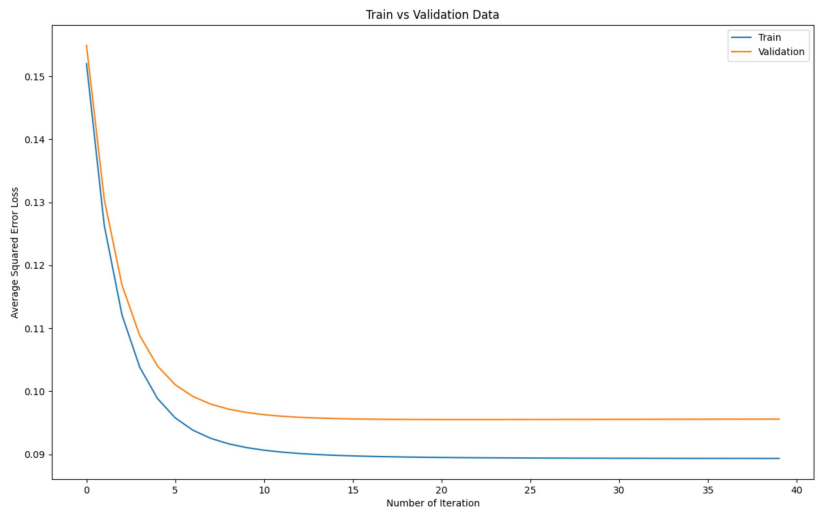

Figure 6: Average Squared Error Loss Train-Validation Accuracy

1.4 Ensemble:

Final Validation and Test Accuracy
The ensemble model, which combines KNN, ALS (Alternating Least Squares), and ITR (Item Response Theory), achieved the following accuracies:

Train Accuracy: 76.51%
Validation Accuracy: 70.83%
Test Accuracy: 70.73%

These results indicate that the ensemble model performs consistently across both validation and test datasets.

Explanation of the Ensemble Process
For each base model (KNN, ALS, ITR), a resampled version of the training data is created. This introduces variability among the models. Then we combines predictions from all three models. For each data point in the validation or test set, the ensemble model considers the predictions from each of the three models. A majority voting system is used, where the

final prediction is based on the majority vote of the three models.

Performance Assessment
Table 5 indicates the comparison of the ensemble model's performance with the individual models:

Table 5: Accuracies for Different Models

Model Type	Validation(%)	Test(%)
KNN by User	68.95	68.42
KNN by Item	69.22	68.16
Item Response Theory	70.65	70.51
SVD	65.95	65.57
ALS	71.15	70.45
Ensemble	70.83	70.73

The ensemble model's performance is competitive, but it does not outperform the best individual model (ALS) on the validation set. However, on the test set, its performance is very close to the best individual models (IRT and ALS).

Analysis
Correlation Among Resampled Sets: If the multiple datasets used for bagging are not independent, the expected variance reduction (by a factor of 1/m) may not be achieved. In our scenario, despite resampling with a different random state for each dataset, the inherent sparsity of data in the recommendation system and the limited availability of data could lead to high correlation among the resampled sets, minimally impacting variance reduction.

Underlying Bias in Base Models: all base models' accuracy hovering around 70% could indicate an underlying bias. Since bagging primarily addresses variance without reducing bias, this might explain why performance didn't improve notably in our situation.

Equal Weighting of Models: Bagging assigns equal weight to each model. In our case, even though all three base models have similar performance in terms of their test accuracy, they operate on different mechanisms. Therefore, a weighted ensemble approach might yield better results.

The ensemble model, which combines KNN, ALS, and ITR, shows good and consistent performance. It is slightly underperformed on the validation set compared to ALS, but it comes very close on the test data. This means the ensemble model is reliable and works well in different situations. It's a good choice because it balances the strengths and weaknesses of the individual models, making it a stable and versatile option for predictions.

2 Part B

For this part, all the figures can be found in "Appendix: Figures".

2.1 Problem with one-parameter IRT

In part A, our one-parameter Item Response Theory Model obtained a 70.51% test accuracy, which is relatively smaller compared to the high-eighties accuracies we've seen during our assignments and lectures. In the rest of this report, we will refer the one-parameter IRT Model as our base model. We believe that our base model is underfitting the data set. Figure 7 shows how the accuracies change on training and validation data sets during 400 iterations. Both training and validation accuracies converge to some small numbers that are below 75%. This figure supports our argument that the base model is underfitting instead of overfitting.
We then further argue that the source of the underfitting comes from the simplicity of the base model. Below are two reasons of why we think the base model is too simple:

1. In the base model, we assume that $P(c_{ij} = 1)$ is totally decided by the ability of the student and the difficulty of the problem. In fact, there's a lot of other factors which can influence the probability of a correct response. For example, since we are dealing with multiple-choices questions, one can always guess an answer. This means that even if a student's ability is very low, they should still have some chances to correctly answer difficult questions based on their instincts, luck, etc.

2. The base model assumes that both the abilities of students and the difficulties of questions are one-dimensional and thus can be evaluated a real number. However, this is no realistic as we can easily classify students' abilities or questions'

difficulties into different categories such as subjects and types of skills.Instead of being one-dimensional, the ability and the difficulty should be multi-dimensional. Each dimension should represent a specific subject or type of skills that is independent to others.

2.2 Model Selection

To address the underfitting problem, we want to increase the complexity of the base model by adding extra parameters. During our research, we found two models that best fit our interest. One is the 2-PL IRT model that we found on the online article "Item Response Theory"(Columbia University Mailman School of Public Health, n.d.). The other one is the 4-PL IRT model(Fu et al., 2021, p.429) where it not only considers the probability of guessing but also incorporates with the fact that student can make mistake and questions could have discrimination on students' abilities. However, because we are dealing with multiple-choice questions, we believe it is less likely that student will wrongly answer the question once they have the enough knowledge for a question. Also, we want to avoid introducing complexity that we can not properly explain, such as the discrimination parameter. Thus, our final decision is the 2-PL IRT model with some modifications, which has the following formula:

$$p(c_{ij} = 1|\theta_i, \beta_j, \delta_i) = \frac{exp(\delta_i)}{1 + exp(\delta_i)} + (1 - \frac{exp(\delta_i)}{1 + exp(\delta_i)})\frac{exp(\theta_i - \beta_j)}{1 + exp(\theta_i - \beta_j)} \tag{1}$$

In the formula, the probability of student i correctly answer question j is determined by the ability of student $i(\theta_i)$, the difficulty of question $j(\beta_j)$ and the guessing ability of student $i(\delta_i)$. In the original model, δ_i should represent the probability of correctly guessing. But that would be hard for us to optimize the parameters. Therefore, we apply a sigmoid function on δ_i and interpreted it as the ability of guessing. Notice that we did not increase the dimension of abilities and difficulties to incorporate reason 2 from the previous section. We will explain this in the "limitation" section later.

2.3 Method

Using the selected model, we want to train a discriminative classifier in the same way as we did in part A. To start, we first derive the log-likelihood of the model. Denote $exp(\theta_i - \beta_j)$ as k_{ij} and $\frac{exp(\delta_i)}{1+exp(\delta_i)}$ as A_i, given the formula (1), we know that:

$$p(c_{ij}|\theta_i, \beta_j, \delta_i) = (\frac{k_{ij} + A_i}{1 + k_{ij}})^{c_{ij}}(\frac{1 - A_i}{1 + k_{ij}})^{1-c_{ij}}$$

Assuming that $\{c_{11}, ..., c_{nm}\}$ are *i.i.d.*, where n is the number of rows of the sparse matrix $\mathbf{C}$ and m is the number of columns of $\mathbf{C}$. We then have the probability of $\mathbf{C}$ given $\boldsymbol{\theta}, \boldsymbol{\beta}, \boldsymbol{\delta}$ as follow:

$$p(\mathbf{C}|\boldsymbol{\theta}, \boldsymbol{\beta}, \boldsymbol{\delta}) = \prod_{i=1}^{n}\prod_{j=1}^{m} p(c_{ij}|\theta_i, \beta_j, \delta_i) = \prod_{i=1}^{n}\prod_{j=1}^{m}(\frac{k_{ij} + A_i}{1 + k_{ij}})^{c_{ij}}(\frac{1 - A_i}{1 + k_{ij}})^{1-c_{ij}}$$

Then we can derive the log-likelihood $\log p(\mathbf{C}|\boldsymbol{\theta}, \boldsymbol{\beta}, \boldsymbol{\delta})$:

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

The derivatives of the log-likelihood with respect to $\theta_i, \beta_j, \delta_i$ are then:

$$\frac{\partial \log p(\mathbf{C}|\boldsymbol{\theta}, \boldsymbol{\beta}, \boldsymbol{\delta})}{\partial \delta_i} = \sum_{j=1}^{m} \frac{c_{ij} exp(\delta_i)(1 + exp(\theta_i - \beta_j))}{exp(\theta_i - \beta_j)(1 + exp(\delta_i)) + exp(\delta_i)} - \frac{exp(\delta_i)}{1 + exp(\delta_i)}$$

$$\frac{\partial \log p(\mathbf{C}|\boldsymbol{\theta}, \boldsymbol{\beta}, \boldsymbol{\delta})}{\partial \theta_i} = \sum_{j=1}^{m} \frac{c_{ij} exp(\theta_i - \beta_j)(1 + exp(\delta_i))}{exp(\theta_i - \beta_j)(1 + exp(\delta_i)) + exp(\delta_i)} - \frac{exp(\theta_i - \beta_j)}{1 + exp(\theta_i - \beta_j)}$$

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

Using these derivatives, we can perform alternating gradient descent on $\boldsymbol{\theta}, \boldsymbol{\beta}$ and $\boldsymbol{\delta}$ to m aximize the log-likelihood.

2.4 Comparison & Demonstration

2.4.1 Comparison with Baseline Models

As a result, we selected 0.02 as the learning rate and 40 as the number of iterations. We then compare it with all the models that we implemented in part A. From table 6, we can see that our extended model obtain the best performance among our baseline models and has improved the test accuracy by 0.62% when compared to the base model. Figure 8 further compares the base model and the extended model on negative log-likelihood of both training and validation data.

Table 6: Extended Model compares to Baseline Models

Model Type	LR or k	Iteration	Validation(%)	Test(%)
1-PL IRT(Base)	0.01	20	70.65	70.51
2-PL IRT(Extended)	0.02	40	71.13	**71.13**
KNN by User	(11)	N/A	68.95	68.42
KNN by Item	(21)	N/A	69.22	68.16
SVD	(25)	N/A	65.95	65.57
ALS	0.01(1)	20	71.15	70.45
Ensemble	N/A	N/A	70.83	70.73

2.4.2 Demonstration with Experiments

In section 2.1, we've discussed that adding a guessing parameter should improve the model's performance on predicting the responses of students with poor abilities. Also, as complex questions are more difficult to guess, we expect that there are more improvement on the easy questions than on the hard questions. To test our hypothesis, we divide questions and students each into 20 groups according to their difficulties or abilities(with a non-decreasing order) and try to compare the accuracies of among these groups under our base and extended model.

Figure 9 and 10 show the result of this experiment. We can clearly see that the accuracies of students with poor abilities(the last 5% of the students) have been increased on both validation data and test data by our extended model. The accuracies on the easiest questions(the last 5% of the questions) have also been improved. Figure 11 and 12 clearly show the amount of improvements. The test accuracy on easy questions is increased by 1.04(%) and the test accuracy on students with poor abilities is increased by 1.46(%). These results are aligned with our hypothesis about how the new guessing parameter could affect our model's performance. However, it is surprised to us that the test accuracy on Hard Question is also increased by 1.72(%). To further explore this phenomenon, we've created Figure 13, which shows the relationship between δ_i and θ_i.

From Figure 13, we can see that delta is positively correlated to theta, which implies that students with higher abilities are also better at guessing. This may explain why we saw an improvement in Hard Questions predictions as the model now produces a higher probability for a smart student to correctly answer a hard question.

2.5 Limitations

1. In our model, We did not address the reason 2 discussed in section 2.1. This was because we didn't find an appropriate math model that connects the multi-dimensional β_j and θ_i to the probability of correctly response. Also all of the diagnostic questions that we are dealing with are math questions, so it is less worthy to do such extension. In a setting where we work on diagnostic questions across multiple disciplines, we can expect the performance of our model to be bad, as there could be multiple determinant latent trails. One possible way to address this is that we first classify the questions into different

categories, then for each category we train a specific IRT model for it.

2. Our model's performance is also heavily relied on the observed data. Newly joined users or students may have to do a large number of questions(probably 100-300 questions) before our model can properly estimate their abilities. To address situation like this, it is better to use method such as matrix factorization that requires less on the observed data.

3. The improvement we obtained by increasing the complexity of our model is relatively small. However, we introduced an unnecessary correlation between the guessing parameter and the ability parameter, as showed in Figure 9. This implies that the new parameter is not providing as many information as we thought but yet brings us high variance. We can of course use bagging to reduce the extra variance. But since the improvement was so little we can also abandon the guessing parameter.

4. Last but not least, since we are performing alternating gradient descent on our parameters, we need to go through the data set again and again. This could be very computational heavy as our database increase. To reduce the burden of computing, we can build a Naive Bayes Model, which only requires one pass through the training data set.

3 Contribution

3.1 Part A:

Yilin Li: Q1
Zheng Duan:Q3 & Q4
Yanbin Huang:Q2

3.2 Part B:

Yilin Li: Discussion, research and latex type up
Zheng Duan:Discussion, research and implementation of the code
Yanbin Huang:Discussion, research, implementation of the code and latex type up

4 Appendix: Figures

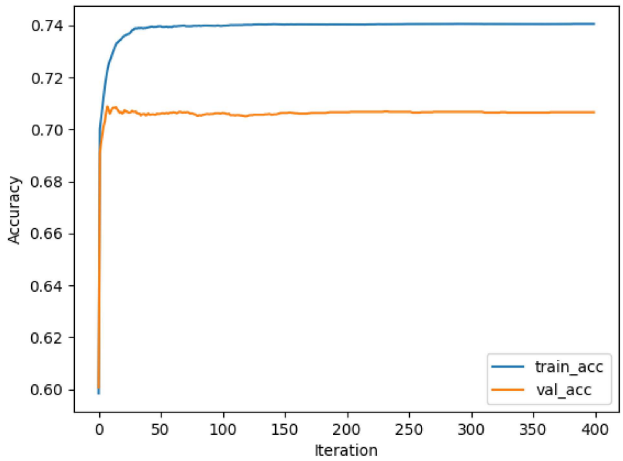

Figure 7: Training and validation accuracies for 500 iterations

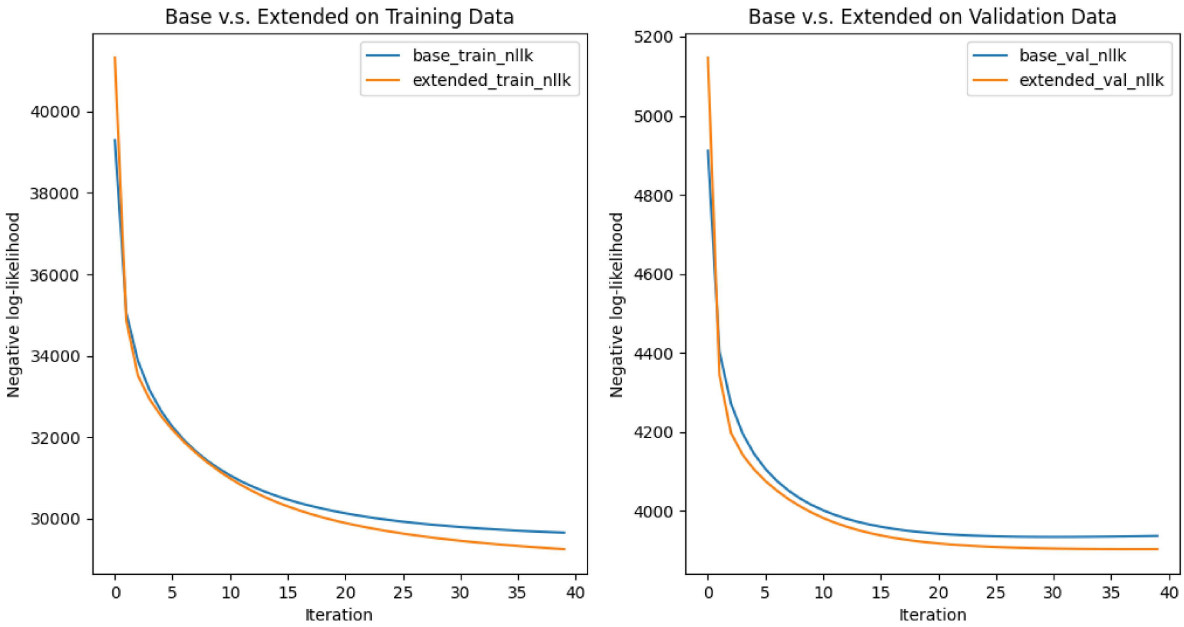

Figure 8: Base Model v.s. Extended Model on NLLK

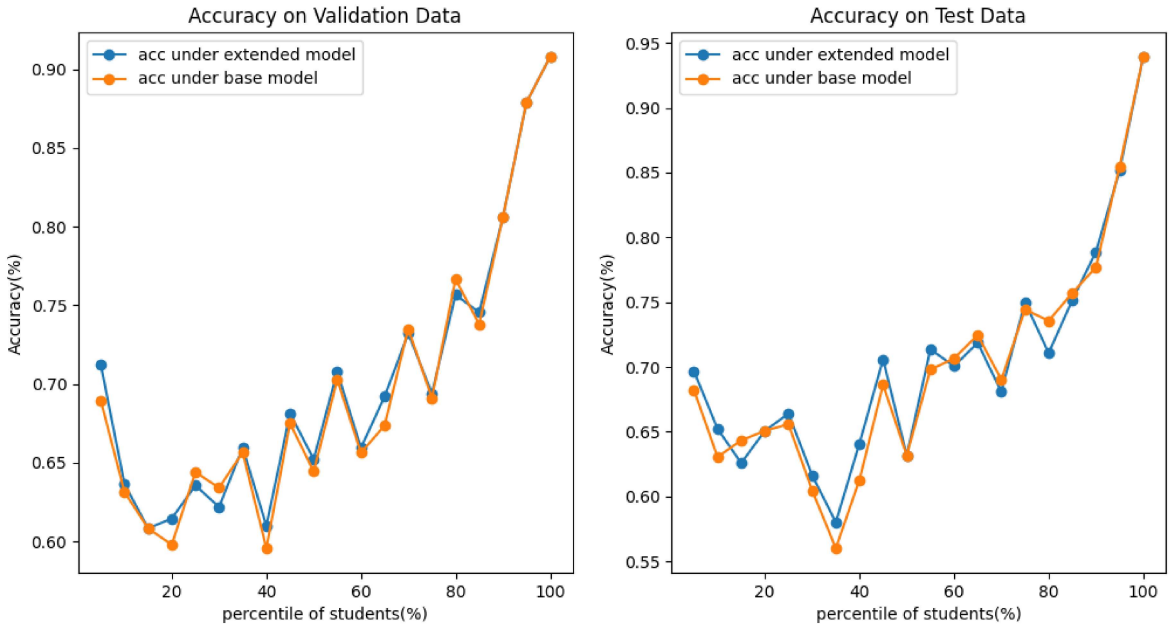

Figure 9: Accuracies comparison among different groups of students

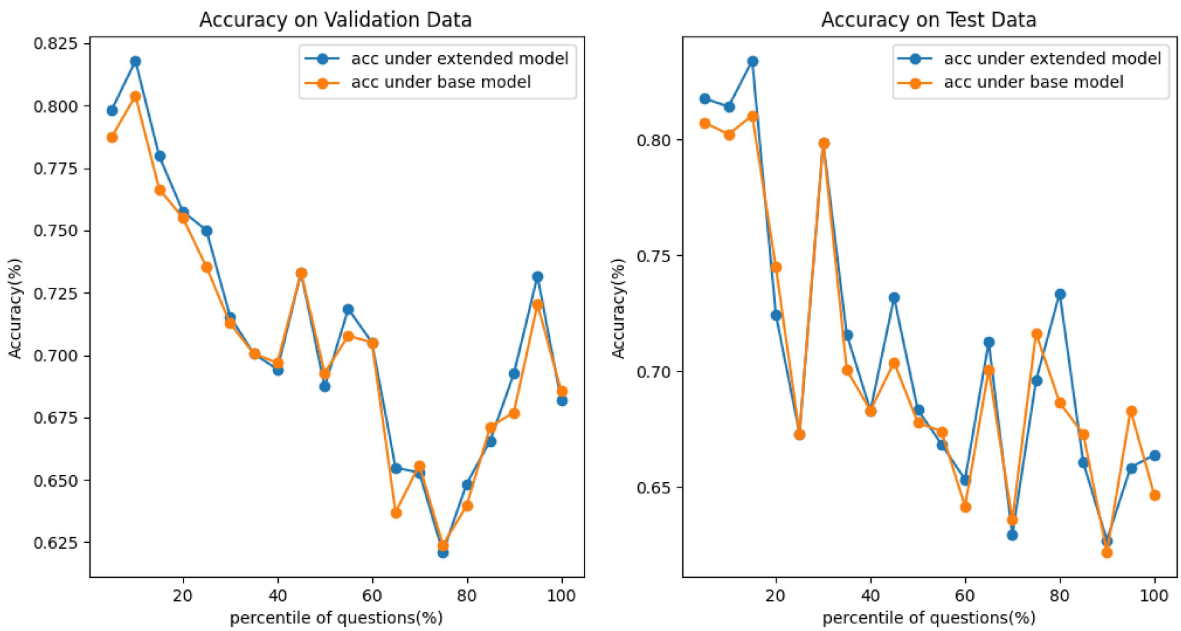

Figure 10: Accuracies comparison among different groups of questions

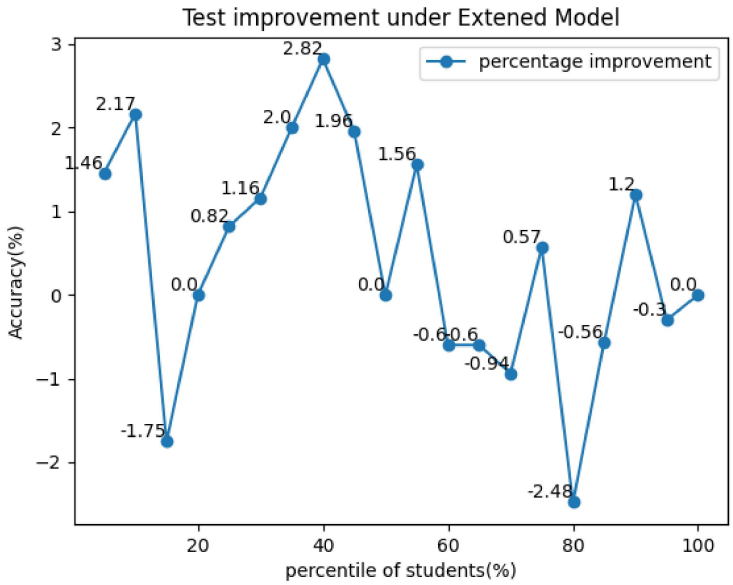

Figure 11: Accuracies improvement on different groups of students

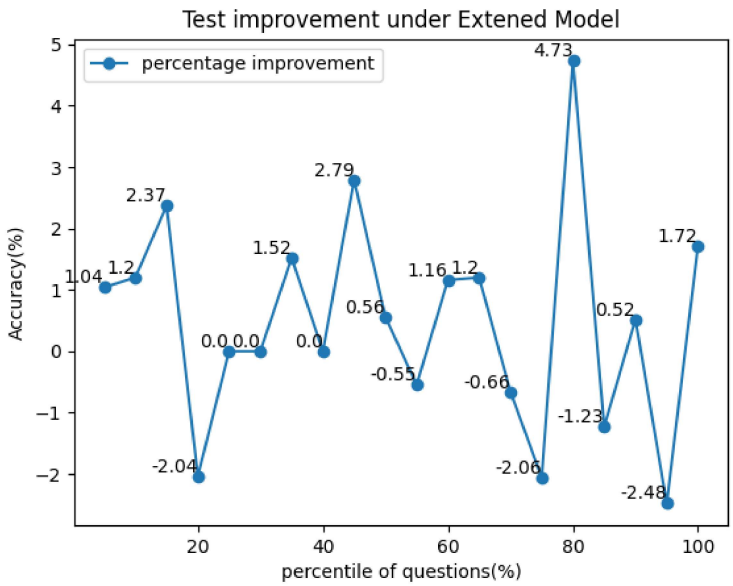

Figure 12: Accuracies improvement on different groups of questions

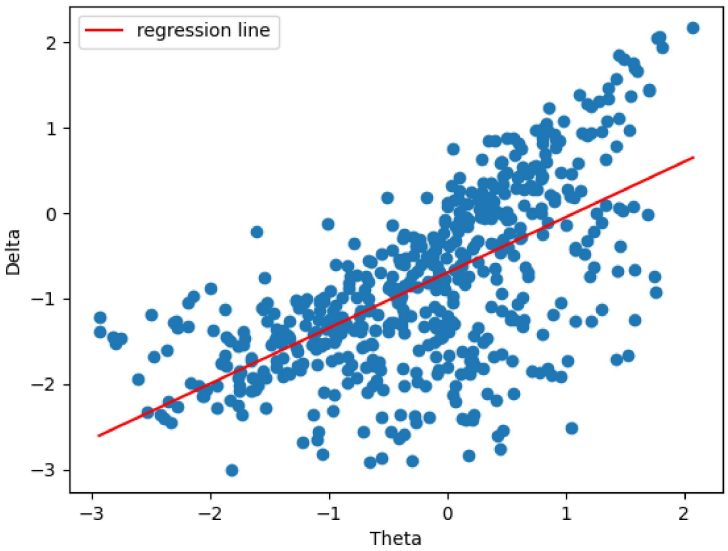

Figure 13: Delta positively correlates to Theta, with high variance

5 References

1. Item response theory.(n.d.).Columbia University Mailman School of Public Health. Retrieved December 3rd, 2023 from https://www.publichealth.columbia.edu/research/population-health-methods/item-response-theory

2. Fu, Z., Zhang, S., Su, Y.-H., Shi, N. and Tao, J. (2021), A Gibbs sampler for the multidimensional four-parameter logistic item response model via a data augmentation scheme. Br J Math Stat Psychol, 74: 427-464. https://doi.org/10.1111/bmsp.12234