CSC311 Final Project

December 3, 2021

Part A

1. k-Nearest Neighbour

a. When k = 1:
Validation Accuracy: 0.6244707874682472

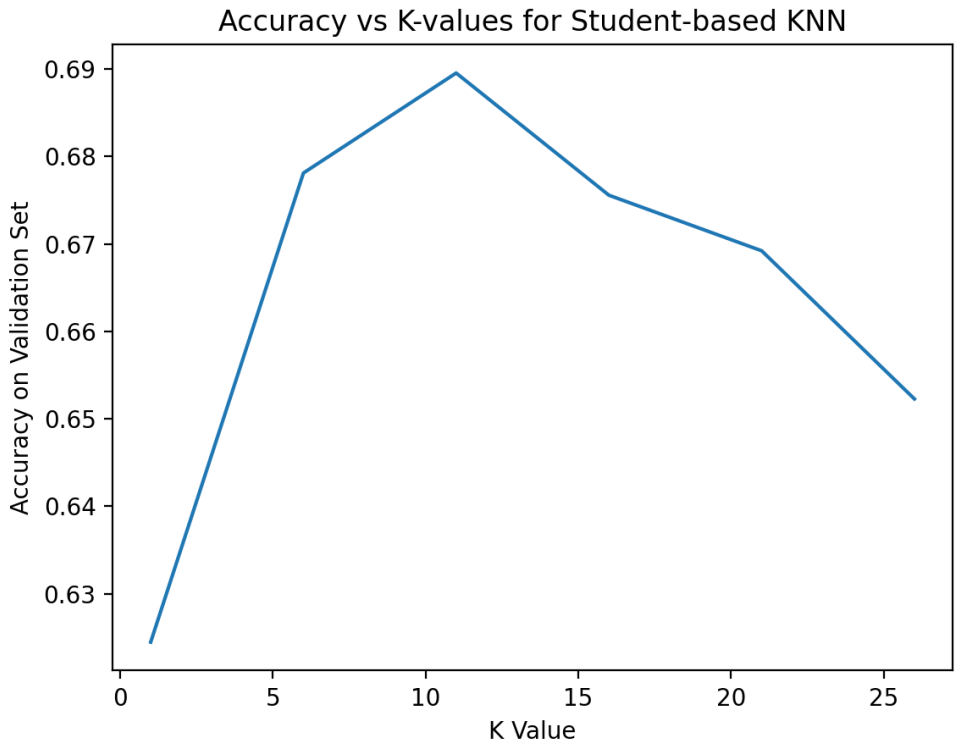
When k = 6:
Validation Accuracy: 0.6780976573525261

When k = 11:
Validation Accuracy: 0.6895286480383855

When k = 16:
Validation Accuracy: 0.6755574372001129

When k = 21:
Validation Accuracy: 0.6692068868190799

When k = 26:
Validation Accuracy: 0.6522720858029918



b. The optimal k is 11 and has an accuracy of 0.6841659610499576 on the test dataset.
c. The underlying assumption for itemized collaborative filtering is that if question A was answered correctly or incorrectly by the same set of users as question B, then question A's correctness for other users is the same as that of question B.

   When k = 1:
   Validation Accuracy: 0.607112616426757

   When k = 6:
   Validation Accuracy: 0.6542478125882021

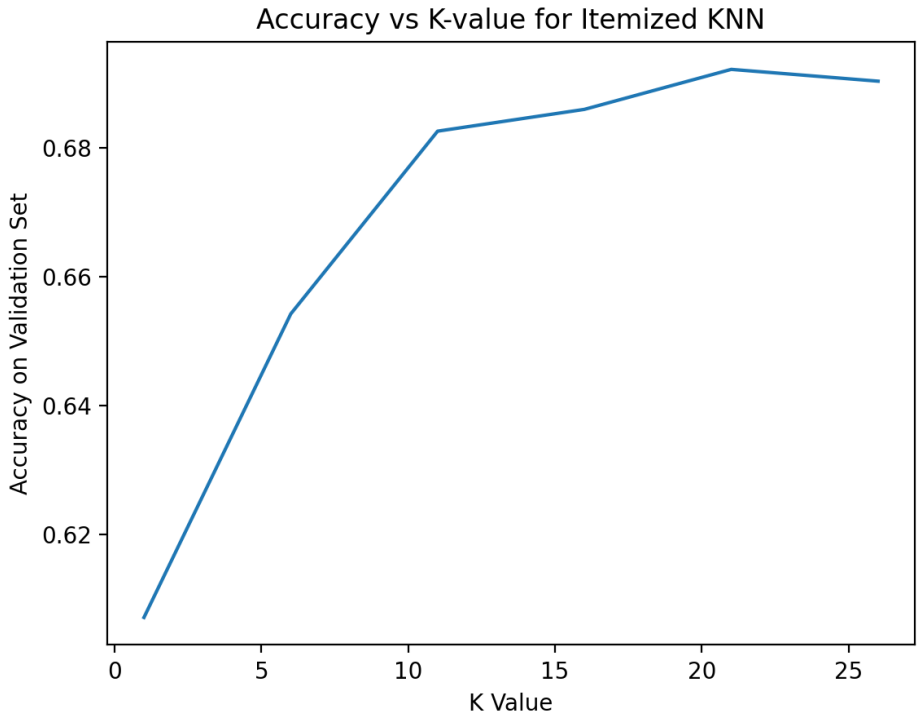
   When k = 11:
   Validation Accuracy: 0.6826136042901496

   When k = 16:
   Validation Accuracy: 0.6860005644933672

   When k = 21:
   Validation Accuracy: 0.6922099915325995

   When k = 26:
   Validation Accuracy: 0.69037538808919

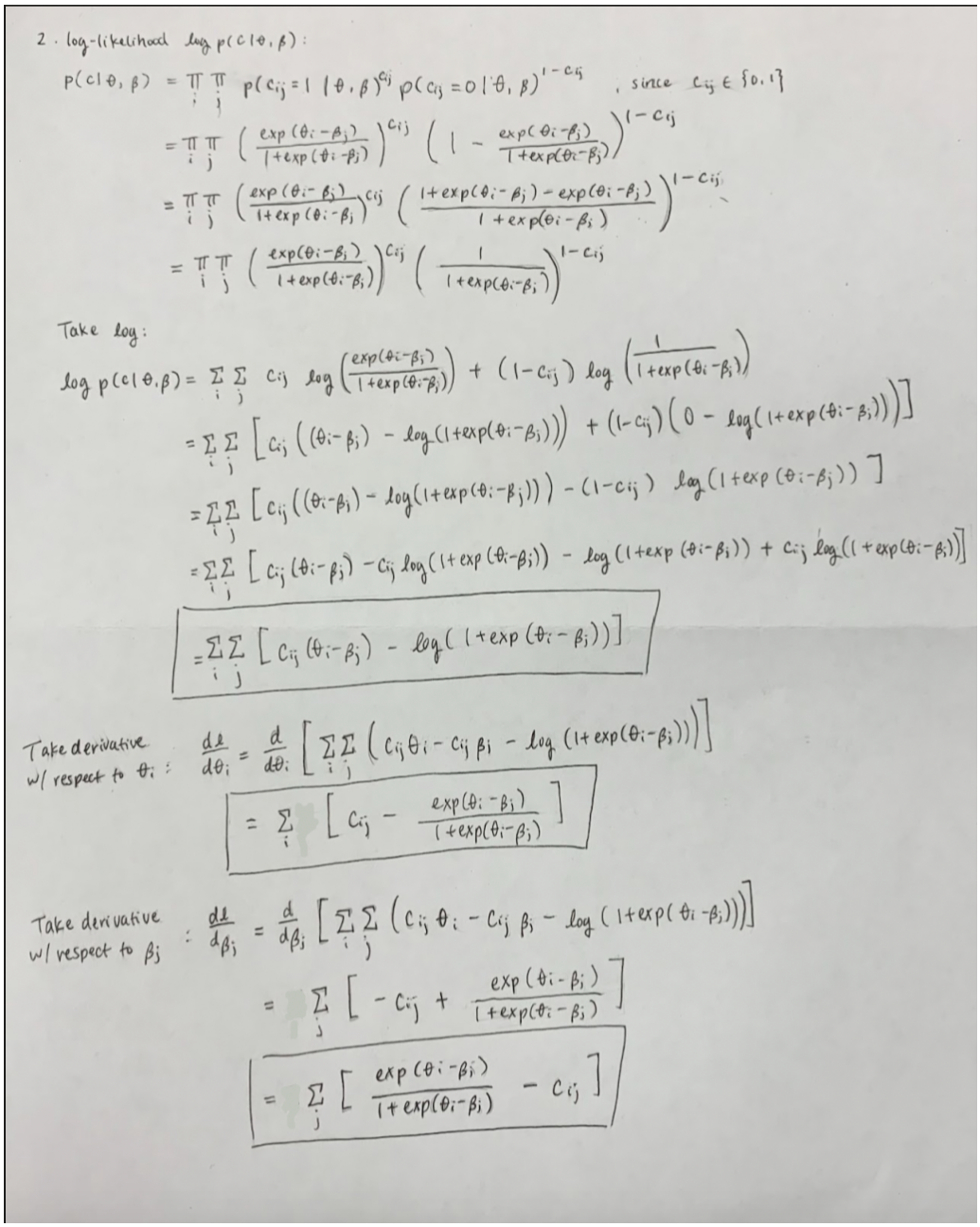
   The optimal k is 21 and has an accuracy of 0.6683601467682755 on the test dataset.



d. In terms of performance, user-based collaborative filtering performs the best on the test dataset.
e. Limitation:
    i. KNN for diagnostic prediction is prone to popularity-bias. This means that the model is prone to predict answers to questions based on what is most commonly answered.
    ii. Isn't able to handle new items fed to the model. Because the matrix is sparse, many items for a (user, item) pair may not be seen during training and thus it fails to be able to predict its correctness properly.

2. Item Response Theory

a)



2. log-likelihood $\log p(c|\theta,\beta)$:

$$p(c|\theta,\beta) = \prod_i \prod_j p(c_{ij}=1|\theta,\beta)^{c_{ij}} p(c_{ij}=0|\theta,\beta)^{1-c_{ij}}, \text{ since } c_{ij} \in \{0,1\}$$

$$= \prod_i \prod_j \left(\frac{\exp(\theta_i-\beta_j)}{1+\exp(\theta_i-\beta_j)}\right)^{c_{ij}} \left(1 - \frac{\exp(\theta_i-\beta_j)}{1+\exp(\theta_i-\beta_j)}\right)^{1-c_{ij}}$$

$$= \prod_i \prod_j \left(\frac{\exp(\theta_i-\beta_j)}{1+\exp(\theta_i-\beta_j)}\right)^{c_{ij}} \left(\frac{1+\exp(\theta_i-\beta_j)-\exp(\theta_i-\beta_j)}{1+\exp(\theta_i-\beta_j)}\right)^{1-c_{ij}}$$

$$= \prod_i \prod_j \left(\frac{\exp(\theta_i-\beta_j)}{1+\exp(\theta_i-\beta_j)}\right)^{c_{ij}} \left(\frac{1}{1+\exp(\theta_i-\beta_j)}\right)^{1-c_{ij}}$$

Take log:

$$\log p(c|\theta,\beta) = \sum_i \sum_j c_{ij} \log\left(\frac{\exp(\theta_i-\beta_j)}{1+\exp(\theta_i-\beta_j)}\right) + (1-c_{ij}) \log\left(\frac{1}{1+\exp(\theta_i-\beta_j)}\right)$$

$$= \sum_i \sum_j \left[c_{ij}\left((\theta_i-\beta_j) - \log(1+\exp(\theta_i-\beta_j))\right) + (1-c_{ij})\left(0 - \log(1+\exp(\theta_i-\beta_j))\right)\right]$$

$$= \sum_i \sum_j \left[c_{ij}\left((\theta_i-\beta_j) - \log(1+\exp(\theta_i-\beta_j))\right) - (1-c_{ij}) \log(1+\exp(\theta_i-\beta_j))\right]$$

$$= \sum_i \sum_j \left[c_{ij}(\theta_i-\beta_j) - c_{ij}\log(1+\exp(\theta_i-\beta_j)) - \log(1+\exp(\theta_i-\beta_j)) + c_{ij}\log(1+\exp(\theta_i-\beta_j))\right]$$

$$= \sum_i \sum_j \left[c_{ij}(\theta_i-\beta_j) - \log(1+\exp(\theta_i-\beta_j))\right]$$

Take derivative w/ respect to θ_i:

$$\frac{dl}{d\theta_i} = \frac{d}{d\theta_i}\left[\sum_i \sum_j \left(c_{ij}\theta_i - c_{ij}\beta_j - \log(1+\exp(\theta_i-\beta_j))\right)\right]$$

$$= \sum_j \left[c_{ij} - \frac{\exp(\theta_i-\beta_j)}{1+\exp(\theta_i-\beta_j)}\right]$$

Take derivative w/ respect to β_j:

$$\frac{dl}{d\beta_j} = \frac{d}{d\beta_j}\left[\sum_i \sum_j \left(c_{ij}\theta_i - c_{ij}\beta_j - \log(1+\exp(\theta_i-\beta_j))\right)\right]$$

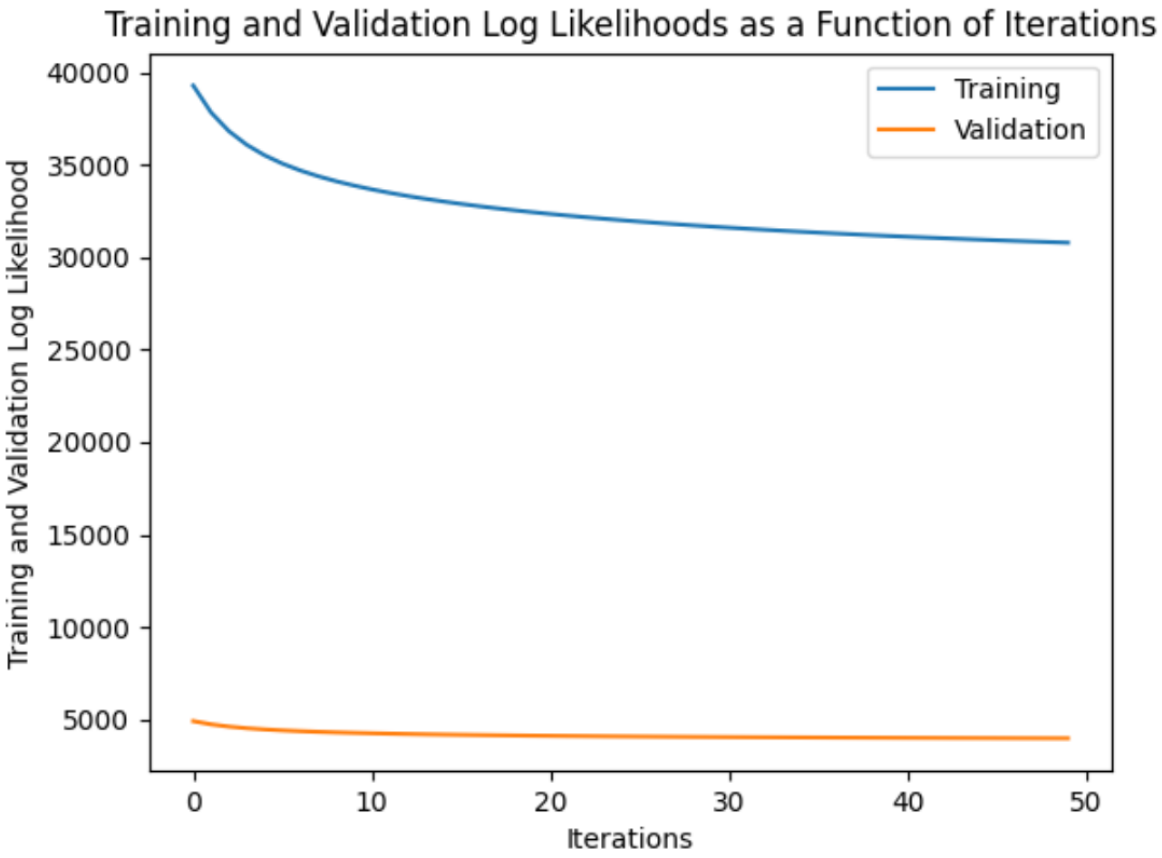
$$= \sum_i \left[-c_{ij} + \frac{\exp(\theta_i-\beta_j)}{1+\exp(\theta_i-\beta_j)}\right]$$

$$= \sum_i \left[\frac{\exp(\theta_i-\beta_j)}{1+\exp(\theta_i-\beta_j)} - c_{ij}\right]$$

b) These hyperparameters were selected as the optimal ones after comparing results by using various learning rates between 0.01 and 0.001 and iterations between 10 and 100

- Learning rate = 0.0025
- Iterations = 50

See plot below for the training curve that shows the training and validation log-likelihoods as a function of iteration.



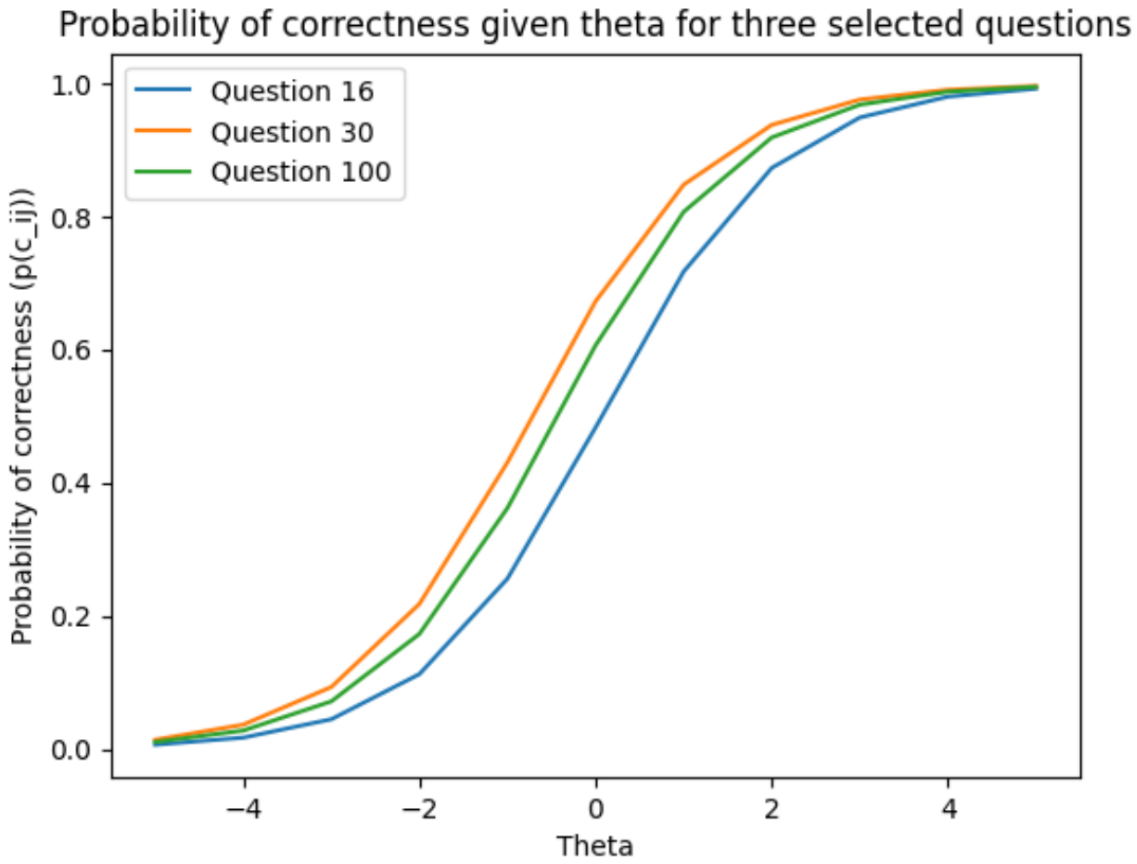
c) Final validation accuracy:

```
Validation Accuracy: 0.7082980524978831
```

Final test accuracy:

```
Test Accuracy: 0.7008185153824442
```

d) Below is the graph of training and validation log likelihoods as a function of theta. The shape of the curve is S-shaped i.e. a sigmoid-shaped curve reminiscent of the commonly seen logistic curve. This curve represents the learning of students over time, measured by the probability of them getting the selected questions (16, 30, 100) correctly. Because they either get the question correct or not (logistic function), it makes sense that the shape of the curve is a logistic curve. Note also that they look like they have the same slope, Question 16 looks like it's shifted to the right, representing that it is likely a harder question.

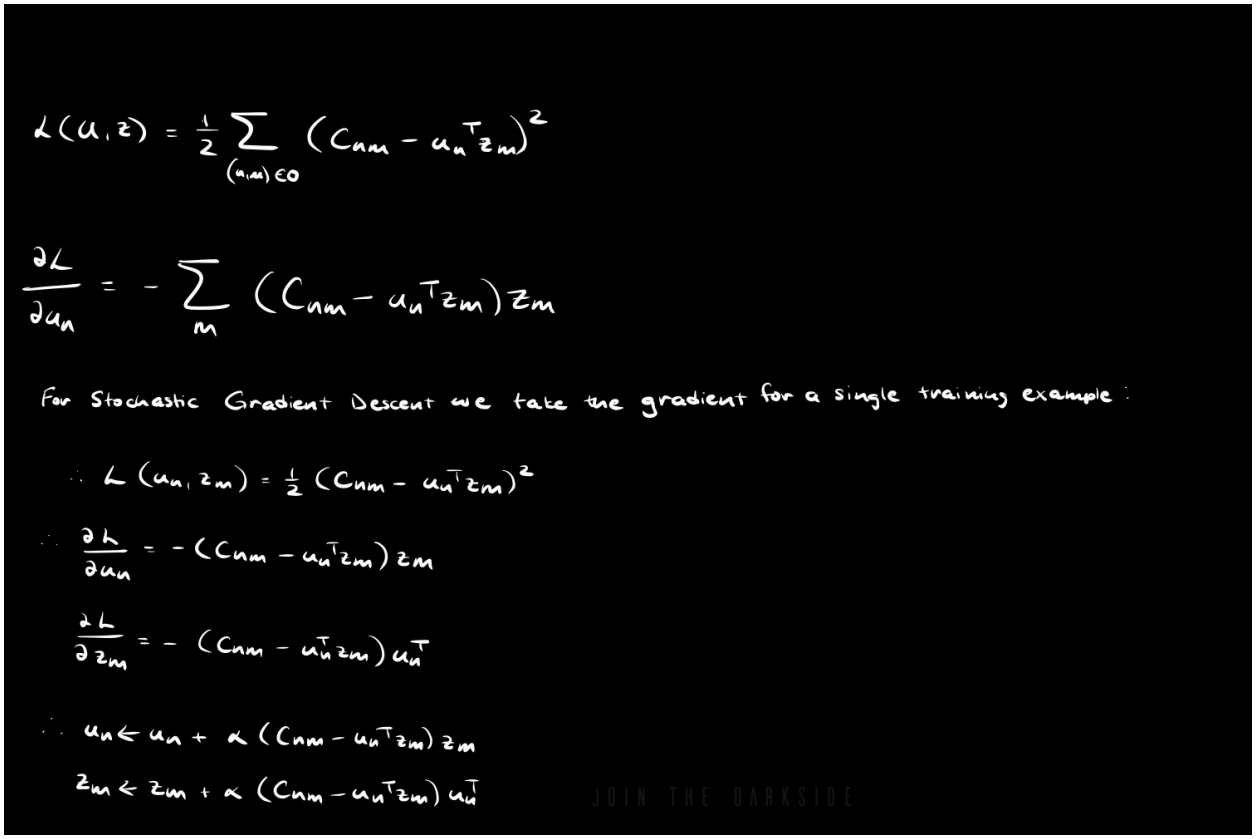


3. Matrix Factorization

a. k = 1: Validation accuracy = 0.6428168219023427
k = 6: Validation accuracy = 0.6577758961332204
k = 11: Validation accuracy = 0.656929156082416
k = 16: Validation accuracy = 0.6558001693480101
k = 21: Validation accuracy = 0.6565057860570138
k = 26: Validation accuracy = 0.6573525261078182
The optimal k is 6 with a validation accuracy of 0.6577758961332204 and test accuracy of 0.6629974597798476

b. Because SVD requires a complete matrix, we fill in all missing values with the average correctness of its respective column (which represents a question). These (user, item) pairs with missing values are unknown to us, and so filling it with the average correctness of a question can pass in errors to the SVD algorithm, resulting in errors in the reconstructed matrix. Essentially the issue is that most users do not answer every single diagnostic question, resulting in the sparsity, and SVD is not a model that is able to impute missing values.

c. Below are the gradient descent update rules for user matrix U, and item matrix Z:



d. For the sake of this problem, I decided to add another hyperparameter n, which represents the # of data points to sample per iteration of the alternating least squares (ALS) algorithm using stochastic gradient descent (SGD). In other words, per iteration, the function update_u_z will update user matrix U, and item matrix Z, n number of times.

   The reason for this addition is because SGD takes many iterations before being able to find the global optimum, or coming close to it, as can be seen below:

   For hyperparameters, learning_rate = 0.01, num_iterations = 10, and n=550000
   k = 1: Validation accuracy = 0.7018063787750494
   k = 6: Validation accuracy = 0.7022297488004516
   k = 11: Validation accuracy = 0.7033587355348575
   k = 16: Validation accuracy = 0.7015241320914479
   k = 21: Validation accuracy = 0.7046288456110641
   k = 26: Validation accuracy = 0.7040643522438611
   The optimal k is 21 with a validation accuracy of 0.7046288456110641.

   Note that the similar accuracies can be obtained as well with the following hyperparameters: learning_rate = 0.01, num_iteration = 10, n = 55000

In total, the algorithm makes approx. 550000 updates to the user and item matrices to find a decent optimum. If $n = 1$ and num_iterations instead was 550000, plotting the average squared error losses as a function of iterations would be expensive if one wants to see how the average training and validation losses converge. However, by mini-batching in each iteration, plotting the average squared error losses takes far less time, and we are able to see convergence with the training and validation datasets.

e. Below is a graph illustrating the average squared error losses as a function of iterations. As can be seen, with mini-batching, one can see convergence in the training and validation losses after approximately 10 iterations.

The final validation accuracy is 0.7046288456110641 for k = 21 with the same hyperparameters in part a.
The test accuracy is 0.7039232289020604 for the same k value with the same hyperparameters as in part a.

4. Ensemble

The objective of bagging ensembles is to reduce overfitting by averaging predictions. We will use the IRT model as the base model from Question 2 to implement bagging. By performing bootstrapping to sample with replacement 3 new data sets to calculate average predicted correctness, we hope that we can obtain higher accuracy scores. Below are the final scores for validation and test accuracy:

Final validation accuracy: 0.695532034998589
Final test accuracy: 0.695532034998589

Note that we used the hyperparameters we found to be best from Question 2, with iterations = 50 and learning rate = 0.0025, as well as the irt() function implemented from Question 2.

Note also that when comparing to the accuracy scores from Question 2, bagging does not perform as well, with about 0.01 less accuracy for both validation and testing. Therefore, bagging with a resampling size of 3 did not improve accuracy. We did initially expect bagging to improve performance since overfitting is less likely to occur as we average predictions. We conclude that resampling only 3 times is insufficient, as bagging is typically implemented with the number of resamples around 10. Additionally, bagging offers less improvement on prediction outputs since there is less room for variability, and since IRT is a logistic model, we can expect that begging offers not much improvement (Boehmke, 2020).

Part B

From Part A Question 2, since there is not a big difference between the validation accuracy and test accuracy, overfitting is not occuring. Therefore for part B, we will be aiming to improve the IRT model by adding more features to increase the accuracy. There are two strategies we will use to try to achieve this:

i) Adding a parameter α_j i.e. a discrimination parameter for each question

ii) Initializing θ by student metadata, specifically students' birth month.

1. Formal Description

The model we are extending is the Item Response Theory (IRT) model. Currently, the model we have implemented is the Rasch model as given by the following equation:

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

As an extension, we are implementing the 2-Parameter Logistic model for IRT as indicated in [1]. Its equation is given below:

$$p(c_{ij} = 1|\theta_i, \beta_j, \alpha_j) = exp[\alpha_j(\theta_i - \beta_j)]/(1 + exp(\alpha_j(\theta_i - \beta_j)))$$

Here, θ_i represents student i's learning ability, β_j is the difficulty parameter of question j, and α_j is the discrimination parameter for each question j.

The 2-Parameter Logistic model differs with the Rasch model of IRT by the addition of the discrimination parameter α_j. Because the discrimination parameter is allowed to vary between questions, the probability of correctness is allowed to intersect, and also differ in slopes for each question. The steeper the slope, the larger the discrimination is for a question. This will allow for the detection of even slight differences between a student's learning ability for each question. Inferring from this effect, the hypothesis is that it will improve predictions on correctness.

Below is the derivation of the log-likelihood as well as the derivatives of the log-likelihood with respect to θ_i, β_j, and α_j:

① Log-likelihood $\log p(C \mid \theta, \beta, \alpha)$:

$$p(C \mid \theta, \beta, \alpha) = \prod_i \prod_j p(c_{ij}=1 \mid \theta_i, \beta_j, \alpha_j)^{c_{ij}} \, p(c_{ij}=0 \mid \theta_i, \beta_j, \alpha_j)^{1-c_{ij}}$$

$$= \prod_i \prod_j \left(\frac{\exp(\alpha_j(\theta_i - \beta_j))}{1+\exp(\alpha_j(\theta_i-\beta_j))} \right)^{c_{ij}} \left(1 - \frac{\exp(\alpha_j(\theta_i-\beta_j))}{1+\exp(\alpha_j(\theta_i-\beta_j))} \right)^{1-c_{ij}}$$

$$= \prod_i \prod_j \left(\frac{\exp(\alpha_j(\theta_i-\beta_j))}{1+\exp(\alpha_j(\theta_i-\beta_j))} \right)^{c_{ij}} \left(\frac{1+\exp(\alpha_j(\theta_i-\beta_j)) - \exp(\alpha_j(\theta_i-\beta_j))}{1+\exp(\alpha_j(\theta_i-\beta_j))} \right)^{1-c_{ij}}$$

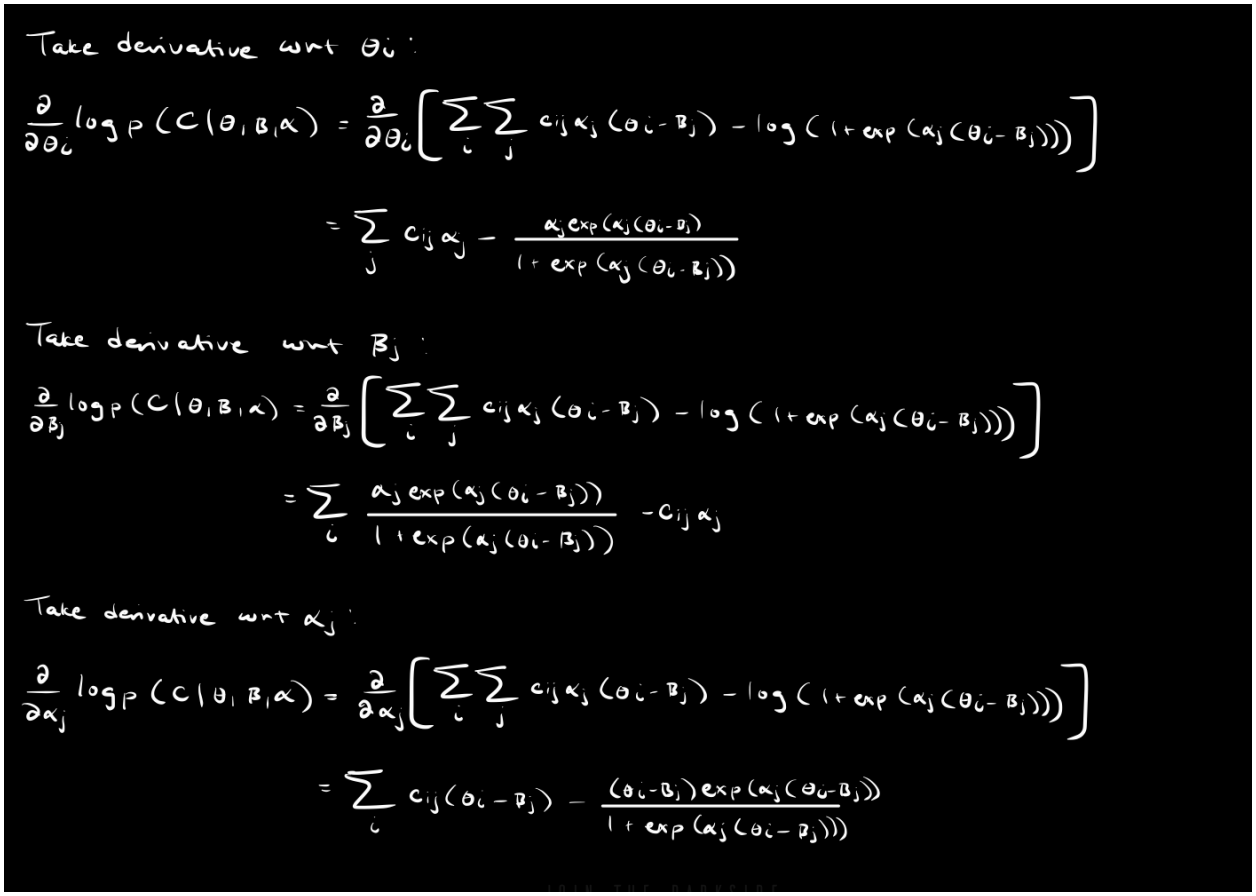
$$= \prod_i \prod_j \left(\frac{\exp(\alpha_j(\theta_i-\beta_j))}{1+\exp(\alpha_j(\theta_i-\beta_j))} \right)^{c_{ij}} \left(\frac{1}{1+\exp(\alpha_j(\theta_i-\beta_j))} \right)^{1-c_{ij}}$$

$$\log p(C \mid \theta, \beta, \alpha) = \sum_i \sum_j c_{ij} \log \left(\frac{\exp(\alpha_j(\theta_i-\beta_j))}{1+\exp(\alpha_j(\theta_i-\beta_j))} \right) + (1-c_{ij}) \log \left(\frac{1}{1+\exp(\alpha_j(\theta_i-\beta_j))} \right)$$

$$= \sum_i \sum_j c_{ij}\alpha_j(\theta_i-\beta_j) - c_{ij}\log(1+\exp(\alpha_j(\theta_i-\beta_j))) - (1-c_{ij})\log(1+\exp(\alpha_j(\theta_i-\beta_j)))$$

$$= \sum_i \sum_j c_{ij}\alpha_j(\theta_i-\beta_j) - c_{ij}\log(1+\exp(\alpha_j(\theta_i-\beta_j))) - \log(1+\exp(\alpha_j(\theta_i-\beta_j))) + c_{ij}\log(1+\exp(\alpha_j(\theta_i-\beta_j)))$$

$$= \sum_i \sum_j c_{ij}\alpha_j(\theta_i-\beta_j) - \log(1+\exp(\alpha_j(\theta_i-\beta_j)))$$



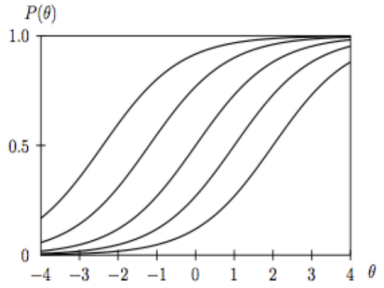
In addition to the above extension, we will be initializing the θ vector with students' metadata, specifically using their birth month. The rationale for doing this comes from a hypothesis presented in a novel titled Outliers (Gladwell, 2008). In his groundbreaking novel about success as a result of inherent advantages of individuals, it is noted an overwhelming number of hockey players in the NHL were born in the early months of the year. We thought it would be interesting to test this hypothesis since the student metadata has information about the students' date of birth. By training the 2-Parameter IRT model on students grouped based on their birth month, we hope that the similarity of student ability within each group shares enough similarities to help the model predict how the students will perform.

2. Figure/Diagram

Contrasting the 2-Parameter Logistic model with the Rasch model, the purpose of the addition of the discrimination parameter is to detect subtle differences in the respondent's ability for a particular question.

Below are graphs (Columbia, 2019) of the item response function and item characteristic curve (ICC), which illustrate the relationship between the individual's ability and the probability of them giving the correct response. Note that in the 1-Parameter model, the

ICC curves have the same slope, and shifted lines indicate that the question is increasingly difficult compared to lines on the left. Compare that to Figure 2.2, where the ICC curves of the 2-Parameter model possess different slopes. That is because the addition of the discrimination parameter (α_j) determines the rate at which the probability of a student identifying the correct answer changes given ability levels. It is essential to determine differences between students possessing similar levels of ability. Hence the ability of the slopes of the ICC to be different. The steeper the slope, the higher the discrimination of the item as it detects subtle differences between the ability of the students.



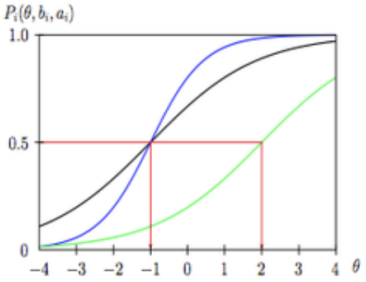


Figure 2.1: 1-Parameter model ICC curves *Figure 2.2: 2-Parameter model ICC curves*

Another extension we are involving in the model mentioned was to split the dataset into groups using student metadata of their date of birth. By extracting the value of the month, we partitioned students into groups titled “early born”, “mid born”, and “late born”, where:

- “Early borns” are students born in the first four months of the year (January - April)
- “Mid borns” are students born in the middle four months of the year (May - August)
- “Late borns” are students born in the latter four months of the year (September - December).

The reasoning behind these three groups is when starting schooling at around age 5, late borns have usually not reached their birthday yet, and thus do not qualify to enroll in school. It’s common for late borns to be held back a year to wait until they reach their 5th birthday before beginning school. In contract, early borns are usually the oldest of a cohort and may be enrolled in school a year early, while mid borns generally enroll in school without being pushed ahead or held behind.

By taking these partitioned datasets and training the 2-Parameter IRT model on them, and then averaging the theta values. Unlike in Part A where theta is initialized to be zero for all students, by training the 2-Parameter IRT model on different groups of students and taking the average theta of the group, we initialize the theta vector based on the value of the group that the ith student is in. Note that in the code, for students that were

missing metadata about their date of birth, we initialized their theta value to be zero, as was done in Part A.

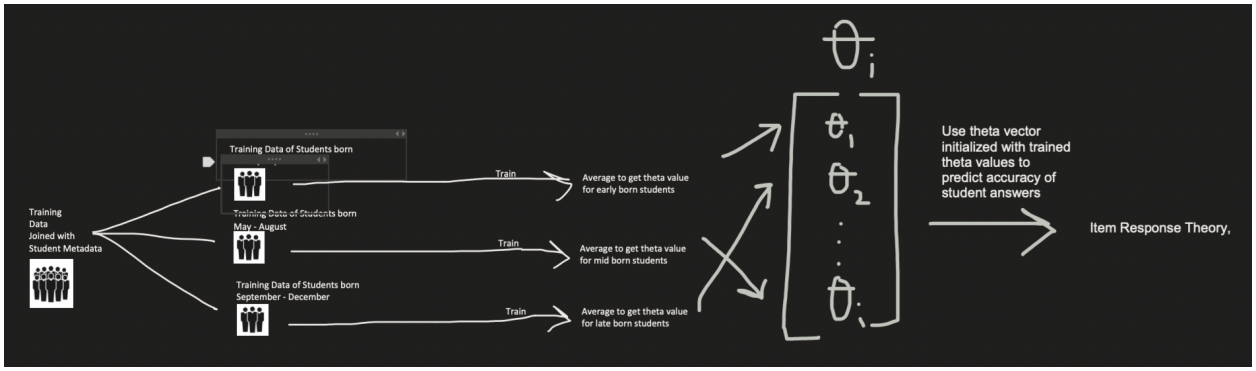


Figure 2.3: Diagram of workflow when splitting dataset based on student birth month

3. Comparison or Demonstration

Comparing Graphs of Probability of Correctness given Theta

Below are two graphs illustrating the comparison between the slopes of the probability of correctness given theta. Note that Figure 1 is the same from Part A Question 2 d), where we selected questions 16, 30, and 100 and graphed them, while Figure 2 is the same three questions graphed using our extended 2-Parameter IRT model in combination with training initializing beta using student birth month metadata.

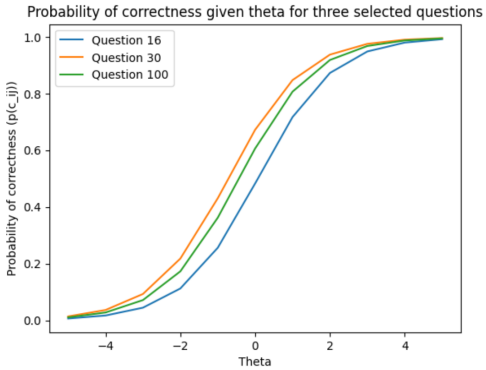


Figure 3.1: 1 Parameter/Rasch Model

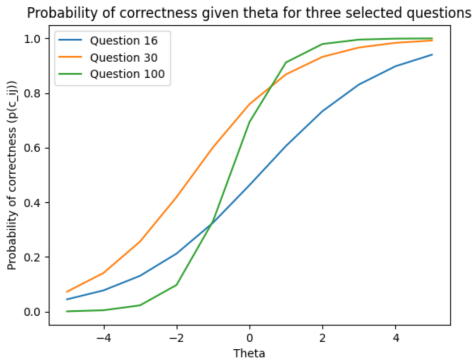


Figure 3.2: 2-Parameter Model

Note also that the slopes in the extended IRT model vary much more than the slopes in the IRT model used previously. In particular, the slope of Question 100 is much steeper compared to question 16 and 30. In Figure 3.1 we cannot conclude differences between the answering ability of the students while in Figure 3.2 we observe that the steeper slope of Question 100 indicates that there are more differences between the abilities of students to answer this question in comparison to Question 16 and Question 30.

Comparing Validation and Testing Accuracies

We have also included a table comparing validation and testing accuracies as well as the change in scores:

	IRT model from Part A Question 2	Extended IRT model using additional discrimination parameter and training beta based on student birth month metadata	Change in scores
Validation accuracy	0.7082980524978831	0.7087214225232854	~ 0.0005 increase
Testing accuracy	0.7008185153824442	0.7033587355348575	~ 0.003 increase

As seen in the right-most column, there is a slight increase in both validation and testing accuracy, with a greater increase in testing accuracy. This suggests that separating students based on some attribute of their metadata on the assumption that the groups share similar properties can increase accuracy.

4. Limitations

a) One limitation with this approach is that not all students in the metadata have a recorded date of birth (DOB). Students with unknown DOB were treated as they were in Part A. Since we were unable to determine their birth month, we could not initialize theta according to the strategy in Part A, so their learning ability was initialized to 0. With a more robust dataset, we expect that the model would do better as well.

b) We did not split students based on age, but as the dataset includes students varying from 2 to 31 years old, the developmental advantages vary greatly, as there is a larger developmental advantage between a cohort of students who are younger. For example: developmentally wise, a cohort of 2 year old children can range from 24 months to 35 months old, and will be in very different developmental stages of learning (NICHD Early Child Care Research Network, 2007). If there wasn't such a wide range of student ages, we expect that the model would perform better.

c) Another limitation is one that concerns the IRT statistical model in general. One of the assumptions that must be true to be able to apply the IRT model is the assumption that a subject's performance on one item must not affect, either negatively or positively any other items on the test. In the context of our dataset, we would like to assume that students do not improve while answering questions on the assessment. But in practice, every previous question may reveal information about the next question, or improve our skill for the next question. Negative effects also exist, such as when students know they are doing badly, it becomes harder for them

to focus on future questions. The larger the dataset, the harder it is to also maintain independence. A remedy to this limitation could be to use a probabilistic network structure in order to relax this assumption that independence is required, since network model structures can be used to assume local independence (Ueno, 2002).

Contributions of Team Members

Part A

1. Rajvi
2. Jingfei
3. Rajvi
4. Jingfei

Part B

1. Rajvi
2. Jingfei
3. Jingfei
4. Rajvi

References

1. Columbia University, 2019. Item Response Theory. Retrieved from https://www.publichealth.columbia.edu/research/population-health-methods/item-response-theory
2. Gladwell, Malcolm, 2008. Outliers : the story of success. New York: Little, Brown and Company.
3. NICHD Early Child Care Research Network, 2007. Age of Entry to Kindergarten and Children's Academic Achievement and Socioemotional Development. Early Educ Dev.
4. Ueno, Maomi, 2002. An Extension of the IRT to a Network Model. Behaviormetrika Vol. 29 No. 1 pg. 59-79.
5. Boehmke, Bradley and Greenwall, Brandon, 2020. Hands-On Machine Learning with R. CBC Press.