CSC311 Project

Yuan Dou, Yiyi Tan, Xi Zheng

November 2021

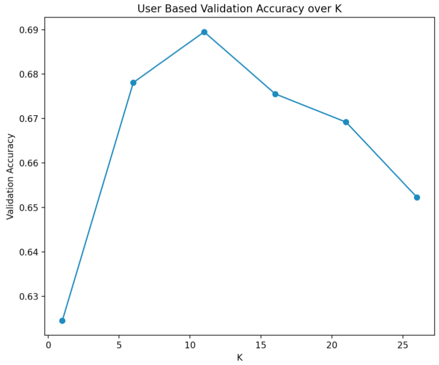
Part A

k-Nearest Neighbor

(a) Following is the result of the accuracy on the validation data as a function of k

```
k = 1:  0.6244707874682472
k = 6:  0.6780976573525261
k = 11:  0.6895286480383855
k = 16:  0.6755574372001129
k = 21:  0.6692068868190799
k = 26:  0.6522720858029918
```

Following is the plot of the accuracy on the validation data as a function of k



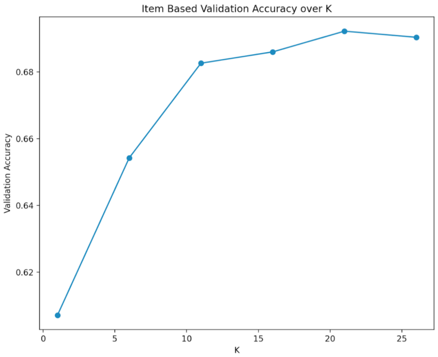
(b) We chose `k* = 11` which gives `test accuracy:  0.6841659610499576`

(c) The underlying assumption on item based collaborative is that if question A is matched the correctness as question B by a student. The correctness of question A and question B answered by other different students are the same. In other words, if

1. Following is the result of the accuracy on the validation data as a function of k:

   ```
   k = 1:  0.60711261642675
   k = 6:  0.6542478125882021
   k = 11:  0.6826136042901496
   k = 16:  0.6860005644933672
   k = 21:  0.6922099915325995
   k = 26:  0.69037538808919
   ```

   Following is the plot of the accuracy on the validation data as a function of k



2. We chose `k* = 21` which gives `test accuracy:  0.6683601467682755`

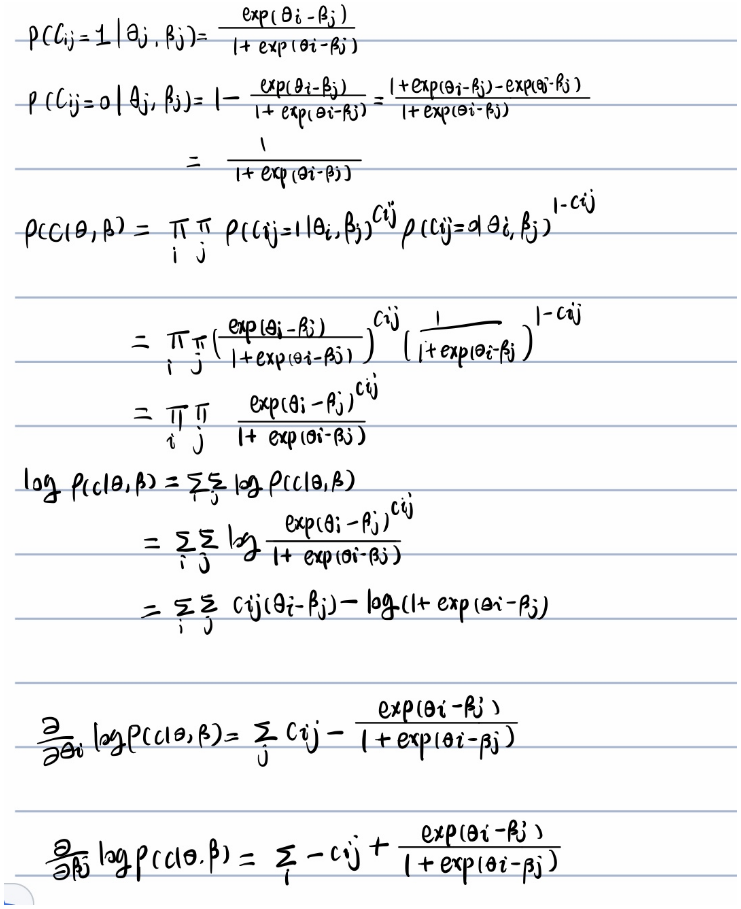
(d) Comparison of the test performance
We believe that user- based collaborative filtering performs better since the final test accuracy of user-based is higher than the item-based.

(e) Potential Limitations of kNN

1. **Inefficiency**(high computational cost): As we have a large full dataset, the cost of calculating euclidean distance becomes higher with larger dataset. As a result, it needs to iterate through each training data point and to calculate the distance between them for each prediction even though we have a low dimensionality data.
2. **Inaccurate**: There are many missing values in the original dataset so we manually impute the missing values using KNNImputer. However, kNN is sensitive to noisy data like “nan” and outliers since a mislabeled data point (if we impute it wrongly) or outliers could change the decision boundary.
3. **Normalization**: If there exist more features (as the provided metadata), nearest neighbors can be sensitive to the ranges of different features. Thus, we need to fix it by normalizing each dimension.

Item Response Theory

(a) See image below.



$$p(C_{ij}=1 \mid \theta_j, \beta_j) = \frac{\exp(\theta_i - \beta_j)}{1+\exp(\theta_i - \beta_j)}$$

$$p(C_{ij}=0 \mid \theta_j, \beta_j) = 1 - \frac{\exp(\theta_i - \beta_j)}{1+\exp(\theta_i - \beta_j)} = \frac{1+\exp(\theta_i - \beta_j) - \exp(\theta_i - \beta_j)}{1+\exp(\theta_i - \beta_j)}$$

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

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

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

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

$$\log p(C \mid \theta, \beta) = \sum_i \sum_j \log p(C \mid \theta, \beta)$$

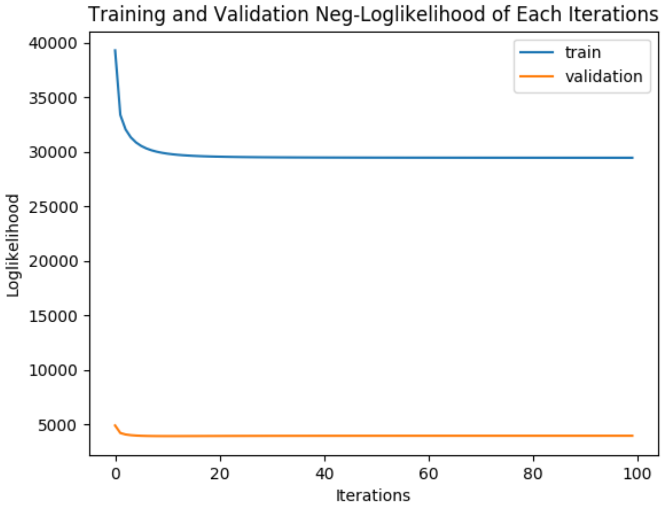
$$= \sum_i \sum_j \log \frac{\exp(\theta_i - \beta_j)^{C_{ij}}}{1+\exp(\theta_i - \beta_j)}$$

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

$$\frac{\partial}{\partial \theta_i} \log p(C \mid \theta, \beta) = \sum_j C_{ij} - \frac{\exp(\theta_i - \beta_j)}{1+\exp(\theta_i - \beta_j)}$$

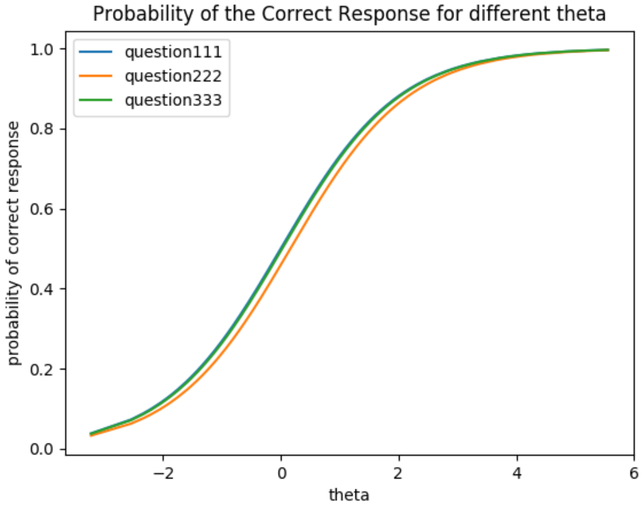
$$\frac{\partial}{\partial \beta_j} \log p(C \mid \theta, \beta) = \sum_i -C_{ij} + \frac{\exp(\theta_i - \beta_j)}{1+\exp(\theta_i - \beta_j)}$$

(b) See code in item_response.py.
Hypermeter we choose: `lr = 0.03, iteration = 100`



(c)
```
lr:  0.03, iterations:  100
Validation Accuracy:  0.7074513124470787
Test Accuracy:  0.7098504092576913
```

(d) Theta in x-axis denotes a student's ability and the y-axis is the probability of the correct response. The shape of the curves looks like a sigmoid, and it denotes that given a question j, the higher a student's ability, the more likely he/she is to correctly answer the question (Note: the blue line for question 111 is almost coincide, so it looks not obvious from the graph).



Neural Networks

(a)
- ALS matrix factorization usually takes in sparse matrix, while neural networks can take in many different data structures, like image, text and etc.
- ALS uses least squares to learn, while NN uses gradient descent to learn.
- ALS matrix factorization usually used to do recommendation systems, while NN can be used to do almost anything, like autoencoder for the project, CNN for image classification and etc.

(b) See code in `neural_network.py`.

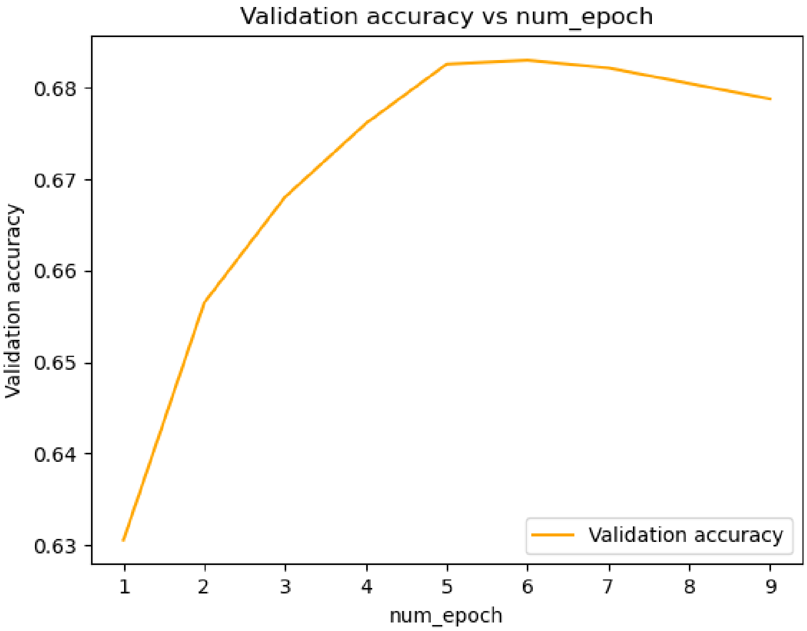
(c) See code in `neural_network.py`.
See full output in `neural_network/3c.txt`.

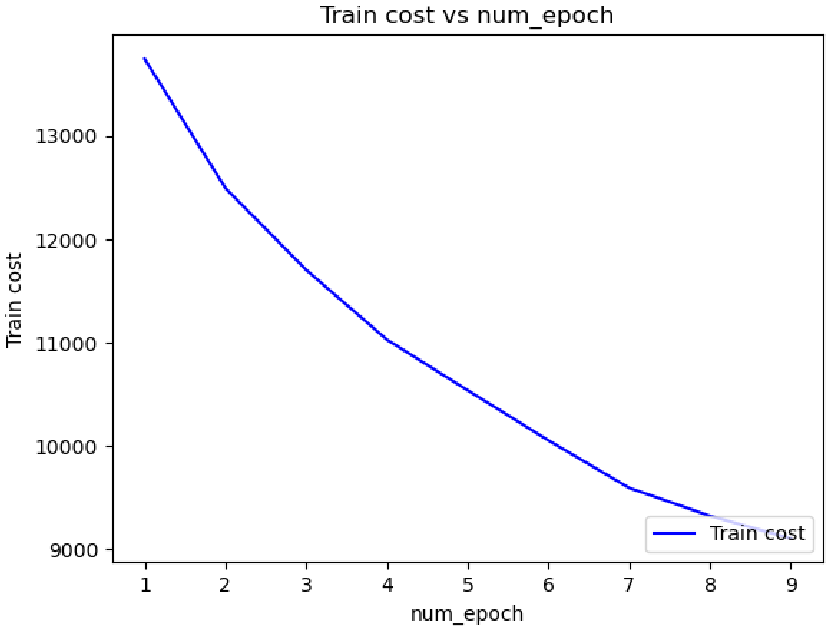
```
best_acc:  0.6311035845328817, best_k:  50, best_lr:  0.1
```

(d) See code in `neural_network.py`.
See full output in `neural_network/3d.txt`.

```
Test accuracy:0.6793677674287327, best_epoch:  6
```

Note: I implemented the NN with early stopping, NN stops if there is no improvement for 3 epochs. Details in `neural_network.py`.





(e) See code in `neural_network.py`.
See full output in `neural_network/3e.txt`.
`Test accuracy:0.6831498729889923, best_lamb:  0.001`
The model does perform better with the regularization penalty, however the improvement is quite small.

Ensemble

1. We will use the User Based KNN as the base model to improve the stability and accuracy of it.
   First, we re-sample 3 training data (draw the data randomly) with replacement. The new sample have a same sample size as the original training data. Then we run User based KNN 3 times using the newly sampled data we generated above with $k = 11$. After that, we evaluate 3 new sample data and get the accuracy for each of the sampled data. Average the predicted correctness and report it. Similarly, we follow the same procedure on validation data set and get the resulting accuracy.

   ```
   Validation Accuracy:  0.6021732994637313
   Test Accuracy:  0.5930002822466836
   ```

   Compare to the validation accuracy and test accuracy from question 1, we can see that both of the accuracy drop down about 10%. One possible reason is that when we generated sample data there might appear some noise data point which could ruin the model. Hence, the bagging ensemble does not have a better performance on the original User Based KNN.

2. We will use the three bases we implemented previously (KNN, Item Response Theory, Neural Network)to improve the stability and accuracy of it by ensemble.
   First, we re-sample 3 training data (draw the data randomly) with replacement. The new sample have a same sample size as the original training data. Then we run User based KNN, Item Response Theory and Neural Networks using the newly sampled data we generated above with $k = 11$ for KNN, $lr = 0.03$, iterations $= 100$ for IRT , $lr = 0.01, \lambda = 0.001$ for Neural Networks. After that, we evaluate 3 new sample data and get the accuracy for each of the sampled data. Average the predicted correctness and report it. Similarly, we follow the procedure on validation data set and get the resulting accuracy.

   ```
   Validation Accuracy for ALL: 0.5850562141311507
   Test Accuracy for ALL: 0.5834274155612005
   ```

   Compare to the validation accuracy and test accuracy from question 1, 2 and 3, we can see that both of the accuracy drop down. Therefore, the bagging ensemble does not have a better performance on the original User Based KNN, Item Response Theory and Neural Networks. We didn't expect this conclusion before doing the ensemble. We believed that bootstrapping the data could improve the accuracy by averaging prediction. A possible reason is we only used bootstrap to generate 3 samples which is not acquire in this circumstance. If we generated more samples from original data, we might able to see the improvement on the prediction.

Part B

Formal Description

We add a discrimination parameter α_j to our 1-Parameter model to get a 2-Parameter model. The discrimination parameter is allowed to vary between questions. Our new 2-P model estimates the likelihood of a correct answer by difficulty β_j and discrimination α_j. The key difference to the 1PL model is that the expression $\exp(\theta_i\text{-}\beta_j)$ is replaced with $\exp[\alpha_j(\theta_i\text{-}\beta_j)]$.
The equations change as follows:

For 1-P model: For 2-P model:

$$\log p(\mathbf{C}\,|\,\boldsymbol{\theta},\boldsymbol{\beta}) = c_{ij}(\theta_i-\beta_j)-\log(1+\exp(\theta_i-\beta_j)) \rightarrow \log p(\mathbf{C}\,|\,\boldsymbol{\theta},\boldsymbol{\beta},\boldsymbol{\alpha}) = c_{ij}\alpha_i(\theta_i-\beta_j)-\log(1+\exp\alpha_i(\theta_i-\beta_j)))$$

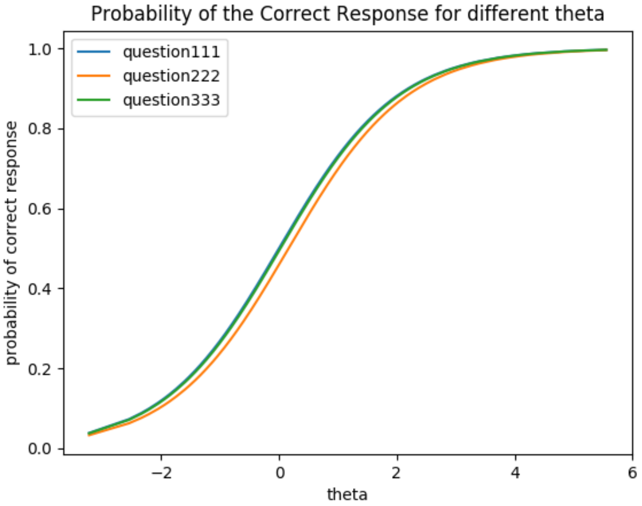
$$\frac{\partial l}{\partial \theta_i} = \sum_{j=1}^{1774}(c_{ij}-\frac{\exp(\theta_i-\beta_j)}{1+\exp(\theta_i-\beta_j)}) \rightarrow \frac{\partial l}{\partial \theta_i} = \sum_{j=1}^{1774}(c_{ij}\alpha_j-\frac{\alpha_j\exp\alpha_j(\theta_i-\beta_j)}{1+\exp\alpha_j(\theta_i-\beta_j)})$$

$$\frac{\partial l}{\partial \beta_j} = \sum_{i=1}^{542}(-c_{ij}+\frac{\exp(\theta_i-\beta_j)}{1+\exp(\theta_i-\beta_j)}) \rightarrow \frac{\partial l}{\partial \beta_j} = \sum_{i=1}^{542}(-c_{ij}\alpha_j+\frac{\alpha_j\exp\alpha_j(\theta_i-\beta_j)}{1+\exp\alpha_j(\theta_i-\beta_j)})$$

$$\frac{\partial l}{\partial \alpha_j} = \sum_{i=1}^{542}(c_{ij}(\theta_i-\beta_j)-\frac{(\theta_i-\beta_j)\exp\alpha_j(\theta_i-\beta_j)}{1+\exp\alpha_j(\theta_i-\beta_j)})$$

.

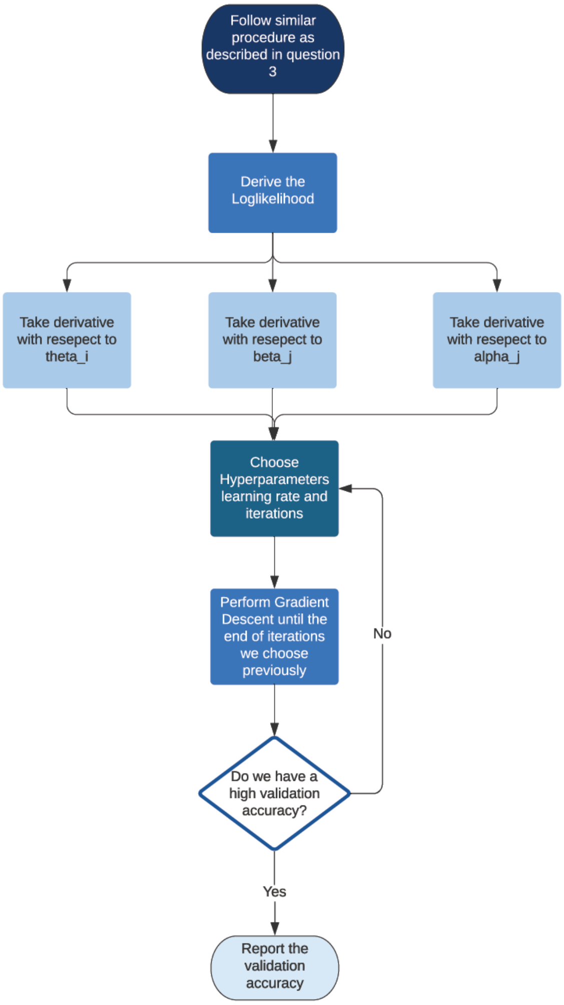
In our 2-P model above, β_j represents the difficulty of the question j , θ_i represents the i-th student ability, and α_j represents the discrimination, the probability of endorsing correctly answering question j.



In our 1-P model from part a, the discrimination parameter α_j is fixed for all questions with the value 1, as we can see, the slope for all lines in figure above are the same. Note that question 111 and question 333 are coincide because of the fixed discrimination parameter. But for a 2-P model, the curves of the various questions might now intersect and have varied slopes, so we expect the curve of question 111 and 333 for our 2-P model are not coincide.
We expect our model would improve the optimization as we introduced a new parameter and that makes our model gets more complicate.

Figure or Diagram

Following flow chart shows the overall idea when modifying IRT we implemented in part A.


Follow similar procedure as described in question 3
Derive the Loglikelihood
Take derivative with resepect to theta_i
Take derivative with resepect to beta_j
Take derivative with resepect to alpha_j
Choose Hyperparameters learning rate and iterations
Perform Gradient Descent until the end of iterations we choose previously
Do we have a high validation accuracy?
No
Yes
Report the validation accuracy

Comparison or Demonstration

After we tested for different parameters, we found the best ones are:

```
best_lr:0.01, best_itr:75
```

With theses parameters, we get:

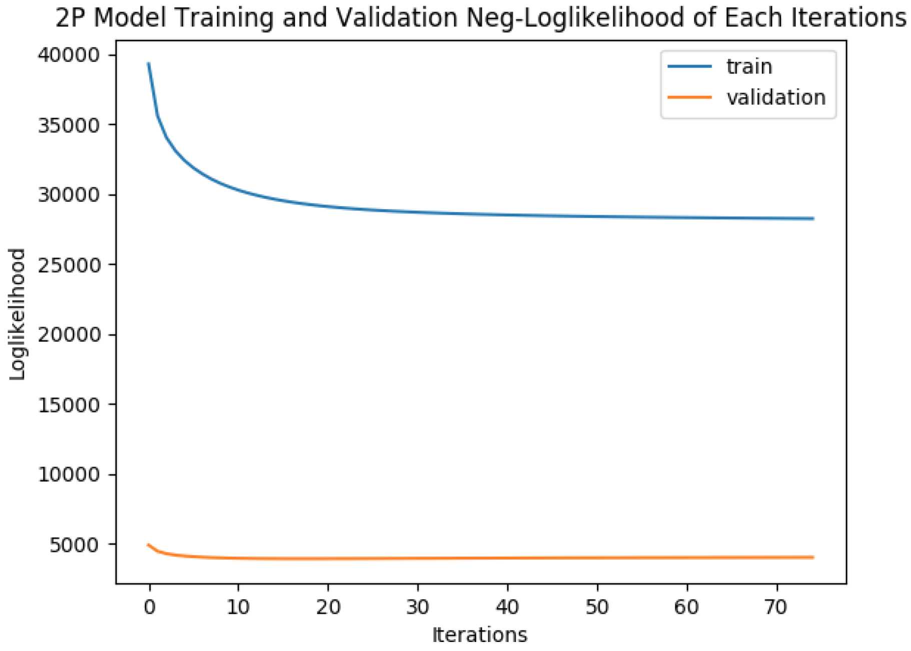
```
Validation Accuracy:0.7084391758396839
Test Accuracy:0.7044877222692634
```

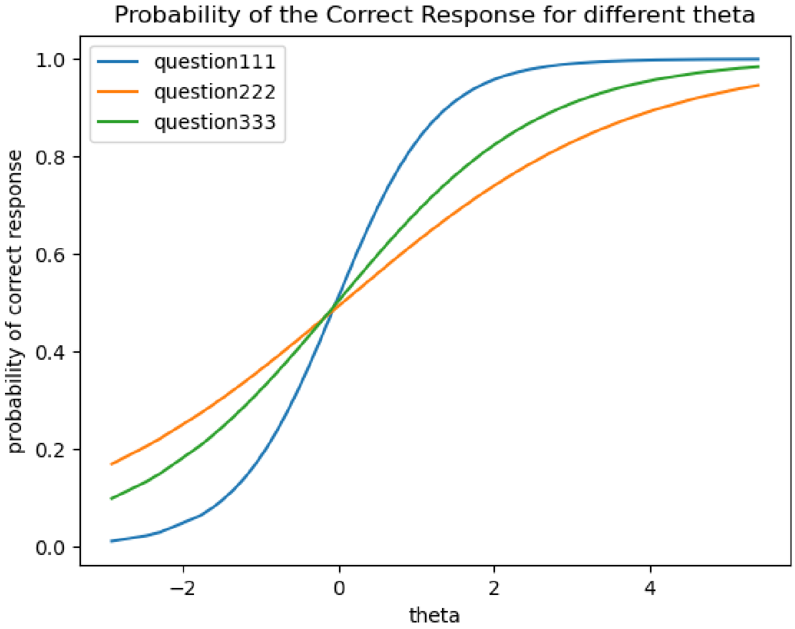
Compare to our original item response theory model which has:

```
Validation Accuracy:0.7074513124470787
Test Accuracy:0.7098504092576913
```

Note that the validation accuracy increased by a little bit which is expected, however the test accuracy went down by 0.005 that is just the randomness of the data. If we take a look at the output of partb.txt which has the detail output of partb, we can see that the average validation accuracy and test accuracy did increase compare to part a.

The negative log-likelihood given by the two models are really similar.





If we compare the plot generated using the newer model to the original plot(Part b first plot). It is not hard to see that different questions now have discrimination and the plot is as expected(curve of question 111 and 333 for our 2-P model are not coincide). Although we do not see that much of the difference in the accuracy, but even a little bit of increase is good.

The benefits we have compare to the original model is due to optimization obviously, to explain again briefly in another way. Questions are different. They are not iid events, so when we consider them as different questions, it is more realistic and gives a little bit more accuracy.

Limitations & Extensions

- Limitations
  1) Our modified 2 parameters item response theory model is only one-dimensional. As a result, the response of the prediction is either correct or incorrect. If there are multiple choices with more than one correct answer in the diagnostic questions, then it is impossible to use item response theory algorithm to predict the correctness of the questions for different students. Hence, our improved version of IRT is useless in this more practical case.

  2) There are two hyper-parameters we need to tune for our model include learning rate and iterations. As tuning hyper-parameters manually is an

inefficient optimization strategy. Even though 2 parameters are not high dimensional, we still need to record all the accuracy and compare them in order to find the one with highest performance. It could potentially occur overfitting with a very well prediction without using cross validation.

3) The number of parameters is limited in our model, in the real world, there are more factors may influence student's responds, below extensions may produce better models than we have now.

- Some Possible Extensions
  1) Student with low ability may get the answer correct by chance, so one possible extension is that adding the guessing parameter g_i to create a 3PL model. Our 2PL model assumes that the guessing parameter g_i is fixed for all question with the value 0. The guessing parameter adds a lower asymptote, meaning that regardless of how low a student's ability is, the probability can only go as low as g_i. The equation of the 3PL model looks like:

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

  2) Add age, gender, premium pupil parameters from student metadata to our model, since our above improvement is only focusing on question, student's ability may be influenced by other factors. Gender and age may also have an great effect on the probability that they answer the question correctly.

Contributions

Yuan Dou: Implemented Neural Networks, Part B
Yiyi Tan: Implemented KNN and Ensemble, Part B
Xi Zheng: Implemented IRT, Part B

Reference

1. Item response theory. Search the website. (n.d.). Retrieved December 3, 2021, from https://www.publichealth.columbia.edu/research/population-health-methods/item-response-theory.