CS6923 Final project

Part B - Modified Algorithm

Jeevan Vasanthan (jv1564)
Aya Oshima (ao2368)

Description

For our algorithm, we extended item response theory. The idea of item response theory is to assign each student an ability value and each question a difficulty value in order to create a probability distribution. The problem with this is we assume unidimensionality and local independence for our items when assigning value. This leads to our model being inaccurate and underfitting as the model is too simplistic. We can see this in our item response theory results as the training data yields an abysmal 0.7404036127575501 accuracy while the validation data yields around 0.7067456957380751 accuracy and the test data yields around 0.7084391758396839 accuracy. Our proposed model diversifies the features that we are using to train our model by adding and/or transforming features that may also have an impact on a student getting a question right or wrong. For this model, we create a completely new feature matrix that we can train using popular classification algorithms such as logistic regression and decision trees. Obviously, we included the question and the student as two features in this matrix. However, student ids and question ids are not meaningful information so we took the alpha and beta vectors that we optimized through IRT and replaced the ids with their respective item values. Finally, we added the subjects that each question has as another feature. This was originally represented as a list of ids for each question. To represent a list of ids in our feature matrix, we encoded the list by creating 387 more features (one for each possible subject). The values for each subject feature were binary : 1 if the question contained that subject and 0 otherwise. This encoding significantly increased the dimensionality of our feature matrix so we used PCA(principal component analysis) to reduce it in an attempt to remove any noise that can hurt our results. We also made an attempt at feature hashing , but it gave very poor results and was thus scrapped.

Equations (not much different)

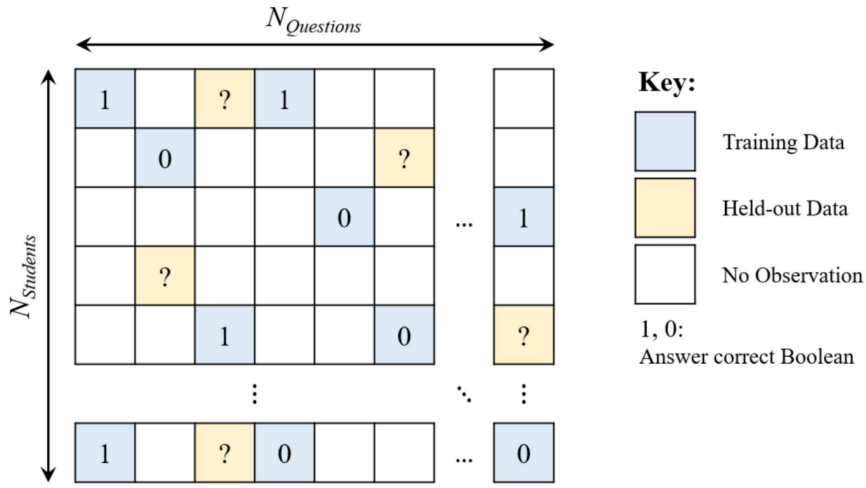
$p(cij = 1|\theta i , \beta j) = \exp(\theta i - \beta j) / 1 + \exp(\theta i - \beta j)$

Prediction Accuracy = number of correct predictions / number of total predictions

feature one value = θ[student_id] (in feature matrix)

feature two value = β[question_id] (in feature matrix)

Sparse Matrix that IRT uses to train prediction model:



Feature Matrix for our algorithm:

	f1	f2	f3	f4	……	f389
1						
2						
3						
4						
5						
6						
...	…	…	…	…		…
…	…	…	…	…		…
…	…	…	…	…		…
n						

f1 is a feature that is filled with the optimized theta values that we got from running IRT.

(θ[student_id])
f2 is a feature that is filled with the optimized beta values that we got from running IRT.
(β[question_id])
f3…f389 represents every possible subject that a question can contain filled with binary values

Decomposed Feature Matrix

We use PCA to decompose the matrix and reduce the dimensionality of the matrix. k represents the number of dimensions we are reducing our matrix to.

	f1	f2	f3	f4	……	fk
1						
2						
3						
4						
5						
6						
...	…	…	…	…		…
…	…	…	…	…		…
…	…	…	…	…		…
n						

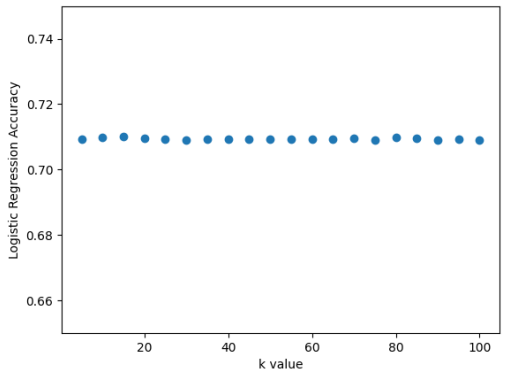
In both feature matrices, n represents the number of samples.
Target vector is a vector of the binary classifications for each student, question pair.

Comparison

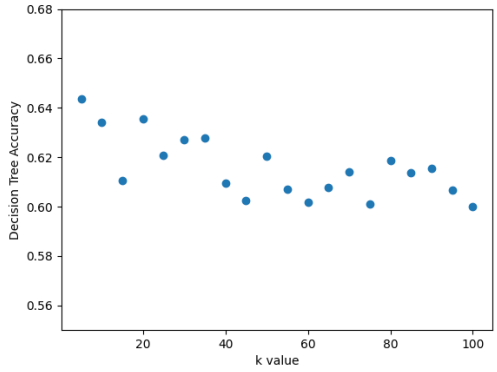
	Training Accuracy	Validation Accuracy	Test Accuracy
IRT classifier	0.74040361275755	0.70674569573807	0.7084391758396
Logistic Regression using our new feature matrix	0.74052709568162	0.70985040925769	0.7098504092576
Decision Trees using our new feature matrix	0.99566045723962	0.65396556590460	0.6553767993226

As we can see here, the logistic regression model on the new feature matrix didn't do much better than the IRT classifier. An interesting result is that the decision tree model on the new feature matrix had almost 100% accuracy on the training data, but did worse than the IRT classifier on the validation and test data.

Logistic regression



Decision tree



This is a scatter plot for both models when we were tuning our k values (for PCA decomposition).

For the logistic regression model, it seems like the k value doesn't have too much of an impact on the overall model accuracy.

On the other hand, for the decision tree model, it generally had better accuracies with lower k values.

Conclusion

As we have seen by the results, our proposed model didn't do much better than the item response theory model. The logistic regression model on our new feature matrix had about the same results. We suspect that this is because logistic regression is a linear model that generally works with two features at a time. Therefore, even though we made an attempt to diversify our model and make it less simple, the features may not be strong enough predictors of an outcome. Another possibility is that we have to encode our features in a better way. Both of these possibilities were demonstrated earlier when doing matrix decomposition where reducing our feature dimensionality barely had an effect on the model accuracy (shown by the scatter plot). The decision tree model on our new feature matrix had a different issue. This model was severely overfitting unlike before as the training data yielded almost 100% accuracy while the test and validation data did worse than the original IRT model. Decision trees are generally prone to overfitting in general , but our encoding of the subject feature leading to high dimensionality certainly didn't help. The scatter plot of accuracy vs k for decision trees again shows this. When k decreases (less dimensionality), the decision tree accuracy generally improves. Pruning and random forests are techniques that we can use to help improve our

results in the future. Finally, we need to figure out a better representation of our features (something that is more meaningful to binary classification).