CS-GY 6923 Final Project

Team - ml_team: Mohit Mehta(N17202629), Surya Narayan(N19473932)

December 20, 2022

1 Introduction

This report summarises the machine learning models and techniques used for predicting whether a particular student was able to correctly answer the given question. The dataset was provided by Eedi, an online education program.
The report analyses the following machine learning techniques

1. K Nearest Neighbour
2. Item Response Theory
3. Support Vector Machine Classifier
4. Trees
5. Neural Network

2 Data

We sub-sampled answers of 542 students to 1774 diagnostic questions from the dataset provided by Eedi , an online education platform that is currently being used in many schools. The platform offers crowd-sourced mathematical diagnostic questions to students from primary to high school (between 7 and 18 years old).

3 Machine Learning Techniques

3.1 K-Nearest Neighbours(KNN) (Part a)

Implementation

We implemented user based collaborative filtering, which treats questions as feature vector and iterate over all the students. In user based collaborative filtering, each user is considered as different entity and we try to interpret a user aptitude for a given question.

We also implemented item based collaborative filtering, which takes into account difficulty of each question rather than the user aptitude as tough questions has less probability of being answered correctly.

Results

- In case of user based collaborative filtering we achieved the best test accuracy of **68.471** for $k^* = 11$.
- In case of item based collaborative filtering, we achieved the best test accuracy of **66.892** for $k^* = 11$.

Inference

As we can observe, user based collaborative filtering outperforms item based collaborative filtering because it is more easy to group the data by users rather than items, because different students will have different strength and user based filtering captures on this fact while item based filtering only take into account the difficulty of the questions.

Limitations

The potential limitation of KNN includes:

- KNN can only capitalize either on item response or user response but not both at the same time
- The KNN algorithm is not robust to outliers, as a single outlier can significantly influence the results of the model.
- The KNN algorithm assumes that the data is homogeneous, meaning that the underlying distribution of the data is the same for all classes. If this assumption is not met, the model may perform poorly.

3.2 Item Response Theory(Part a)

Derivation of Gradient and Loss

Below, we derive negative log likelihood of the item response part and calculate its gradient.

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

$$p(c_{ij} = 0|\theta_i, \beta_j) = \frac{1}{1 + exp(\theta_i - \beta_j)}$$

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

$$p(C|\theta, \beta) = \prod p(c_{ij}|\theta_i, \beta_j)$$

$$NLL = -\log p(C|\theta, \beta) = -\sum \log(p(c_{ij}|\theta_i, \beta_j))$$

$$NLL = -\sum c_i \log\left(\frac{exp(\theta_i - \beta_j)}{1 + exp(\theta_i - \beta_j)}\right) + (1 - c_i) \log\left(\frac{1}{1 + exp(\theta_i - \beta_j)}\right)$$

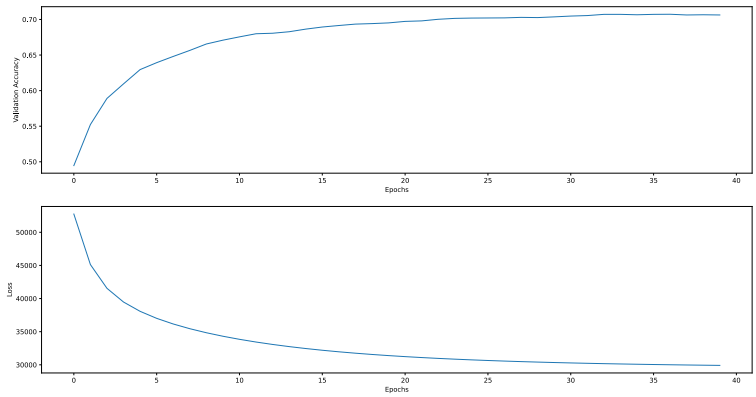

Figure 1: Item Response Theory Training

$$\frac{\partial NLL}{\partial \theta_i} = \left(\frac{exp(\theta_i - \beta_j)}{1 + exp(\theta_i - \beta_j)} \right) - c_i$$

$$\frac{\partial NLL}{\partial \beta_j} = c_j - \left(\frac{exp(\theta_i - \beta_j)}{1 + exp(\theta_i - \beta_j)} \right)$$

Implementation

As opposed to KNN, which only capitalizes either on student aptitude or difficulty of question, Item Response Theory(IRT) takes into consideration both of the factors.The IRT assigns each student an ability value and each question a difficulty value to formulate a probability distribution.

For the current implementation, we used negative loss likelihood coupled with stochastic gradient descent for training the model.

Results and Plots

After tuning the hyper-parameters($lr = 0.008$ and $iterations = 40$), we achieved best validation accuracy of **70.434** and test accuracy of **70.769** with NLLK loss at **29931.25**. The training plot has been shown in Figure-1.

Furthermore, we take random 3 questions out of the dataset and evaluate it over all the students. Figure-2 summarizes the results we obtain. By simply looking at the plot, we can see that all three questions have same high and low, with high and low being separated by same factor across the graph. One can infer, that the spacing between the three question for a particular students indicated the difference in question difficulty in context of the student while the high and low indicate the student aptitude.

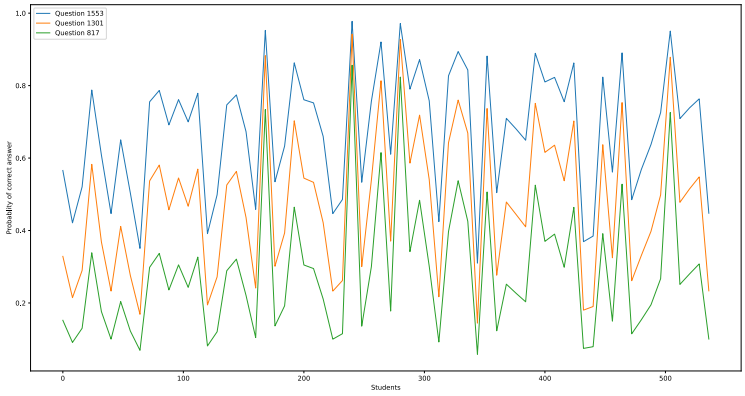

Figure 2: Variation of probability of correct answers over all students for 3 random questions

Inferences

IRT outperforms the KNN in terms of accuracy. This is mainly due to the fact that it considers both the question difficulty and the user aptitude.

Limitations

One of the main limitation of IRT is its simpler representation of weights and a hard coded transfer function. Furthermore, the IRT only has a single layer(in context of NN).

3.3 Standard Neural Networks(Part a)

Implementation

We use two layer neural network with sigmoid activation function.The implementation iterates over all the questions for each student and generalizes the answer based on question representation.

Results and Plots

After tuning the hyper-parameters, we achieved best validation accuracy of **68.44** and test accuracy of **67.9** for k = **50** learning rate = **0.01** and epoch = **40** Optimal value of λ is **0.001** and validation accuracy is **68.19** and test accuracy is **68.81**.Regularisation increases test accuracy.

Limitations

Main problem with standard NN is that is considers only question representation and does not consider student representation.

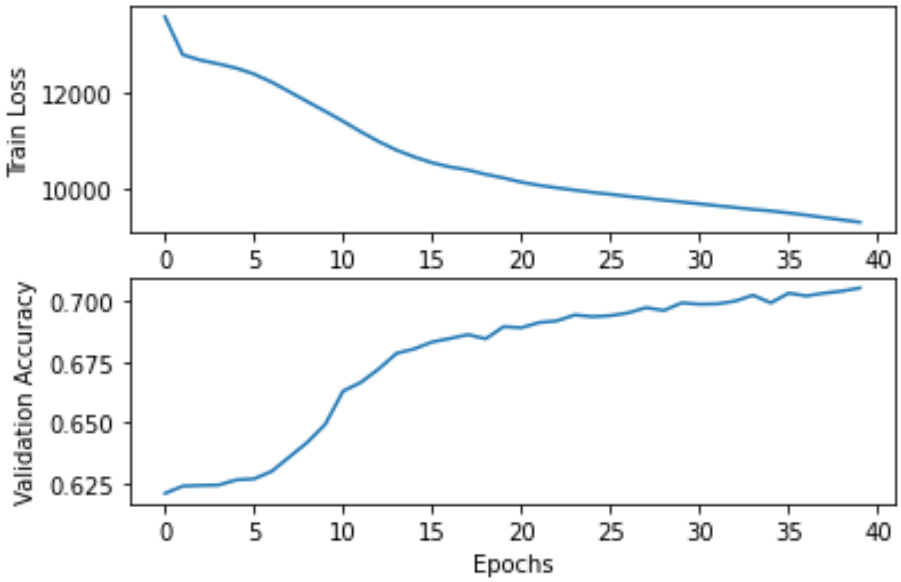

Figure 3: Plotting graph of Train Loss and Validation Accuracy for K=50

3.4 Support Vector Machine Classifier

Implementation

We applied support vector classifier(SVC) with radial basis function as our kernel. In order to implement SVC, we first convert our data in 3D format with question and student being x and y axis of the graph and z axis being the correct answer (z can either take value of 0 or 1).

Results

We achieved validation accuracy of **60.076** and test accuracy being **59.5822**

Limitations

The main reason behind the poor performance of SVM is due to the fact there is no relation in consecutive values in both x and y axis i.e. y_n and y_{n+1} are unrelated for x_n and x_{n+1} respectively. However, this problem can be mitigated by carefully feature engineering the input data.

3.5 Trees

Implementation

We applied XGBoost and LightGBM with one hot encoding as inputs, as done with our custom neural network. In order to implement these algorithms, we first convert our data to one hot format and then pass it on.

Results and Plots

After adjusting the hyper-parameters for LightGBM we got **67%** train accuracy and **65%** test accuracy. For XGBoost we got **73%** train accuracy and **68%** test accuracy

Limitations

As we are passing one hot encoding data is very sparse. So decision trees are very prone to outliers.

3.6 Custom Neural Network

Implementation

The standard implementation of neural network given in part a of the project only considers type and difficulty of the questions. The implementation iterate over all the questions for each student and generalizes the answer only based on question representation. As, we noted earlier, the similar pattern can be seen in KNN and IRT, in which IRT outperforms KNN mainly due to the fact that it consider both student and question representation.

We used the fact stated above and instead of only passing the question through the neural network, we are passing both the question and student to the neural network in form of a one-hot vector over all possible student-question pair. **The main intuition behind this is to bank on the fact that neural network will be able to learn both question and user representation, without being restricted by hard coded form of the IRT.**

For training the network, we used 56688 samples with 7086 samples for validation and 3543 samples for testing. We used a 3 layer linear neural network of size (100,100,1) with input shape being (number of students + number of questions). For loss, we used binary cross entropy loss accompanied by Adam optimizer with l_2 regularization.

Results

Using $learning\_rate = 0.005$, $batch\_size = 128$, $regularization = 0.00005$ for $epoch = 20$, we achieved validation accuracy of **71.056** and testing accuracy of **71.521**

Figure-4 summarises the training loss and validation accuracy for custom neural network.

Inference

Our custom made neural network module outperforms all the algorithm tested. This is mainly due to the fact that it is better generalization of IRT and takes into account both student representation and question representation.

Limitations

- The following approach is highly prone to over-fitting, thus reducing the generalization of the model. This is mainly due to the fact that our input is sparse and small size of the dataset. This can be mitigated by using

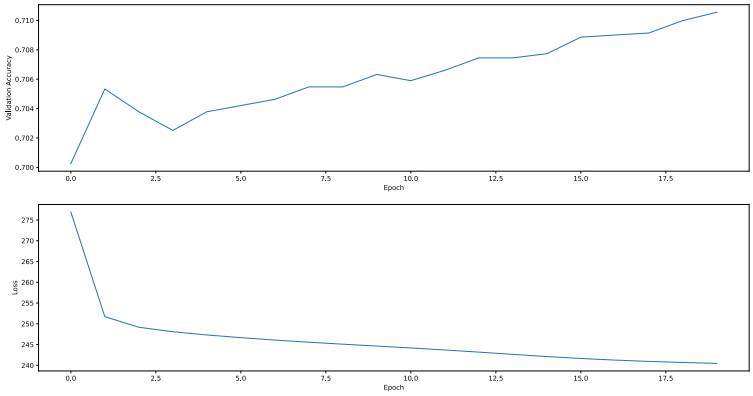

Figure 4: Custom Neural Network Training

Method	**Test Accuracy**
User Based KNN	68.471
Item Based KNN	66.892
Item Response Theory	70.769
Standard Neural Network(Part a)	70.307
Support Vector Machine	59.582
Boosted Gradient Tree(LightGBM)	67.0
Boosted Gradient Tree(XGBoost)	68.0
Custom Neural Network	71.521

Table 1: Final obtained results

condensed question representation using the question meta data instead of passing every question as an element in one-hot.

- It doesn't take into account any prior student information. We can increase the complexity of the model by using student metadata and fields like *premium_pupil* to accurately gauge the student representation.

4 Results

Table-1 summaries all the test accuracy obtained for different methods:

5 Conclusion

This report analyses various machine learning techniques on student-question dataset which is sub-sampled from Eedi dataset. In particular, we proposed a custom neural network architecture which is motivated by Item Response Theory, but not limited by any distribution. The custom neural network achieves state of the art accuracy as compared to various other methods. This demonstrates the fact that in order to correctly predict whether the given student will

answer a particular question, both student representation as well as question representation is necessary for accurate prediction.