Modeling MOOCs dropouts using various AI Models

Jyoti Sinha
University at Buffalo
jsinha@buffalo.edu

Abstract

Massive Open Online Courses(MOOCs) have caught everyone's attention for their potential to scale higher education. Lots of platforms such as Coursera, Udacity, edX etc. are appearing to provide a better learning experience to anyone and anywhere. More than 76 million learners from the world has joined the online courses to hone their skills in various fields. Despite their success, a major problem faced by MOOCs is the higher dropout. Joining rate to the course is very high as compared to the completion rate. In this paper, I am trying to explore the early identification of the students who are at a risk of not completing the courses. Several features based on the click streams and the interaction of the users with the various course modules are generated. Forum interaction and discussion modules are avoided in the study as only few enrolled students use forum and discussion module. Various Machine Learning and Deep Learning models such as Random Forest, Logistic Regression, Decision Tree, Naive Bayes, KNN, Multi-layer perceptron, Support Vector Machine, LSTM and Gated Recurrent Unit are used in the study to predict the dropout of the students .Comparison between various Machine Learning and Deep Learning models are performed to explore which classifier works better in this problem.

1. Introduction

Massive Open Online Course(MOOCs) has captured the attention of many people with its increasing popularity all over the world. Millions of students registering in the online courses and new courses are adding at a rapid rate suggests that in the incoming years, Online courses will supersize the traditional classroom programs. It aims to provide higher education to the world by offering online courses from various universities for free or at very low fees and has attracted millions of students from a variety of age groups, nationality and educational background. Despite its success, MOOCs face a major problem which is the low completion rate. Many people join the course but they don't complete the course which is the major hindrance of MOOCs success. For example, Table 1 shows the students participation in the Coursera MOOCS Discrete Optimization course which was offered by University of Melbourne in 2013.It shows a similar low completion rate as seen by the other MOOC courses. Only 795 students had completed the course out of 51,306 enrolled students which shows only a 1.5% completion rate. Only 27,649 students ever engaged in the lectures and assignments, quizzes.

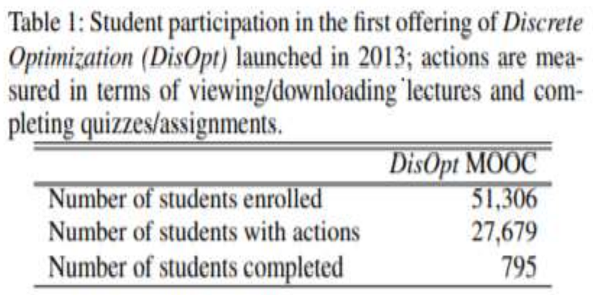


Table 1: Student participation in the first offering of *Discrete Optimization (DisOpt)* launched in 2013; actions are measured in terms of viewing/downloading lectures and completing quizzes/assignments.

	DisOpt MOOC
Number of students enrolled	51,306
Number of students with actions	27,679
Number of students completed	795

Table 1. MOOC Course statistics[1]

The goal of the study is to identify the students at the risk by identifying their click pattern across various modules of the course. Early Prediction can help the instructors to design the course or take any action to encourage the students to not drop the course. Data is taken from one of the largest MOOC dropout platforms of China which lasted for several months and students were engaged in various activities like participating in quizzes, assignments, watching/downloading the lectures and time spent on various modules of the course.
Exploration of predictive analysis on MOOCs is important but it is limited also as the data distribution across various offerings is not stationary. Different groups of people are enrolled in different courses and course materials(Lectures and assignments) also change over time which makes the offering non stationary. Proper intervention would help the students to know about their progress in the course and help them to persist in the course. So, by predicting the early dropout, the instructor can come with some interventions to prevent the dropout of students from the course.

2.Related Work

The major problem in MOOCs is the low completion rate.

One way to address this problem is to early predict the dropout and provide the timely intervention to avoid dropout. A few studies were made to predict dropout of students in MOOCs. Some researchers used the frequency of the forum post to measure the user's interest and retention in MOOCs.

Ramesh et al[3] predicted dropout on the basis of forum posts and forum clickstream data. They took the posting pattern and number of upvotes and downvotes in the discussion forum into consideration to predict the result. The researcher looked into the linguistic features, tone and nature of the post to determine how the student is using the course module. Various features are combined using Probabilistic Soft Logic(PSL) to conclude whether the user is going to complete the course or not. Sinha et al[4] analysed the clickstream on the video modules such as pausing , playing, forwarding, skipping the video lectures to predict the dropout. Jiang et al.(2014) used the user's one week data to predict the user's final performance. Kloft et al (2014) used only the clickstream data and implemented Support Vector Machine(SVM) to classify the weather the user will dropout or not.

Taylor, Veeramachaneni, and O'Reilly (2014) utilize crowd-sourced feature engineering (Veeramachaneni, O'Reilly, and Taylor 2014) used logistic regression to predict the dropout. Balakrishnan (2013) implemented Hidden Markov Models (HMMs) to predict the retention of the users and they extracted features mainly from video lectures and discussion forums.

3.Methodology

3.1 Problem Statement

Most of the work discussed above is contextual and used the forum to extract the features. It is believed that only 5% to 10% of the enrolled users use the forum (Rose and Siemens, 2014[1]). So, rather than using features from forum interaction, the study focussed on the count based features such as wiki, access, page navigation, problem accessed etc. MOOCs is challenging the traditional classroom education system but it is facing an issue of high dropout. If early prediction of user's dropout can be analyzed then using proper intervention, dropout of the users can be hampered. This can lead to a great success of the MOOCs. The goal of the project is to build various Machine Learning and Deep Learning models to predict the dropout of the users on the basis of clickstream data. Then, compare between various models and find the best model for this study.

3.2Data Description

The dataset that is used for this study is KDD Cup challenge 2015 [5]. The dataset was collected from XuetangX, which is on the largest MOOC platforms in China. Basically, the dataset is an event based and doesn't provide any contextual information. Lots of Feature processing is performed to draw the contextual information from the dataset. Features can be extracted as per our choice. In this project, I am using count based features rather than forum interaction.

The details of the dataset are as follows [6]

1)Train_Truth data - <enrollment_id, result>

In this dataset, the result i.e. whether the student has dropped or not is mentioned as per the Enrollment IDs.

a) Enrollment_ID:- Enrollment Id of the student
b) Result = 0 indicates that the user has completed the course and 1 indicates that user has dropped out the course.

2)Log_Train data - <enrollment_id, time, source, event, object>

In this dataset, each line corresponds to the login event. Each events contains the following information:- enrollment_id, time, source(server or browser), event and object.

a) Enrollment_ID:- Enrollment Id of the student
b) Time :- Time of the event
c) Source:- Event Source(server or browser)
d) Events:- In terms of event types, the dataset contains seven different types of events:-

- Problem:- Working on the course assignments
- Video :- Watching course videos
- Access:- Accessing other course objects other than assignments and videos
- Wiki:- Accessing the course wiki
- Discussion:- Accessing the course forum
- Navigate:- Navigate to the other part of the course
- Page Close:- Closing the web page

e) Object:- ID of the module

3)Enrollment_Train data - <enrollment_id,username, course_id>

In this dataset, each line is a course enrollment record with an enrollment_id, username and course_id indicating User, U enrolled in Course, C.

a) Enrollment_ID:- Enrollment Id of the student
b) Username:- Student_Id
c) Course_Id:- Course_ID

4) Date data - <course_id, from, to>

In this dataset, each line contains the timespan of each course. The timespan of each course for calculating dropouts is 10 days after the last day of the course. For Example, course C is from 2014.4.1 to 2014.4.30 in the given data, a user enrolled in the course C will be treated as a dropout if he/she leaves no record from 2014.5.1 to 2014.5.10.It has the following fields:-

a) Course_Id:- Course_ID
b) From:- The first day of the course in the log
c) To :- The last day of the course in the log

5) Object data - <course_id, module_id, category, children, object,start>

In this dataset, each line describes a module in a course with its category, children objects and release time. The module contains various online materials of the course like chapters, videos, problem sets etc. The modules are itself organized in a tree. For example, each course contains several chapters, each chapter contains several sections, and each section contains several objects(videos, problems etc)

a) Course_id:- The course to which the module belongs
b) Module_id:- The ID of the course module
c) Category:- The category of the course module
d)Children:- The children modules of the course module
e) Start:- The time at which the module was released to students.

3.2 Model Architecture

3.2.1. Long Short-Term Memory RNN (LSTM network)

Long Short Term Memory network is a special kind of RNN, capable of learning long-term dependencies. This makes it suitable for solving problems involving sequential data like a time series

The architecture of a LSTM consists of units called memory cells. The LSTM memory cell contains self-connections and special multiplicative units known as gates. These gates can regulate the flow of information and remember the temporal state of the memory cells. These gates can learn which data in the information are important to keep and throw away. In this way, the unimportant data from the information are discarded and the relevant information is passed in a long chain of sequence to make the prediction. There are three different types of gates that regulate the flow of information in an LSTM cell i.e Input gate, Output gate and Forget gate**.**

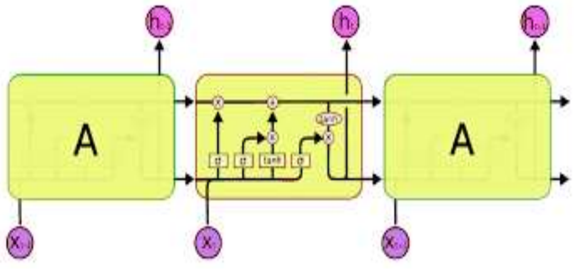
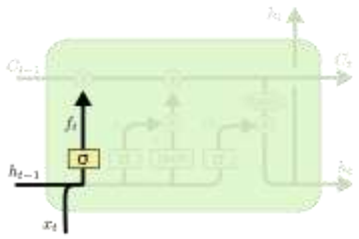


Fig1. LSTM Architecture

3.2.1.1. Forget Gate

This gate decides what information should be thrown away or kept. This decision is made by sigmoid layer called forget layer. It takes the current input, x_t and the previous hidden state h_{t-1} and outputs a number between 0 and 1. If the output is closer to 0 means to forget the data and closer to 1 means to keep the data.

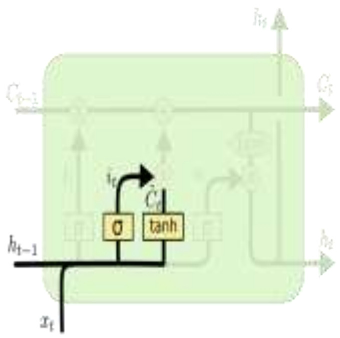


$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \ + \ b_f\right)$$

Fig2. Forget Gate[18]

3.2.1.2. Input Gate

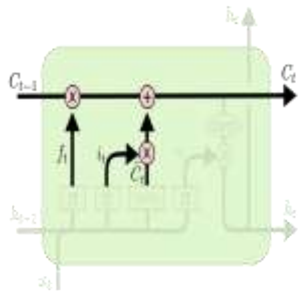
This gate update the cell state. It decides what new information are going next to store in the cell state. It has two part. First, the previous hidden state and current input are passed into the sigmoid function. This decides which values will be updated by transforming the values to be between 0 and 1. 0 means the information is not important and 1 means that the information is important. Next, tanh layer creates a vector of new candidate values, $\ddot{C}_t$, that are added to the state. Next, these two values are combined to create an update to the state.



$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \ + \ b_i\right)$$
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \ + \ b_C)$$

Fig3. Input Gate[18]

Next, the old cell state, C_{t-1} will be updated to the new cell state C_t. The old state C_{t-1} is multiplied by f_t, forgetting the things that were decided to forget earlier and then add $i_t * \ddot{C}$.

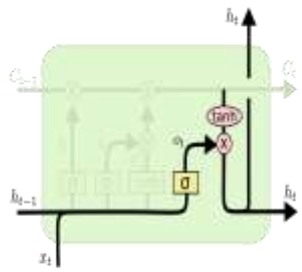


$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

Fig4. Cell State Update[18]

3.2.1.3. Output Gate

Output gate decides what to produce the output. The output are totally based on the cell state but will be in the filtered fashion. First, the current input and the previous hidden state are passed into the sigmoid function. Sigmoid layer will decide what part of the cell state are displayed as output. The cell state are passed to the tanh function and multiply it by the output of the sigmoid gate to output the values that are decided to do.



$$o_t = \sigma(W_o [h_{t-1}, x_t] + b_o)$$
$$h_t = o_t * \tanh(C_t)$$

Fig5. Output Gate[18]

3.2.2. Gated Recurrent Network (GRU)

GRU is similar to LSTM and are aimed to solve the vanishing gradient problem of standard RNN. It can be trained to keep the information long without removing the irrelevant information for prediction. It uses hidden state rather than the cell state to transfer the information. It has two gates:- Reset gate and Update gate.

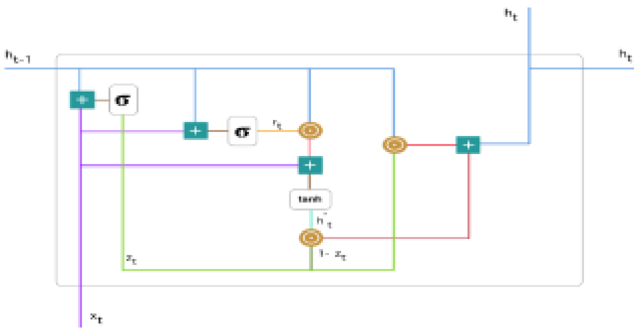


Fig6. Gated Recurrent Unit[10]

3.2.2.1. Update Gate

This gate decides the model to determine how much of the past information needs to be passed along to the future

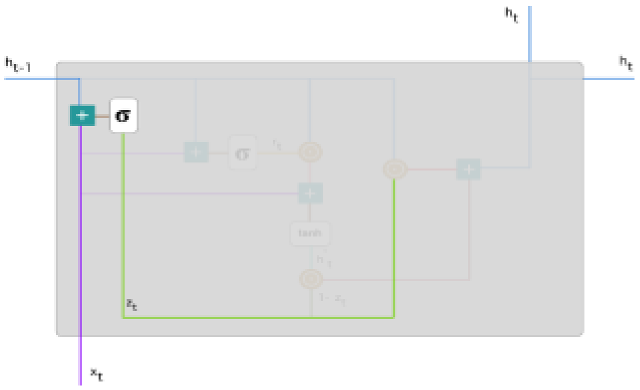


Fig7. Update Gate Mechanism[10]

Consider $W^{(z)}$ and $U^{(z)}$ are the weights of x_t and h_{t-1} then z_t is given by,

$$z_t = \sigma(W^{(z)}x_t + U^{(z)}h_{t-1}) \quad (1)$$

3.2.2.2. Reset Gate

This gate decides how much past information to forget.

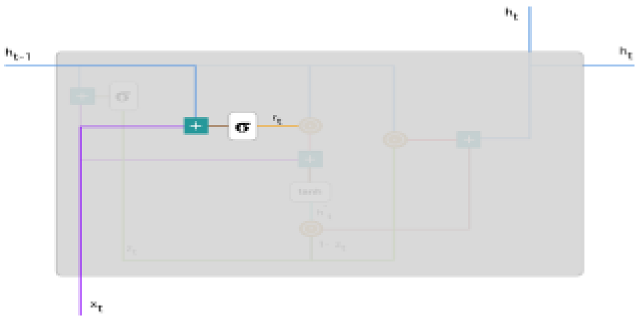


Fig8. Reset Gate Mechanism[10]

$$r_t = \sigma(W^{(r)}x_t + U^{(r)}h_{t-1}) \quad (2)$$

3.2.2.3. Memory content

Reset and Forget gate affects the final output. The new memory content will use the reset gate to store the information from past.

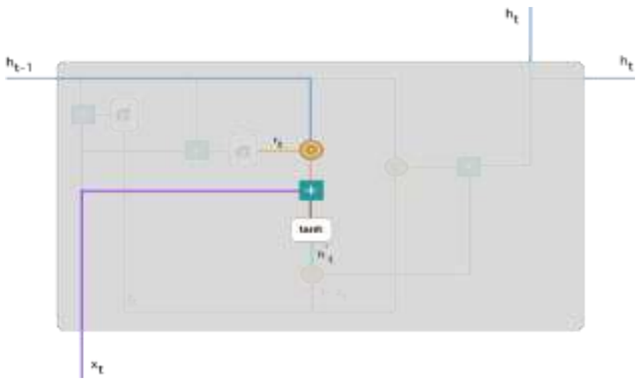


Fig9. Memory Content Mechanism[10]

$$h'_t = \ tanh(W\, x_t \ + r_t \ \odot U h_{t-1}\ ) \qquad (3)$$

3.2.2.4. Final memory at current time step

At the end the output is calculated which holds the information or the current unit and passes it down to the network. For this update gate is needed. It determines what information should collect from the current memory content and from the previous steps.

$$h_t = \ \ z_t \odot h_{t-1} \ \ + (1 - z_t) \ \odot \ h'_t \qquad (4)$$

GRU doesn't remove the new input every time but it keeps the relevant information and pass it to the next step of the network. It can also perform well in the complex scenarios.

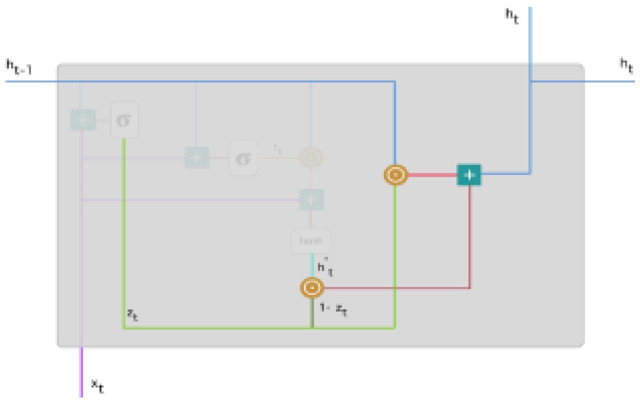
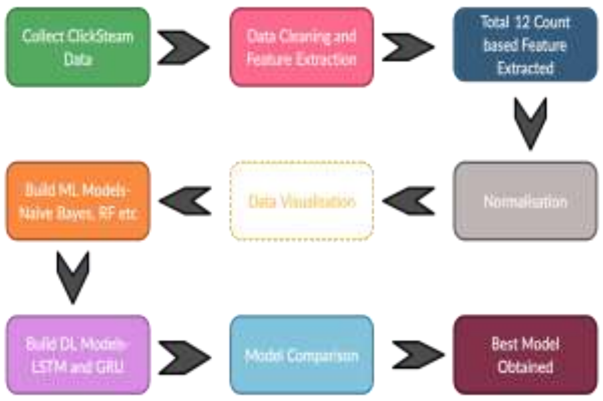


Fig10.Final Update Mechanism[10]

4.Experimentations

4.1. Feature Extraction and Modeling the predictive system



Collect ClickStream Data
Data Cleaning and Feature Extraction
Total 12 Count based Feature Extracted
Normalisation
Data Visualisation
Build ML Models- Naive Bayes, RF etc
Build DL Models- LSTM and GRU
Model Comparison
Best Model Obtained

Fig11. Flow Chart of Study

After processing the data sets, 12 features are extracted and th are used to implement various Machine learning as well Deep Learning models. These are as follows:- <enrollment_id,date,access,discussion,navigate,page_close,problem, video ,wiki, session, tot_time, result>

The log_train data is used mainly to form the clickstream attributes. Each Enrollment_Id was grouped to get a count of number of times a user has participated in the discussion forum, access course objects other than assignments and videos, navigated the page, closed the web page, solved the assignments, watched the video lecture and accessed the course wiki.

Session and total time are also the column of the final processed dataset. A session is the period of time when the user is active. All the click activity that is performed within a window of 50 minutes is the part of one session. If the user is inactive for fifty minutes and the session gets expired than any other further log after the window of inactive fifty minutes will be considered as the part of the next session. A session can be more or less than fifty minutes but it expires when the gap between two consecutive clickstreams in the record is greater than the fifty minutes.

The total time is the total time spent by the user for the course. Time is calculated by subtracting the last time of click stream and the first time of the clickstream each day. And the total time is calculated by adding the time spent by the user every day throughout the period of the course. [7].

In order to implement LSTM and GRU, the length of the timestamp should constant. For each enrollment id, there are 4 rows. The dataset that are used for Deep Learning models is obtained by merging two git hub and it had some errors. Most of the code for feature processing was taken from one github[6] and only two functions are taken from another github and I forget from which github it was taken as it was done way back. The git code had some bug, I fixed the error and used it as my dataset.

For ML Models, the dataset that was used for Deep Learning Model is modified by dividing the number of rows in the original processed dataset by 4 and summed all the events for each subpart. The total time and session will be the same for every timestamp of each enrollment ids.

4.2. Visualization

Various visualization techniques are applied to get the idea of the data and the relation between various features. The correlation between the extracted features are shown using the heatmap.
From the figure 12, it is clear that the Navigate and Page close are highly correlated. Also Session and total time shows higher correlation. And, Total time and enrollment ids are negatively correlated. Enrollment id is not correlated with any of the features. This is true also as the course completion can't be depend on the enrollment ids. Similarly, the time spent by users on various modules can't depend on the enrollment ids.

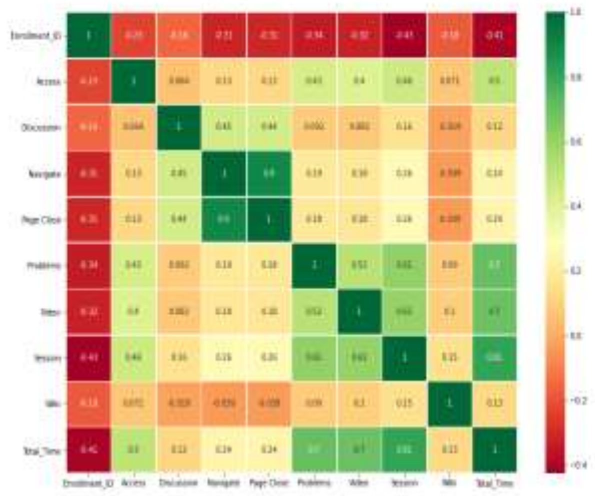


Fig12.Heat Map representing correlation among features

4.3 Model Implementations and Hyperparameters

After cleaning the data and performing feature extraction, a clean dataset is obtained on which various Machine Learning and Deep Learning Models are implemented. All implementations are done using libraries.
Data is fed to LSTM in batches. Batch size is decided by the user (32 in my case). It defines the number of rows from the dataset processed by the model before it updates its weights.
Without Normalization, the accuracy of LSTM was 74% using SGD optimizer with a learning rate of 0.01 and the losses were increasing. The model was getting overfitted without normalization. Thus, Normalization was performed. The columns of the original dataset on which LSTM was implemented without normalization was extracted and then normalization was done and finally a new data-frame having normalized columns is created. Normalized data-frame is passed in the data-gen for Implementing LSTM and GRU. By normalizing the dataset, accuracy of the models increased to 83.2% using the same optimizer(SGD) and with the same learning rate(0.01).
LSTM and GRU is implemented using regularizer and without regularizer for various optimizers to tune the model.
Early Stopping is also used to avoid overfitting issue.

4.3.1. Baseline Models:-

Six Machine learning models are used as baseline models in the experiments. These models are the basic Machine learning models and are commonly used in various Machine Learning applications. Baseline models are presented briefly and they focus on the dropout prediction in the experiment setting.

4.3.1.1. Support Vector Machine

SVM has been used in the dropout prediction of the user in the MOOC environment[17]. It is a Supervised Machine Learning algorithm and can be used for both classification and regression problems. In SVM, each data item is plotted in n-dimensional feature space. Then classification is performed by finding the hyperplane that differentiate the two classes. Linear SVM is used in the study which has given an accuracy of 81%. The parameter of C of the linear SVM is tuned via 5-fold cross- validation.

4.3.1.2.Logistic Regression

Logistic Regression has been implemented on the dropout prediction of students[7]. This classifier is used to solve the classification problem. Classifier models the posterior probability by observing the target class for a single trial as a logistic function of the weighted sum of the input features:-

$$p(y|x) = \frac{1}{(1+ e^{(-Wx)}} \qquad (5)$$

Where x is the input feature vector and y is the class label. The output of Logistic Regression is interpreted as probability because the range of logistic function is between 0 and 1[12]. Logistic Regression is giving an accuracy of 81..2% with the max_iteration of 5000.

4.3.1.3.Naive Bayes

Naive Bayes is a Machine Learning algorithm which is based on Bayes Theorem. It has been implemented on the dropout prediction of students in the MOOC environment[7]. It is based on “naive” assumption that the presence of a particular feature in a class is not related to the presence of any other feature[13] Naïve Bayes is giving an accuracy of 74% which is the lowest accuracy among all the Machine Learning as well as Deep Learning models that are used in the study. Naive Bayes have no hyperparameters to tune[16].

4.3.1.4. Random Forest

It is a supervised Machine Learning Algorithm. It is used for both classification and regression. It has been implemented on the dropout prediction of students in the MOOC environment[7]. The ensemble of decision trees usually trained with the bagging method leads to building a forest. Random Forest builds multiple decision trees and merges them together to achieve more stable and accurate prediction[14]. Hyperparameter of a random Tree are tried to change to get the better performance of the model. n_estimator is a hyperparameter of random forest which states the number of trees in the forest. When the n_estimator was set as None then it was giving 80.29% accuracy. As the value of n_estimator increased, the accuracy of the model also increased but on increasing the value of n_estimator beyond a certain limit, it negatively affected the performance of the model.

4.3.1.5. Decision Tree

Decision Tree is a supervised Machine Learning Algorithm which is used for both classification and regression problems. Decision Trees are constructed via an algorithmic approach which finds out the ways to split the dataset on different conditions. The main idea behind the decision tree is to create a model that predicts the value of a target variable by learning simple decision rules that are inferred from the data features. Depth of the tree, complexity of the rules and fitness of the models are directly proportional to each other[15]. The model was giving an accuracy of 74.16 % when the depth is set as None. Max_depth = None means that the nodes are expanded until all leaves are pure or until all leaves contain less than min_samples_split samples. Max_depth is a hyper parameter that controls the maximum depth of the tree.The accuracy of Decision Tree increases to 80.98% by setting the Max_depth as 3.

4.3.1.6.Multi-Layer Perceptron

Multi-Layer Perceptron is a class of feed forward artificial neural network. It consists of input layer, hidden layer and output layer. The model is giving an accuracy of 67% having hidden_layer_sizes = (256,256,256), ‘tanh’ activation and using ‘SGD’ as the solver. Adam refers to the stochastic gradient based optimizer.hidden_layer_sizes is used to set the size of the hidden layers. In the study, 3 hidden layers are used having 256, 256 and 256 neurons in each layer. On changing the hyperparameter and setting the hidden_layer_sizes = (256,128,64), ‘relu’ activation and using ‘SGD’ as the solver, the model performed well as compared to the previous setting giving an accuracy of 73.42%. On changing the solver from SGD to Adam the accuracy gets increased to 80.92%. Model’s performance has increased by tuning the hyper parameters.

5. Results

The table shown below describes the accuracy, precision and recall of various Machine Learning and Deep Learning Models

	Accuracy	**Precision**	**Recall**
LSTM_SGD_NoR	**0.790**	**0.791**	**0.897**
LSTM_Adam_NoR	**0.802**	**0.8036**	**0.888**
LSTM_Adam_WR	**0.827**	**0.837**	**0.919**
LSTM_SGD_WR	**0.663**	**0.674**	**0.78**
LSTM_RMSDROP	**0.834**	**0.858**	**0.892**
GRU_SGD	**0.791**	**0.818**	**0.860**
GRU_Adam_NR	**0.822**	**0.844**	**0.873**
GRU_Adam_WR	**0.827**	**0.850**	**0.897**
GRU_SGD_WR	**0.626**	**0.642**	**0.72**
GRU_RMSDROP	**0.844**	**0.881**	**0.896**
Logistic Regression	**0.8127**	**0.8107**	**0.8127**
Random Forest	**0.8098**	**0.8085**	**0.8098**
Decision Tree	**0.8098**	**0.8076**	**0.8098**
SVM	**0.8144**	**0.8120**	**0.8144**
Naïve Bayes	**0.748**	**0.7532**	**0.748**
MLP	**0.8082**	**0.8067**	**0.8082**

Table2.Comparison among metrics of various DL/ML Models

The diagram mentioned below shows the performance of various Machine Learning as well as Deep Learning Models. It is clear from the Graph that LSTM performs better as compare to the other models that I used for the study.

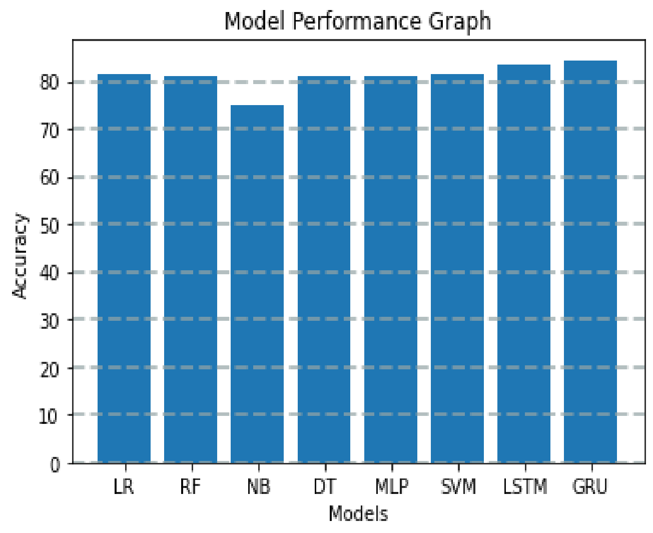


Fig13.Performance Graph of Various ML/DL Models

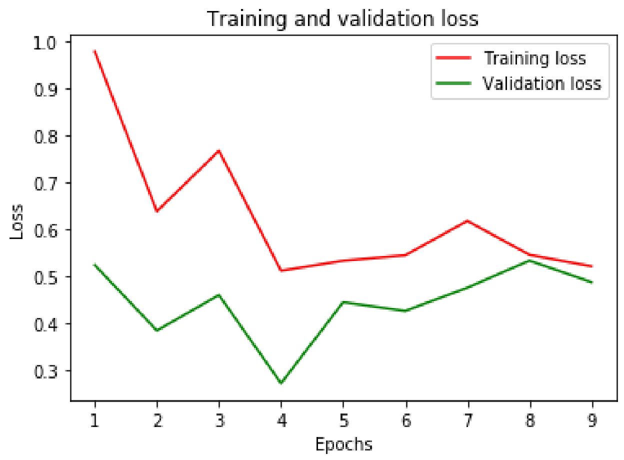


Fig 14. Training vs Validation Loss in GRU

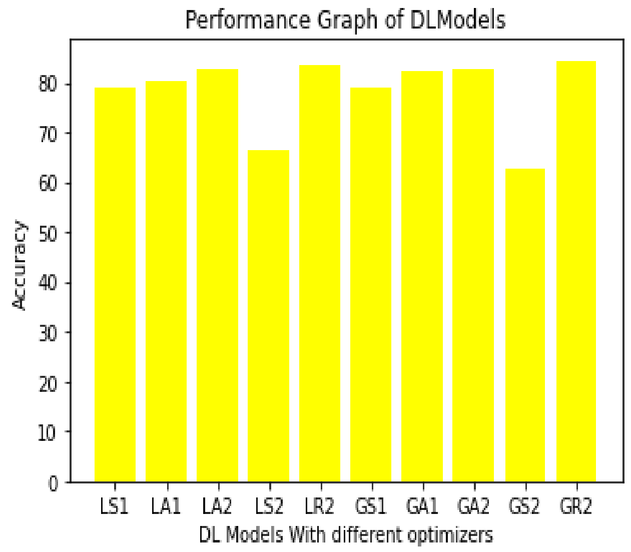


Fig14. Performance Graph of DL Models using various optimizers

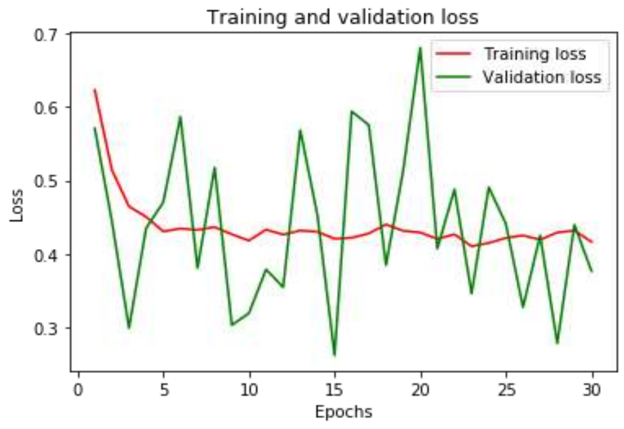


Fig 15. Training vs Validation Loss in LSTM

Where,
LS1 = LSTM Using SGD With No Regularizer
LA1 = LSTM Using ADAM With No Regularizer
LA2 = LSTM Using ADAM With Regularizer
LS2 = LSTM Using SGD With Regularizer
LR2 = LSTM Using RMSPROP With Regularizer
GS1 = GRU Using SGD With No Regularizer
GA1 = GRU Using ADAM With No Regularizer
GA2 = GRU Using ADAM With Regularizer
GS2 = GRU Using SGD With Regularizer
GR2 = GRU Using RMSPROP With No Regularizer

6.Conclusion

A model is proposed which is mainly based on click based features and counts to predict the dropout of the student's in the MOOCs environment. Several classification algorithm are used to predict if a user will drop the course or not using Long Short Tern Memory, GRU, Logistic Regression, SVM, Multi-layer perceptron, Decision Tree, Naïve Bayes and Random Forest by observing the clickstream data. LSTM performs best with an accuracy of 86% with respect to all other models that are used in this study. Deep Learning models performs better than the Machine Learning in this study.
In the future ,We still need to use some optimization technique to adjust the parameters of the model to improve the efficiency of the training.

References

[1]https://static.aminer.cn/upload/pdf/program/555048d445ce0a409eb71cc5_0.pdf

[2]https://dl.acm.org/doi/pdf/10.1145/3063955.3063959?casa_token=c8kCBknEQKUAAAAA:LbaeCSnAMg7wXwEnXKA0xcl4JaC4POXQBqcbxMyatZqBA9NMZ6EPUCPcdRsnNTW7KdgaO6JSoJxq0w

[3]Learning latent engagement patterns of students in online courses by Arti Ramesh, Goldwasser, Bert Huang, Hal Daume III

[4]Your clicks decides your fate: Inferring Information and Attrition Behaviour from MOOC video clickstream Interactions by Tanmay Sinha, Patrick Jermann, Nan Li, Pierre Dillenbourg

[5]http://moocdata.cn/challenges/kdd-cup-2015 https://github.com/TythonLee/kddcup-2015/blob/master/DataDescription.txt

[6]https://github.com/Siddhant085/modelling-MOOC-dropout/blob/master/code.py

[7]Arti Ramesh, Dan Goldwasser, Bert Huang, Hal Daume III, and Lise Getoor. 2013. ´Modeling learner engagement in MOOCs using probabilistic so‰ logic. In NIPS.Workshop on Data Driven Education, Vol. 21. 62

[8]https://learninganalytics.info/index.php/JLA/article/view/4212/4429

[9]http://moocdata.cn/challenges/kdd-cup-2015

[10]https://towardsdatascience.com/understanding-gru-networks-2ef37df6c9be

[11]https://towardsdatascience.com/illustrated-guide-to-lstms-and-gru-s-a-step-by-step-explanation-44e9eb85bf21

[12]https://aspiringminds.com/drupal/pages/assess/2015/papers/ICDM-ASSESS.pdf

[13]https://www.analyticsvidhya.com/blog/2017/09/naive-bayes-explained/

[14]https://builtin.com/data-science/random-forest-algorithm

[15]https://www.hackerearth.com/practice/machine-learning/machine-learning-algorithms/ml-decision-tree/tutorial/

[16]https://datascience.stackexchange.com/questions/73506/hyper-parameter-tuning-of-naivebayes-classier

[17]B.Amnueypornsakul, S. Bhat, and P. Chinprutthiwong, “Predicting attrition along the way: The UIUC model,” EMNLP 2014, p. 55, 2014

[18]https://colah.github.io/posts/2015-08-Understanding-LSTMs/