2020-2021

Survey On Machine Learning in Learning Management Systems(LMS):

RAJIV GANDHI COLLEGE OF ENGINEERING, RESEARCH AND TECHNOLOGY, CHANDRAPUR

Bachelor of Technology in Computer Science Engineering

Submitted by:
Pranay A. Haramwar CSEB530
Deep R. Bailke CSEB522
Dipak A. Vaidya CSEB524

Under the Guidance of Prof. Rupa ma'am

ABSTRACT

In the era of globalization, knowledge becomes necessary. Today it is very easy to share and disseminate knowledge due to evolution in technology. In this paper, we have included study of Machine learning algorithms which are used in Learning management systems(LMS) like Moodle, which plays a major role in it. To understand learners and analyse an online learner's past performance and combine it with new information to predict the outcome. As such, you're able to set online learners on the right path based on skill, performance, or knowledge gaps. This is all based on a variety of predetermined criteria, such as current organizational objectives, goals, and job duties. Learning Analytics in an LMS covers many different aspects: finding students at risk of abandoning a course, predicting students failing a quiz or students not reaching the end of a lesson in less than 15 minutes. A prediction model to detect late assignments submissions to analyse an online learner's past performance and combine it with new information to predict the outcome. Our prediction is that which student having high risk of dropping out from the particular course.

INTRODUCTION

LMS is a piece of software that manages, analyses and runs educational courses and training programs. Also includes student's registration curriculum management, skill and complementary management and features. Most modern LMS packages are web-based.
Top open source LMS are Blackboard, Schoology, SAP Litmus, Nocebo, Talentless, Edmodo, Absorb LMS, Moodle. What can I do with an LMS? With an LMS, you can gather all your learning materials in one place and make them available to all the learners in one click. You can create different groups of learners according to their function or job position. A wide range of different organizations from high schools to large corporations can benefit from using learning management system by reducing expenses for things like

- Transport and accommodations
- Instructors' salary
- Learning material print

The right LMS helps you to Save time, set unified learning standards, Assess and develop employees' knowledge regularly Large, medium and small! Any kind of business can benefit from an LMS. Access to learning materials anywhere, from any device and an unlimited number of times to refresh knowledge anytime they need to, as well as an opportunity to collaborate with each other during the learning process.

The most significant role that Machine Learning plays in eLearning is personalization. This is achieved through more effective data analysis and automation. An LMS that uses Machine Learning is able to access user data and use it to improve the eLearning experience. However, it can also be fully integrated with your HR systems to analyse learner data and pinpoint trends with greater efficiency. This allows you to identify areas for improvement based on analytical patterns and pre-set algorithms.

For example, the LMS uses algorithms to analyse an online learner's past performance and combine it with new information to predict the outcome. As such, you're able to set online learners on the right path based on skill, performance, or knowledge gaps. This is all based on a variety of predetermined criteria, such as current organizational objectives, goals, and job duties.

So, here we are studying algorithms which used in LMS (Learning Management System) in MOODLE to analyse an online learner's past performance and combine it with new information to predict the outcome. Our prediction is that which student having high risk of dropping out from the particular course.

BACKGROUND

MOODLE:

Moodle is a learning platform designed to provide educators, administrators and learners with a **single robust, secure and integrated system** to create personalised learning environments. You can download the software onto your own web server or ask one of our knowledgeable Moodle Partners to assist you.

Moodle is built by the Moodle project which is led and coordinated by Moodle HQ, which is financially supported by a network of over 80 Moodle Partner service companies worldwide.

Some Benefits of Moodle:

- All-in-one learning platform
- Highly flexible and fully customisable
- Scalable to any size
- Robust, secure and private
- Use anytime, anywhere, on any device
- Extensive resources available
- Backed by a strong community

By default, the Moodle API only provides basic reporting capabilities. Its reporting tools allow users, mainly teachers, to access course activity logs, to group data by students or by activity, and to generate graphs for aggregated data. Analytics tools for extracting insights from activity logs and for visualization are not available in the default core system of Moodle; however, a wide range of plugins that are compatible to Moodle can be separately and easily installed.

• The Moodle Analytics API12 written in PHP is responsible for generating labelled and unlabelled CSV13 files from Moodle's database contents. The CSV format is chosen because it is a simple text format that is portable on all platforms. For all prediction models, multiple CSV files are generated to capture information about the courses and the students.

Machine Learning backend are responsible for processing these files. They process the labelled and unlabelled CSV files. These backends can be written in any programming language.

Machine Learning backend. In the testing mode, the Machine Learning backend split the CSV files into training and testing sets and evaluate the prediction accuracy. In the production mode, Machine Learning backend are trained with finished courses data and return predictions for ongoing courses.

The Moodle core includes two Machine Learning backend: A Feed-Forward Neural Network written in Python using the Tensor flow framework and a Logistic Regression classifier written in PHP for the Moodle sites where installing Python is not an option. New Machine Learning backend can be plugged on the framework.

Supervised Learning Abstraction

The framework is an abstraction of a Supervised Learning problem. In this section we list the main elements that compose the framework and how they are mapped to the typical elements needed in a Supervised Learning task.

Each prediction model is composed of a single Analyser, a single Target, a set of Indicators, one Time Splitting Method and a single Machine

Learning backend. All these elements are implemented as PHP classes14 although Machine Learning backend can be written in other programming language. The whole Supervised Learning framework has been designed so that the elements described below are reusable and extensible independently across different prediction models.

Analysers: Each Analyser is responsible for defining the subject of the model. They select and pass all the Moodle data associated to these subjects to Targets and Indicators (described right below).

The following analysers are included in the framework and can be reused by researchers to create new prediction models:

• Student enrolments: The subject of the model are students in a course.

• Users: The subject of the model are site users.

• Courses: The subject of the model are courses.

Targets: They are the key element of a prediction model. They represent the labels of a Supervised Learning dataset and define the event of interest. Obviously, Targets depend on Analysers, because Analysers provide the subjects that Targets need for calculating the label. The framework includes an identifying student at risk of Target. Here are a few more examples of Targets in prediction models and their associated subjects:

• Identifying spammer user: The subjects of the model are site users.

• Classifying ineffective course: The subjects of the modelare courses.

• Assessing difficulties to pass a specific quiz: The subjects of the model are quizzes

Indicators: They represent the features in a Supervised Learning problem. Indicators are responsible for performing calculations on Moodle data. They are calculated for each subject using data available in the time period defined by the Time splitting method (described right below). They were designed to avoid normalisation issues so the CSV file features generated from indicators always have values in the range [−1, 1]. Indicators are in one of the following categories:

• Linear: The indicator values are floating point numbers in the range [−1, 1]. An example would be "the weight of quiz activities in a course".

• Binary: The indicator values are Boolean values ∈ {0, 1}. An example would be "has this student completed all activities in the course?".

• Discrete: The indicator values are a closed list of values. The framework one-hot encodes the list of values and generates N features with values ∈ {0, 1}. An example would be "How often do this student access the course?", with values "never", "in monthly basis" and "in weekly basis”.

The framework includes a set of indicators that can be used in new prediction models. Not all indicators can be included in any model. E.g. student-related indicators could not be calculated if the predictions subjects are courses. Below are some of the indicators included in the framework:

• The number of clicks made by the student in a course. It is implemented as a linear indicator.

• Student posts to forum activities in a course. It is implemented as a linear indicator.

• Was the course accessed after the course end date? It is implemented as a binary indicator.

• Was the course accessed before the course start date? It is implemented as a binary indicator.

Time splitting methods: They define when the framework should generate predictions and the time period that should be used to calculate the indicators. The Moodle core includes a few time splitting methods that researchers can use in their prediction models. For example:

• Split the course duration into four parts and generate a prediction at the end of each part.

• Generate a prediction one week before each assignment's due date and generate a second prediction two days before the assignment due date.

Machine Learning backend. In the testing mode, the Machine Learning backend split the CSV files into training and testing sets and evaluate the prediction accuracy. This process is described in more detail.

Machine Learning backend are trained with finished courses data and return predictions for ongoing courses.

The Moodle core includes two Machine Learning backend: A Feed-Forward Neural Network written in Python15 using the Tensor flow framework16 and a Logistic Regression classifier written in PHP for the Moodle sites where installing Python is not an option. New Machine Learning backend can be plugged on the framework

MACHINE LEARNING ALGORITHMS:

The CSV files are consumed by Machine Learning algorithms. The Machine Learning backend of the framework include two classifiers. These classifiers are described in detail in the following subsections.

Logistic Regression:

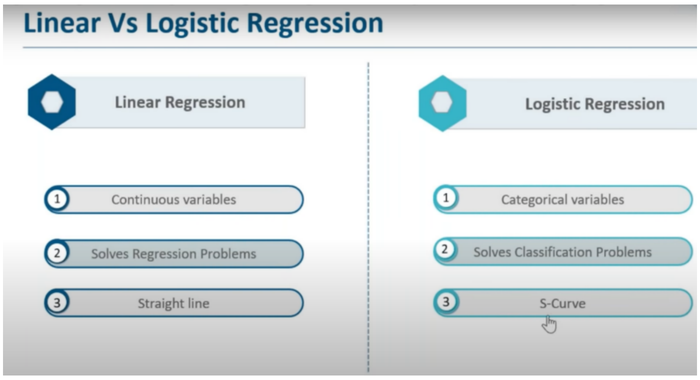
Logistic regression is a supervised learning classification algorithm used to predict the probability of a target variable. The nature of target or dependent variable is dichotomous, which means there would be only two possible classes.

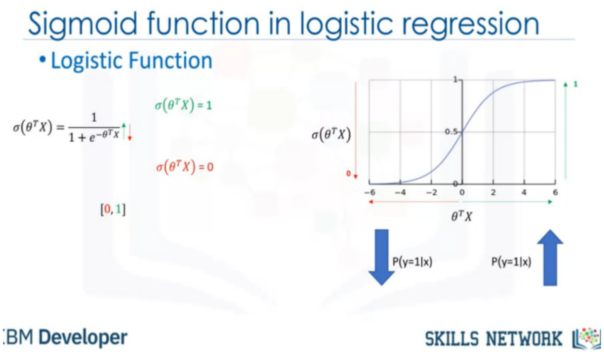
Logistic Function: Logistic regression is named for the function used at the core of the method, the logistic function.

The logistic function, also called the sigmoid function was developed by statisticians to describe properties of population growth in ecology, rising quickly and maxing out at the carrying capacity of the environment. It's an S-shaped curve that can take any real-valued number and map it into a value between 0 and 1, but never exactly at those limits.

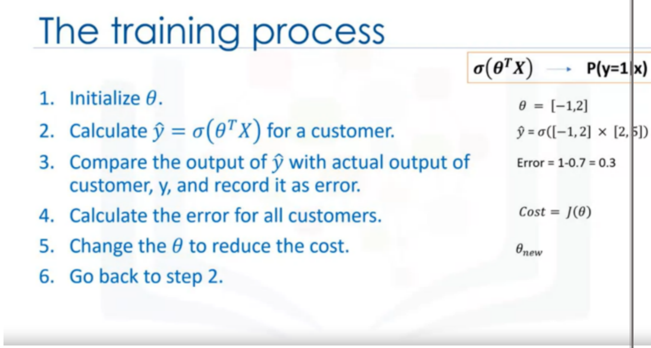
$$1 / (1 + e^{-value})$$

Where e is the base of the natural logarithms (Euler's number or the EXP() function in your spreadsheet) and value is the actual numerical value that you want to transform. Below is a plot of the numbers between -5 and 5 transformed into the range 0 and 1 using the logistic function.

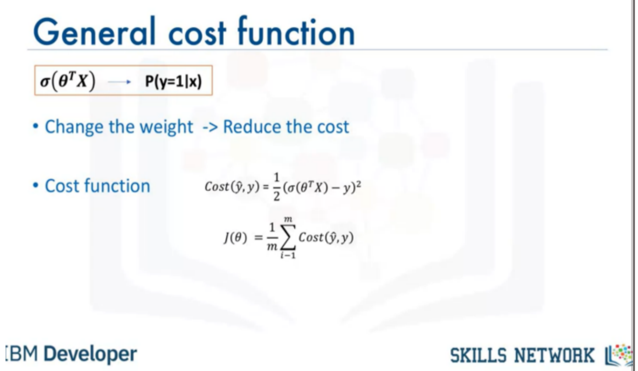


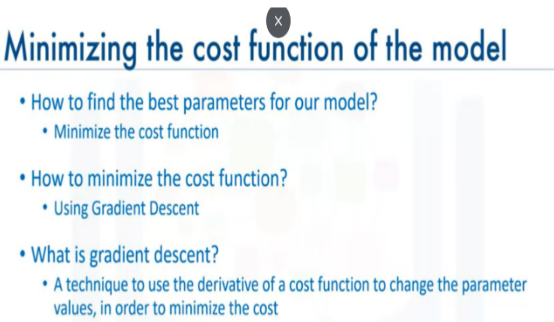


The PHP Machine Learning backend uses a Logistic Regression binary classifier to perform its predictions. In Logistic regression, a Logistic function hθ (x) = 1/ (1 + exp (−θT x)) is applied to the feature vector x to produce a value in the range (0, 1). The parameter θ ∈ Rn is a vector of weights that need to be learned. The function behaves like a thresholding function with a soft boundary. Because of the range the output value hθ (x) can be interpreted as the probability whether x belongs to the target class in a two-class classification problem. Logistic Regression tries to find the best fitting model for the relation between features and their labels by optimising the following cross-entropy cost function:

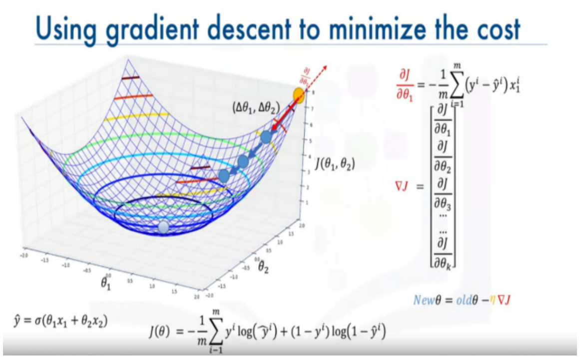


$$J(\theta) = \left(-\frac{1}{m}\right) * \left[\sum_{i=1}^{m} y_i \log(h_\theta(x_i)) + (1 - y_i)\log(1 - h_\theta(x_i))\right]$$

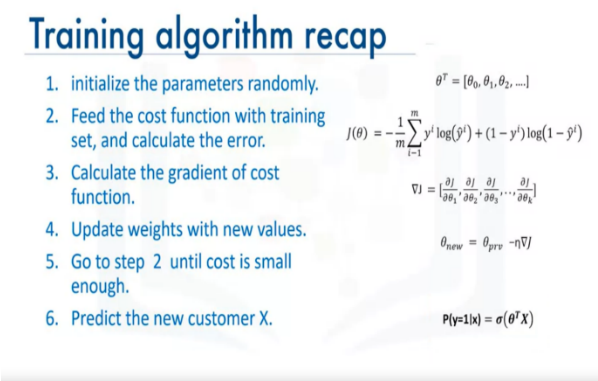




Where m is the total number of samples, xi denotes the it h sample and yi denotes the corresponding ground truth label. The cost function gives the error between a set of weights where all samples' fit perfectly and the labels predicted by the algorithm. The parameter θ is updated according to the gradients of the cost function. Gradient Descent is the common algorithm used to optimise cost functions in Machine Learning. It is iteratively used to update the set of weights θ using the following formula:



$\theta_j := \theta_j - \alpha * (\partial/\partial\theta)\, J(\theta)$, for $j = 1, . . . ,n$.



where α is a constant called learning rate which defines how significant θ updates are and ∂ /∂θj J (θ) is the partial derivative of J (θ). Logistic Regression is an effective algorithm for binary classification although some problems would arise when the number of features increases: The algorithm becomes computationally very expensive and it is very easy for the algorithm to over fit the training data. Logistic Regression popularity is decreasing in favour of other Supervised Learning algorithms.

Feed-Forward Neural Network:

Artificial Neural Networks (ANN) mimic biological Neural Networks. They are networks containing relations between units, also called neurons. Neural Networks organise units into layers. The connections between neurons in different layers are represented as a set of weights θ. The Neural Network improve its accuracy during training by back propagating [9] the difference of the calculated label of a sample versus the real value to the network's previous layers, updating its weights. The connections between different layer neurons become stronger or weaker after the back-propagation process.

Feed-forward Neural Networks contain a first layer of units called input layer, with as many units as features in the input CSV file. They then contain a number of hidden layers $l \geq 1$ and a final output layer with as many units as different labels the input CSV file has. The Neural Network included in the framework contains one hidden layer with ten hidden units. The network training process is composed of two different steps: feed-forward and back-propagation.

The feed-forward process computes the predicted labels for a given set of samples by multiplying input features values by the matrices that connect different layer units. The following formulas use a set of samples of size N. In $z1 = \theta 1 \cdot x + b$ we multiply the input features matrix for the weights matrix θ1 that connects the input features x with the first hidden layer units, adding a bias b. z1 is multiplied by a non-linear activation function g(z). Different activation functions can be used, some examples are g(z) = sigmoid(z), g(z) = tanh(z) or g(z) = relu(z). The activated matrix (one vector for each sample) in the following layer is therefore expressed as a1 = g(z1). This calculation is repeated until the output layer is reached. It can be generalised as al = g(θl · al−1). The softmax function can be used as the output layer activation function as it returns a ∈ [0, 1] value for each possible label. The softmax formula is expressed as follows.

$$softmax_i = e^{alt} / \sum_j e^{alj}$$

where i and j are each of the possible labels .We finish the forward pass by calculating the error using the cross-entropy cost function J (θ) given in Eq. 2.During back-propagation we minimise the cost function J (θ) by updating the weights that connect neurons in different layers. The updated value depends on the partial derivative of the error with respect to each of the network weights in the previous layer ∂J (θ)/∂θij and the learning rate α. Weights are updated using the Delta rule, expressed as follows:

$$\Delta\theta_l = -\alpha * (\partial J(\theta)/\partial \theta l)$$

The partial derivatives calculations are based on the chain rule which allows us to compute a derivative as the composition of two or more functions, in our case:

$$\partial J(\theta)/\partial\theta_l = \partial J(\theta)/\partial g(zl) \cdot \partial g(zl)/\partial zl \cdot \partial zl/\partial\theta l$$

where θl represents the vector of weights in layer l and g(z) the activation function. ∂zj/∂θl = al , the partial derivative of the activation output is the derivative of the activation function. E.g. Sigmoid function g′ (zl) = g(zl) ˙(1 − g(zl))

Finally, ∂J (θ)/∂g(zl) calculation depends on the value of l as we need to consider all the layers from l to the output layer. We calculate deltas δl for each layer starting from the output layer one, which is δoutput = (yˆ − y), where y and yˆ are, respectively, the actual and predicted labels (both are vectors usually). With δout put we can go backwards calculating the previous deltas until the first hidden layer with δl = θlT δl+1 · g′ (zl). The partial derivative of any weight in the network is therefore represented as:

$$\partial J(\theta)/\partial\theta l = \delta l{+}1 \cdot g(zl).$$

Testing mode

we described that the framework generates a labelled CSV file from Moodle site contents based on the prediction mode defined by the researcher. we describe how the framework evaluates the defined prediction model using Machine Learning techniques.

The first thing the evaluation process detects are the CSV files with highly skewed classes. The provided Machine Learning backend do not cope well with really unbalanced classes. Even if the Machine Learning backend reports high accuracies the recall or the precision will probably not be high which would lead to a low predictive power. Prediction models with highly skewed classes are not further evaluated in the current framework.

The Machine Learning algorithm is trained with the training dataset and the test dataset is used to calculate the Matthews' correlation coefficient of the prediction model. The Matthews' correlation coefficient is a good evaluation metric for binary classification problems because it takes into account both true positives and true negatives.

To calculate the Matthews' correlation coefficient we first fill out the confusion matrix with the predicted results and the test dataset labels. The confusion matrix looks like this:

$TN\ FN\ FP\ TP$

where TP = True positives, FP = False positives, TN = True negatives and FN = False negatives. The Matthews' correlation coefficient (MCC) formula is given by:

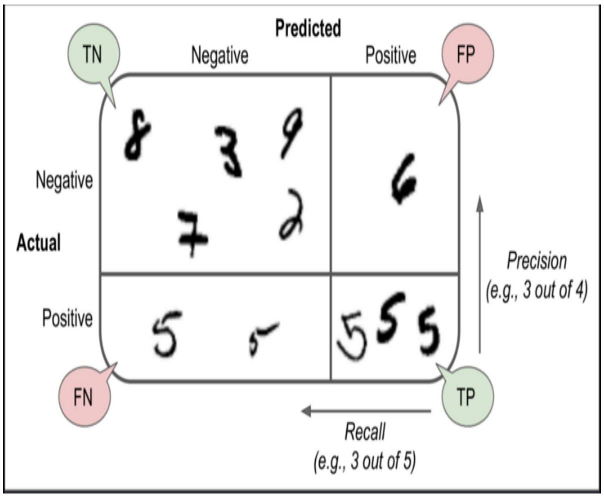
MCC = (TP × TN − FP × FN) /√[(TP + FP) × (TP + FN) × (TN + FP) × (TN + FN)]

The entire process is automated. So this training and MCC calculation process described above is repeated multiple times and the MCC for each iteration is be recorded. The average MCC and standard deviation is later calculated from all iterations.

An average MCC of 0 indicates that the model is no better than a random model ,a negative value indicates an inverse relation and a positive value a positive relation. Therefore, the higher the MCC value is, the better the model is at predicting. The standard deviation of all the calculated MCC will be used to detect variations in the coefficients. Small variances in each iteration's MCC can be expected because the CSV file contents are shuffled before each iteration, but high variances are a good sign that the CSV file is not large enough to guarantee that the evaluation results are reliable. The average MCC is automatically converted to a score in the range [0, 100] using the following formula:

score = (MCC + 1) / 2

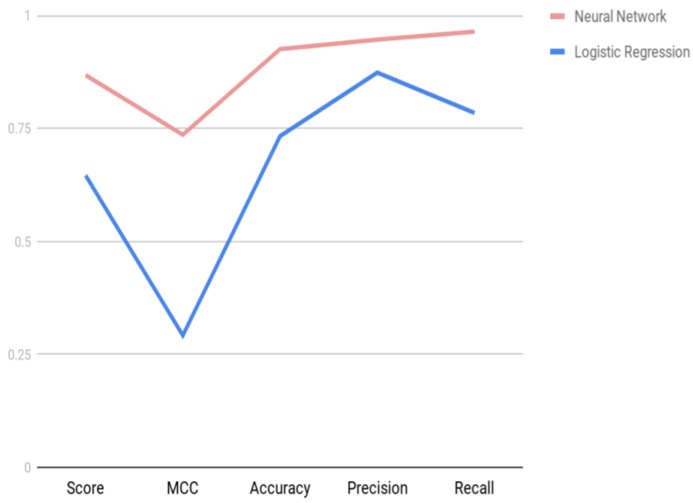
The computed score is provided to the researcher as a quality measure for the prediction model.



Precision:
precision = (TP) / (TP+FP)
TP is the number of true positives, and FP is the number of false positives.
A trivial way to have perfect precision is to make one single positive prediction and ensure it is correct (precision = 1/1 = 100%). This would not be very useful since the classifier would ignore all but one positive instance.

Recall
recall = (TP) / (TP+FN)

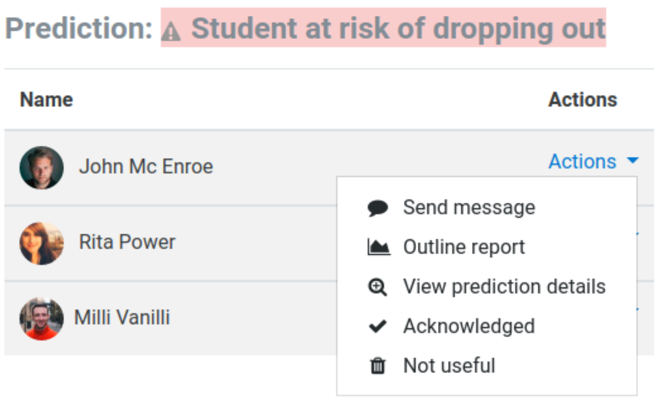
Algorithms:	Score	MCC	Average accu
Neural Network	0.868	0.736	92.56%
Logistic Regression	0.646	0.292	73.30%



Prediction model evaluation results

Test results:

To test the prediction model we used 8 finished anonymised MOOCs with a total of 46,895 students. Results provided by the framework testing mode are shown in Table and Figure . Our test results show that the proposed at-risk students model gave an average prediction accuracy of 92.56% using the Neural Network described and an average prediction accuracy of 73.30% using the Logistic Regression classificator described. The Neural Network appear to be better at modelling the relation between the input CSV file features and the label, probably due to the extra set of weights in the hidden layer which should allow the Neural Network to model more complex relations between features.



Actionable insights generated by an at-risk student's model.

Once the production mode is enabled, predictions can be generated for ongoing courses The actionable insights generated by this at-risk student's model are shown in Figure.

CONCLUSIONS

The algorithms cover prediction models like at-risk students and allows researchers to evaluate them to determine if they are effective or not. A prediction model to detect late assignments submissions to analyse an online learner's past performance and combine it with new information to predict the outcome. Our prediction is that which student having high risk of dropping out from the particular course.

The Supervised Learning framework presented in this paper is part of the Moodle core from version 3.4.0 onward, but is disabled by default as it requires sufficient computer power to run. Moodle is an open source Learning Management System so the exact number of unregistered users is unknown. Given that there are more than 130 million registered users around the world and given that approximately 50% of more than 100,000 registered Moodle sites use Moodle 3.4.0 or above 18 (June 2018), it would not be too unrealistic to claim that the framework is used by millions of users.

REFERENCES

1) David Monllaó Olivé The University of Western Australia Crawley, WA, Australia Moodle Pty Ltd West Perth, WA, A Supervised Learning framework for Learning Management Systems Australiadavid.monllaoolive@research.uwa.edu.au
2) https://www.researchgate.net/publication/257948587_A_survey_paper_on_e-learning_based_learning_management_Systems_LMS
3) https://elearningindustry.com/machine-learning-artificial-intelligence-future-elearning
4) https://machinelearningmastery.com/logistic-regression-for-machine-learning/
5) https://docs.moodle.org/35/en/Course_reports
6) https://www.moodlerooms.com/resource/x-ray-learning-analytics/
7) https://moodle.org/plugins/local_intelliboard