An Artificial Neural Network Approach to Student Study Failure Risk Early Warning Prediction Based on TensorFlow

Mi Chunqiao[1,2](✉), Peng Xiaoning[1,2], and Deng Qingyou[1]

[1] Huaihua University, Huaihua 418000, Hunan, People's Republic of China
michunqiao@163.com, hhpxn@163.com, dengqingyou@163.com
[2] Key Laboratory of Intelligent Control Technology for Wuling-Mountain Ecological Agriculture in Hunan Province, Huaihua 418000, Hunan, People's Republic of China

Abstract. Higher education is now facing big challenges about low course study completion and graduation degree completion rate, for which student study failure in course is the main reason. However, the failure in course study is a comprehensive result of various factors and is characterized by uncertainty. Artificial neural network approach is advantageous for dealing with this issue, and in this study we provided such an approach to predicting student study failure risk for early warning in course study based on the TensorFlow platform. In our model, for each student, four input variables: (1) times of login onto the online study system; (2) times of downloading study resource; (3) attendance earned points; and (4) assignment earned points, and one target variable: the final course grade point were collected for network training. At last, by validating with the observed data, consistency is shown between our predicted results and the actual observed data, which indicates that the employed model is a promising approach for identifying at-risk student. It is helpful for educators to timely apply corresponding strategic pedagogical interventions to help at-risk students avoid academic failure, and to effectively improve early warning education management.

Keywords: Artificial neural network · Study failure risk
Early warning prediction · TensorFlow

1 Introduction

Study failure is now a very common phenomenon in college and university education, which can results in failed graduation and unsuccessful job-hunting. However, in today's college and university, there are many existing problems in early warning education, such as very late warning time (usually long after the final examination, or even in the next semester), only single evaluation factor (always only focus on evaluation of the final examination grades), very outdated technology (often done only by hand), etc. So how to quantitively identify the risk of study failure by modern information technologies such as data mining techniques in early time is a very important issue in college education management. Fortunately, with the further development and application of modern information technology in education, at present many varieties of educational data about student study process have been collected and stored in college and university. Besides, many new data analysis techniques such as big data analysis, learning analytics and data mining methods are becoming more and more widely available in education applications. So it is becoming feasible and a new trend to do further study on early warning risk identification based on data science and machine learning methods.

Some related initial attempts about using data mining methods for prediction of academic performance or study risk can be traced back to the very begin of 2000s [1]. For example, Chen et al. [2] identified potential weak students and profiled student groups based on methods including association rules and decision trees. Morris et al. [3] predicted student final grades and the successful completion of online courses using the method of discriminant analysis. Macfadyen and Dawson [4] detected underperforming students with methods like logistic regression and network analysis. Jay et al. [5] studied the early indicators of student success and failure. Geraldine et al. [6] provided a methodology to classify students using interaction data and predicted first-year students at risk of failing. Kevin and David [7] presented a classification system for detection of poor performers before the end of the course. These related studies provided a useful foundation for early warning prediction of study failure risk and good insights for identify at-risk students. However, the number of studies that can be able to make concept transit into implementation is still few, and there is still little report on application of artificial neural network approach in study failure risk prediction. In addition, most of the current researches mainly focus on modeling and analysis of static historical educational data, while dynamic analysis based on learning process data before the completion of course final examination is insufficient.

The goal of this study is to predict student study failure risk in early time using artificial neural network (ANN) model with TensorFlow platform and Python language, so as to timely identify the at-risk students and improve the efficiency and effectiveness of early warning education in today's college and university.

2 Materials and Methods

2.1 Data Description and Preprocessing

In this study, totally 391 students during the course of "Introduction to Computer Science", "Fundamentals of Computer", "Software Engineering", and "Software Architecture" in 2016 were chosen as study samples. They were from eight different majors. Among the 391 samples, 296(3/4) samples were chosen by random for the network model training, and the remaining 95(1/4) samples were used for validation.

For each student, the daily study process data during the courses were collected from an online study system developed by the authors. These data were used to calculate the value of input variables which could affect student's final performance. The accuracy of student study failure risk prediction depends on the significance of the chosen input variables with respect to their effects on final course grade. There are many factors related to student study performance in a course, of which the student's participation is the most important one that reflects the student's attention and effort spent in the course. In addition to that, the earned points in assignments can reflect student's mastery of course knowledge, and the attendance performance is related to the time and attention that the student spent in the course. Therefore, for each student the following four input variables were chosen: (1) times of login onto the online study system (denoted as X1); (2) times of downloading studying resource (denoted as X2); (3) attendance earned points (denoted as X3); and (4) assignment earned points (denoted as X4). Besides, for each student, the final course grade point was also collected as target variable (denoted as Y), which uses the hundred percentage point system with 60 points as the passing grade. In order to quantify student study failure risk, a three-level risk classification scheme of red (R), yellow (Y), and green (G) was also developed, in which R (serious risk) means actual final grade < 50 points, Y (moderate risk) means 50 $\leq$ actual final grade < 60 points, and G (no risk) means actual final grade $\geq$ 60 points.

In practice, the magnitude of different input variables with different units may differ very greatly. So in order to make a balance, the original data have to be preprocessed at first. In our study, all original data including input and output variables were normalized at first using the following expression (1). Where Z is the original data; Z_r is the normalized value; Z_{max} and Z_{min} are the max value and min value of the original data respectively.

$$Z_r = \frac{Z - Z_{\min}}{Z_{\max} - Z_{\min}} \tag{1}$$

2.2 ANN Model and TensorFlow Implementation

The ANN model is a powerful tool in many fields, especially the well-known backward propagation algorithm. A backward propagation network (BPN) with one hidden layer can in a reasonable way approximate an arbitrary non-linear function [8]. So in this study, a three-tiered network construction including one input layer, one hidden layer and one output layer was selected. It is well-known that the generalization of ANN model is both dependent on the network topology and the values of network parameters, like the value of learning rate and so on [9]. But there is no unified solution for network parameter determination, so the trial and error method was used in this study. And the performance of different trials was measured by the value of RRMSE (relative root mean square error) showing in expression (2) and the value of MARE (mean absolute relative error) showing in expression (3), where U is the number of samples; T_α is desired value; O_α is predicted value.

$$\mathrm{RRMSE} = \sqrt{\frac{\sum_{\alpha=1}^{\mathrm{U}} (O_\alpha - T_\alpha)^2}{\mathrm{U}}} \Bigg/ \frac{\sum_{\alpha=1}^{S} T_\alpha}{\mathrm{U}} \tag{2}$$

$$\mathrm{MARE} = \frac{\sum_{\alpha=1}^{\mathrm{U}} |(O_\alpha - T_\alpha)/T_\alpha|}{\mathrm{U}} \tag{3}$$

In order to implement our ANN model, the TensorFlow was used. It is a powerful machine learning platform and APIs, in which the computation is represented by dataflow graphs. It supports a variety of applications, especially for neural network modeling and calculations, helping users easily implement training, optimization and inference. In the current study of this paper, our ANN model included three layers, one input layer having four input nodes, one hidden layer (the hidden node number was to be determined) and one output layer having one output node. The issue of our study is mainly about continuous value prediction, so for hidden layer, the sigmoid activation function can be used, and for output layer, the linear activation function can be used. The main implementation codes by TensorFlow 1.1.0 and Python 3.5.3 are shown in the following (the hidden node number was assumed as 3 for illustration).

The main implementation computer program codes based on TensorFlow and Python

```
import tensorflow as tf
import numpy as np
##input layer
iptlyr_nero_num=4
inputs =
tf.placeholder(tf.float32,[None,iptlyr_nero_num])
##hidden layer
hidlyr_nero_num=3
hidlyr_weights = \
      tf.Variable(tf.random_normal([iptlyr_nero_num,\
      hidlyr_nero_num]))
hidlyr_biases =tf.Variable(tf.zeros([1,hidlyr_nero_num])\
      + 0.1)
hidlyr_wx_plus_b = tf.matmul(inputs, hidlyr_weights) + \
      hidlyr_biases
hidlyr_outputs=tf.nn.sigmoid(hidlyr_wx_plus_b)
##output layer
optlyr_nero_num=1
optlyr_weights = \
      tf.Variable(tf.random_normal([hidlyr_nero_num,\
      optlyr_nero_num]))
optlyr_biases =tf.Variable(tf.zeros([1,optlyr_nero_num])\
      + 0.1)
optlyr_wx_plus_c = tf.matmul(hidlyr_outputs,\
      optlyr_weights) + optlyr_biases
predictions=optlyr_wx_plus_c
##improvement of error between prediction and observation
##during training
observations = tf.placeholder(tf.float32, [None, 1])
loss = tf.reduce_mean( tf.reduce_sum(tf.square( \
      observations - predictions),reduction_indices=[1]))
learning_rate=0.1
train = tf.train.GradientDescentOptimizer( \
      learning_rate).minimize(loss)
##validation
valid_rrmse = tf.div( \
      tf.sqrt(tf.reduce_mean(tf.reduce_sum(tf.square(\
      observations-predictions),reduction_indices=[1])))\
      ,tf.reduce_mean(tf.reduce_sum(observations,\
      reduction_indices=[1])))
valid_mare = tf.reduce_mean(tf.reduce_sum(tf.abs(tf.div(\
      predictions - observations,observations)),\
      reduction_indices=[1]))
```

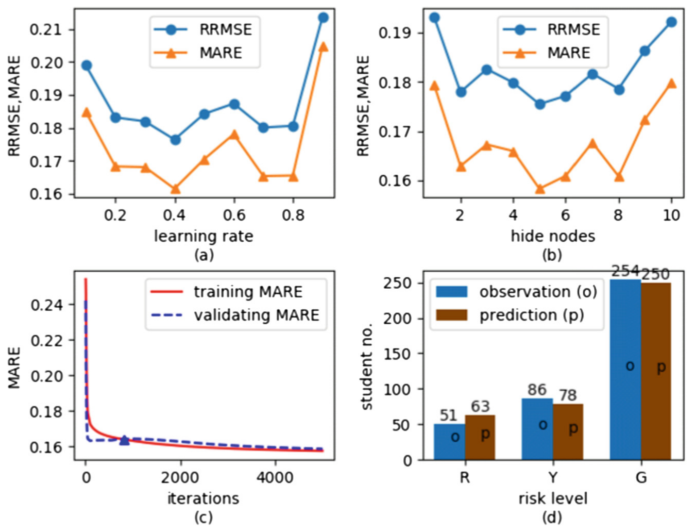


Fig. 1. (a) Learning rate effects, (b) Hidden nodes effects, (c) Iterations effects, (d) The observed and predicted risk results of all students

In order to complete our ANN model, the network parameters at first needed to be tested and optimized with the trial and error method. At the begin, the initial values of learning rate (denoted by η) and training iterations (denoted by n) were chosen by experience as η = 0.01 and n = 2000, and for the initial values of weights and bias in hidden layer and output layer, the values between 0 and 1 were chosen by random. For the value of node number in hidden layer (N_{Hidden}), there is an empirical rule showing in expressions (4) can be used to initialize it, in which N_{Input} is the value of node number in input layer and N_{Output} is that in output layer. In our study, we had $N_{Input} = 4$ and $N_{Output} = 1$, so we used $N_{Hidden} = 3$ as the initial value.

$$N_{Hidden} = (N_{Input} + N_{Output})/2 \tag{4}$$

So first, we kept $N_{Hidden} = 3$ and n = 2000, and for different values of η the tested errors were shown in Fig. 1(a), from which we got optimized η = 0.4 with lowest MARE and RRMSE errors. Next, the values of 1–10 were used to determine hidden node number N_{Hidden}, with η = 0.4 and n = 2000. The tested errors were shown by Fig. 1(b). It was noted that $N_{Hidden} = 5$ was the optimal one. Finally, as the iterations becomes more and more, the MARE of training data will become smaller and smaller, but there will be an over fitting problem, so in conjunction with η = 0.4 and $N_{Hidden} = 5$, we tested MARE on both training and validating data when changing the number of training iterations. Our criterion of stop training is to get the point when MARE on validating data begins to increase while that on training data still decreases. The tested results were shown in Fig. 1(c), from which we could see that the optimal iterations number was 800, after this point there was an over fitting phenomenon. So we set n = 800 at last.

3 Results and Discussions

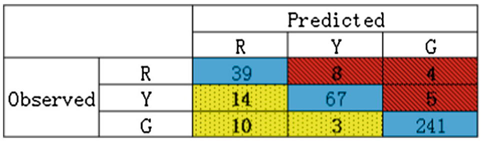
After the process of training, the network parameters were determined, and the final optimal values were η = 0.4, $N_{Hidden} = 5$, and n = 800 for current study. Based on these optimized network parameters, the values of weights and biases in input-hidden layers and hidden-output layers could be determined, which were shown in Table 1. And the corresponding accuracy values of our ANN model on training, validating and total data were shown in Table 2. In which, it was shown that the training data have better accuracy than validating data in general.

Table 1. The values of trained weights and biases of our network model.

The weights between input and hidden layers:				
$W_{11} = -0.315$	$W_{12} = 0.6376$	$W_{13} = -0.0514$	$W_{14} = -0.1973$	$W_{15} = 0.8970$
$W_{21} = -0.2666$	$W_{22} = -0.3215$	$W_{32} = 3.0499$	$W_{24} = -0.7013$	$W_{25} = 0.7192$
$W_{31} = -0.2789$	$W_{32} = -0.6059$	$W_{33} = 0.1020$	$W_{34} = -0.9749$	$W_{35} = -1.1315$
$W_{41} = -0.4460$	$W_{42} = -2.9768$	$W_{43} = 2.4153$	$W_{44} = -1.3014$	$W_{45} = -1.4673$
The biases between input and hidden layers:				
$b_1 = 0.1671$	$b_2 = -0.2705$	$b_3 = -0.3220$	$b_4 = -0.1432$	$b_5 = 0.1837$
The weights between hidden and output layers:				
$W_{11} = -1.2114$	$W_{21} = 1.6841$	$W_{31} = 1.3088$	$W_{41} = 0.6375$	$W_{51} = -0.4901$
The biase between hidden and output layers:				
$c = -0.1908$				

Table 2. The accuracy on training, validating and total data of our network model

	Training	Validating	Total
RRMSE	0.1706	0.1823	0.1735
MARE	0.1575	0.1659	0.1596



		Predicted		
		R	Y	G
Observed	R	39	8	4
	Y	14	67	5
	G	10	3	241

Fig. 2. The risk classification results of all students (Color figure online)

The obtained classification results (prediction) of all students and their actual final grade categories (observation) were shown in Figs. 1(d) and 2. In the observed categories, R (red, serious risk) means actual final grade < 50 points, Y (yellow, moderate risk) means actual final 50 $\leq$ grade < 60 points, and G (green, no risk) means actual final grade $\geq$ 60 points.

From Fig. 1(d), we could see that there were 51 R students, 86 Y students, and 254 G students in the observed, while the model resulted in 63 R students, 78 Y students, and 250 G students. Overall, the model could accurately predict every students into the 'R', 'Y' and 'G' category 88.7% (= (39 + 67 + 241)/391) of the time, which is shown in light blue background in Fig. 2. Furtherly, the model made 'Type II' error (predicting an R student as Y or G student, or predicting a Y student as G student) at a rate of only 4.3%, which mean that only 17(= 8 + 4 + 5) out of 391 students were classified to be performing well or near well, but their actual final course grade put them into R or Y category, which is shown in red diagonal background in Fig. 2. The model also made 'Type I' error at a rate of 6.9%, namely putting 27(= 14 + 10 + 3) students out of 391 in the R or Y category while these students actually had passed the course, which is shown in yellow dotted background in Fig. 2. However, as far as the importance of helping student overcome learning difficulties is concerned, in order to identify at-risk student in early time during their course study process, it is somewhat better to mistakenly predict a student as at-risk student than being unable to identify a student who is really at-risk and needs additional help for his/her study. So it is relatively of less concern about Type I error occurrence. In sum, the model used in this study has a good performance in prediction of student study failure risk.

What's more, in order to further test the performance of our obtained model, the determination coefficient (R^2) and paired t-test were used on our total sample data set. First, after calculation we got the determination coefficient value of R = 0.93, showing excellent agreement between the observed data and predicted results. Besides, in the t-test at significance level of $\alpha = 0.05$, the null hypothesis (H_0: $\rho_1 = \rho_2$) was accepted, where ρ_1 was the mean of observed data and ρ_2 was that of predicted results. It implied that the predicted results had no significance difference from the actual data.

All these results above showed that our obtained network model could well learn the relationship existing between our input and output variables, and it was also reliable in predicting student study failure risk.

4 Conclusions

In constructing neural network model for early warning prediction of student study failure risk, there are some network parameters, like learning rate and training iterations, need to be optimized using a trial and error method [10]. With TensorFlow APIs and Python language, they were easy to be optimized. According to our obtained results predicted by the employed model, college and university educators can implement corresponding pedagogical and learning strategic interventions more timely to help student avoid academic risk. The model is promising in identifying at-risk students who have study difficulties, and makes sense in helping the student who almost failed or failed their courses and may have passed the courses with some earlier learning supports and pedagogical interventions. In sum, all obtained results of this study showed that the neural network model is a reliable and powerful tool to predict student study failure risk. However, further study is also needed, for example more comprehensive input factors should be added, to yield more precision results.

Acknowledgments. The Hunan Province Educational Science 13th Five-Year Planning Program (XJK016QXX003) and the Hunan Provincial Social Science Foundation (17YBQ087) support this study, we are very grateful to them.

References

1. Sandeep, M.J., Erik, W.M., Eitel, J.M.L., et al.: Early alert of academically at-risk students: an open source analytics initiative. J. Learn. Anal. **1**(1), 6–47 (2014)
2. Chen, G., Liu, C., Ou, K., et al.: Discovering decision knowledge from web log portfolio for managing classroom processes by applying decision tree and data cube technology. J. Educ. Comput. Res. **23**(3), 305–332 (2000)
3. Morris, L.V., Wu, S., Finnegan, C.: Predicting retention in online general education courses. Am. J. Distance Educ. **19**(1), 23–36 (2005)
4. Macfadyen, L.P., Dawson, S.: Mining LMS data to develop an early warning system for educators: a proof of concept. Comput. Educ. **54**(2), 588–599 (2010)
5. Jay, B., James, M., Anne, Z., et al.: Using learning analytics to predict at-risk students in online graduate public affairs and administration education. J. Public Aff. Educ. **21**(2), 247–262 (2015)
6. Geraldine, G., Colm, M., Philip, O., et al.: Learning factor models of students at risk of failing in the early stage of tertiary education. J. Learn. Anal. **3**(2), 330–372 (2016)
7. Kevin, C., David, A.: Utilizing student activity patterns to predict performance. Int. J. Educ. Technol. High. Educ. **14**(1), 1–15 (2017)
8. Geng, C., Dandan, L., Haowen, W., Guochang, W.: Analysis and prediction model of financial income in Guangzhou. Stat. Appl. **4**(3), 187–195 (2015)
9. Daniel, B.M., Muttucumaru, S.: Prediction of urban stormwater quality using artificial neural networks. Environ. Model Softw. **24**, 296–302 (2009)
10. Holger, R.M., Graeme, C.D.: The effect of internal parameters and geometry on the performance of back-propagation neural networks: an empirical study. Environ. Model Softw. **13**, 193–209 (1998)