Engineering Science and Technology, an International Journal 23 (2020) 1301–1307

HOSTED BY

Contents lists available at ScienceDirect

Engineering Science and Technology, an International Journal

journal homepage: www.elsevier.com/locate/jestch

Full Length Article

Prediction of middle school students' programming talent using artificial neural networks

Ali Çetinkaya *, Ömer Kaan Baykan

Konya Technical University Engineering Faculty, Computer Engineering Department, Konya, Turkey

ARTICLE INFO

Article history:
Received 12 December 2019
Revised 8 July 2020
Accepted 13 July 2020
Available online 9 August 2020

Keywords:
Prediction of programming skills
Prediction of students' academic Performance
Code.org
ANN
Machine learning

ABSTRACT

Nowadays, the softwarization and virtualization of resources and services rapidly continue, and along with reading and writing, programming is going to be one of the basic human ability. Thus, the detection of skilled programmers at an early age has become important for economies to strengthen their workforce and compete globally. The current technological momentum shows that when the middle school students of today reach the 2030s, the demand for advanced programming skills will be rapidly increased, expanding as high as 90% between 2016 and 2030. Thus, the identification of these skilled people at an early age is important. Accordingly, this study focused on predicting middle school students' programming aptitude using artificial neural network (ANN) algorithms. A participant survey was developed and applied to middle school students consisting of fifth, sixth, and seventh graders from Konya Science Center, Turkey. After the completion of the survey, the participants then took the 20-level Classic Maze course (CMC) on Code.org. The participants' final scores in the CMC were calculated based on the level they completed and the lines of codes they wrote. The best results were obtained using the Bayesian regularization algorithm: Training-R = 9.72284e−1; Test-R = 9.12687e−1, and All-R = 9.597e−1. The results show that ANN is an appropriate machine learning method that can forecast participants' skills, such as analytical thinking, problem-solving, and programming aptitude.

1. Introduction

Concerns regarding the workforce scarcity in science, technology, engineering, and mathematics (STEM) fields have stimulated thriving attention on the relationships between STEM skills and career trajectories [1]. Information technology (IT)-related works are estimated to constitute two-thirds of all new jobs in STEM-related areas [2].

Recent research [3] estimated that the demand for advanced IT and programming skills will rapidly increase and can expand as much as 90% between 2016 and 2030. When the middle school students of today reach the 2030s, their career will begin, and people with programming abilities will undoubtedly be on demand and will be a minority [3]. Therefore, it is becoming important for economies to strengthen their workforce with talented programmers [4].

In a highly automated world, identifying students' programming skills early and directing them into appropriate career paths are increasingly crucial. One of the methods used to detect the programming ability of a student is machine learning prediction techniques. The most commonly used machine-learning methods are supervised learning methods, including decision trees, decision forests, logistic regression, support vector machines, neural networks, kernel machines, and Bayesian classifiers, which are widely employed to predict a numerical value [5]. Machine learning methods used in the prediction of students' academic performance, dropout rate, course registration, grades, GPA, and learning, have recently gained a lot of attraction. Some of the related works in this field for the last 10 years are presented as follows:

In [6], a forecasting model employing a neural network approach was presented to identify the potential elements in determining the online courses picked by pupils. The study showed that the suggested model performs better than three well-known machine learning methods and two previous traditional models. In [7], the advantage of using recurrent neural networks to model pupil learning was investigated. The employed ANNs outperformed the latest prediction models on real pupil data, which proposes a new line of research for knowledge tracing. In [8], to predict pupils' academic performance, ANNs that utilize 10-fold cross validation were employed on the grade data of 300 pupils. The suggested model offered better performance than basic train-

* Corresponding author.
E-mail addresses: ali.cetinkaya@gmail.com (A. Çetinkaya), okbaykan@ktun.edu.tr (Ö.K Baykan).
Peer review under responsibility of Karabuk University.

https://doi.org/10.1016/j.jestch.2020.07.005
2215-0986/

ing techniques and association rule mining. A new method to predict students' academic performance based on a supervised neural network was developed based on the data of 120 students. The suggested model outperformed traditional methods and is expected to help educational institutions identify appropriate students for academic research. The results also showed that students' academic performance was affected by family and social media [9]. In [10], the academic performance of future mathematics teachers was forecasted by employing logistic regression analysis (LRA) and ANN. The prediction success rate of the ANN models outperformed that of LRA models. In [11], the ratio of pupils' registration in the future semesters was forecasted. They used different prediction models, such as ANNs with bagging and boosting models, decision trees, and Bayesian simple logistic regression. In conclusion, the ANN bagging method can be distinguished as a better technique with 96% success. In [12], an ANN-based prediction model was employed to estimate student success during the academic year. The results showed that the observed and predicted values were significantly similar. In [13], an ANN model was presented to predict the academic performance of Turkish and Malaysian vocational school students. The results were significantly successful: 98% sensitivity rate over 922 samples for the Turkish students and 95.7% sensitivity rate over 1050 samples for Malaysian students.

Considering the future programming workforce, more people with programming skills are needed. It is important to identify these talented people at an early age. Therefore, this study focused on predicting middle school students' programming aptitude using ANN algorithms and visual web-based programming tools. Most of the works in the literature focused on the prediction of students' academic performance in higher education, and very few studies focused on students' programming talent. The following research questions (RQs) were stated by taking into account the research gap on this topic:

RQ1: How are ANN algorithms going to predict middle school students' programming aptitude using the data gathered from students' demographic information and answers to analytical thinking, map-sketching, and paper-folding problem questions.

RQ2: How can middle school pupils' parents and teachers be supported to encourage their pupils to pursue computer science-related fields in career trajectories?

The rest of the paper is organized as follows: Section 2 describes the materials and methods, Section 3 presents the results and discussion, and Section 4 discusses the conclusions of the study.

2. Materials and methods

The study developed a participant survey and applied it to middle school students consisting of fifth, sixth, and seventh graders from Konya Science Center, Turkey.

After the completion of the survey, the participants then took the 20-level Classic Maze course (CMC) on Code.org. The participants' final scores in the CMC were calculated based on the level they completed and the lines of codes they wrote. The ANN algorithms were executed on MATLAB using the questions of the survey as the inputs and the calculated participants' final scores in the CMC as the output.

This study was conducted using Konya Science Centers' physical spaces and resources. All precautions were performed to ensure that all the participants do not experience any test anxiety.

2.1. Code.org

Code.org's CMC is a game-like, Blockly-based, drag/drop-style, and self-instructed programming tool. CMC consists of a set of instructions to acquire well-known game characters, such as Angry Bird, through a maze to hit a Green Pig [14].

Code.org was launched in 2013 by twin brothers Hadi and Ali Partovi with a video promoting computer science [15]. The critical goal of Code.org is to help pupils realize that computer science is easier to attain and relevant to everyone [16,17]. Code.org is one of the coding literacy campaigns launched in 2013 that fosters the development of critical reasoning and communication abilities in a highly digital world [15]. Teachers can also provide several learning options. Some students may choose to use five level-ups in-game programs that are codified like Code.org [18]. The current learning setting is designed not only for learning the essential elements of computer work but also for school pupils of all ages interested in computer programming [19].

There are many types of courses on Code.org, and CMC is one of the most popular among them. In CMC, key programming concepts, such as comparisons, loop structures, conditions, are embedded playfully. Each level of CMC focuses on the different programming concepts with a particular difficulty level. A participant can complete the CMC without needing an instructor. The administrator of the Code.org class can access each participant's written lines of codes and completion status data at every level. The Code.org CMC scores were calculated by examining each participant's level data and written lines of codes at each level in the CMC.

2.2. Participants

This study was conducted in the academic year of 2018–2019, with three groups of 200 fifth, sixth, and seventh graders randomly selected from a middle school in the city of Konya in Turkey. The analysis of the participants' demographic information and other details are presented in Table 1.

Among the participants, 109 (54.5%) were female, and 91 (45.5%) were male. The age distribution of the students is as follows: 32 (16%) 10-year-olds, 92 (46%) 11-year-olds, 63 (31.5%) 12-year-olds, 12 (6%) 13-year-olds, and 1 (0.5%) 14-year-old. The educational background of the students' parents is as follows: primary school (43/21.5%, mother; 25/12.5%, father); middle school (97/48.5%, mother; 77/38.5%, father); secondary school (35/17.5%, mother; 45/22.5%, father); and university level (25/12.5%, mother; 53/26.5%, father).

2.3. Survey

The survey consists of four sections: demographic information, paper-folding problems, map sketching, and analytical thinking questions.

Table 1
Demographic Information of the Participants.

Variable	Group	%
Gender	Female	45.5
	Male	54.5
Age	10	16
	11	46
	12	31.5
	13	6
	14	0.5
Mother's educational background	Primary school	21.5
	Middle school	48.5
	High school	17.5
	University	12.5
Father's educational background	Primary school	12.5
	Middle school	38.5
	High school	22.5
	University	26.5

The demographic information questions consist of gender, age, parent's educational levels, parent's job, mathematics grade, homework time, possession of an electronic device and time spent with the device, coding knowledge, and computer screen time.

The paper-folding problems were derived from [20]. The paper-folding test was employed to evaluate visualization and spatial reasoning, based on the skills to handle and transform spatial patterns and to recognize whether one image is a transformation of another. In the survey, three paper-folding problems with different levels of difficulties were given. The participants were asked to decipher how the holes on the folded paper surface will be when the paper is unfolded.

The map sketching was adapted from [20]. The map was designed to include various landmark locations, such as airports, parking areas, school, lake, castle, hospital, library, and science center. The participants were asked to choose two places that they would like to visit. Then, the participants selected two numbers from 1 to 7, which indicates different paths in the map, for the starting and endpoints. After the path number selection, the participants must follow the path from the starting point to the first landmark, then go to the second landmark, and arrive at the endpoint.

Analytical thinking questions were cited by experts from the general aptitude tests prepared by the Directorate of Measurement, Evalution and Examination Services in the Ministry of National Education.

As the literature indicates, mathematics grades are an influential factor in predicting student programming skills [21–23]. Therefore, mathematics grades were taken into account in the prediction model.

2.4. Data analysis

This study investigated the relationship between the participants survey results and the participants' success in CMC by employing the ANN algorithms.

The most common ANN training algorithms, such as scaled conjugate gradient [24], Levenberg–Marquardt [25], and Bayesian regularization [26] were employed. The algorithms were incorporated by varying the number of neurons and hidden layers to obtain the best predictive models of the students' programming talent in MATLAB [27].

Below, the ANN models, which are generated by the input and target parameters, and the measurement of performance among the models, are described in detail. The evaluations were made over 100 points for all input and target parameters.

Input parameters: 13 input parameters for each ANN model was used. The input parameter distributions are as follows: demographic information (five), mathematics grade (one), paper-folding problems (three), map sketching (one), and analytical thinking questions (three).

Target parameters: CMC consists of 20-level puzzles, where each puzzle has a difficulty level and targets to get a notion of a particular type of programming concept, such as comparisons, loop structures, and conditions, to a participant.

Through Code.org teacher's admin panel, which can be seen in Fig. 1, the students' completed level and lines of code data on CMC can be identified.

The final Code.org CMC score for each participant was calculated as follows. Each level was weighted according to its difficulty. If a user completes each level within its expected lines of code, it will get 100 points as a final score. If there is a missing level or more lines of code than expected, then some penalty is applied to the final score. The calculated final score was used as the target parameter for the ANN model.

Training, validation, and test subsets were produced by randomly splitting data in the ANN model. During the ANN learning process, training data were utilized, and validation data gauged the network generalization. The network was modified with respect to its errors and stopped training when generalization stopped improving. The test process did not make any changes on the network structure; rather, it provided an independent evaluation of the performance of the network before and after training [27].

To determine coherent data splits of training and validation, several combinations were considered. In this study, the dataset of 200 was split into 140 (70%) for the training, 30 (15%) for the validation, and 30 (15%) for the testing of the models.

The following steps were followed to measure the performance of the prediction models: First, each of the training algorithms was combined with one hidden layer with various numbers of neurons. Next, the *MSE* and R^2 values were utilized to evaluate the models.

The correlation between the predicted and observed values of Y is R. The R^2 value measures the regression and shows the proportionate quantity of variation in the reaction variable Y explained by the independent variables X. R^2, which has a value between 0 and 1, evaluates the regression and shows the proportionate quantity of variation in the reaction variable Y described by the independent variables X. The linear correlation increases between inputs and target when R^2 is close to 1. The mean quadratic difference between outputs and targets is the *MSE*value. The lower the *MSE* value is, the better. The *MSE*equation is depicted in (1):

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 \tag{1}$$

where N is the number of the cell array outputs, y_i is the matrix or cell array of targets, and $\hat{y}_i$ is the matrix or cell array of outputs. The R^2 equation is depicted in (2):

$$R^2 = 1 - \frac{MSE(\text{model})}{MSE(baseline)} \tag{2}$$

The *MSE* of the model is computed as above, and the *MSE*of the baseline is defined in (3):

$$MSE(baseline) = \frac{1}{N}\sum_{i=1}^{N}\left(y_i - \bar{y}\right)^2 \tag{3}$$

where y with a bar is the mean of the observed y_i.

When a feedforward network is initialized, the network's goal is to identify model parameters that minimize the error between the predicted and the real value by updating the weights. Thus, if the network is not adequately accurate, then the network could be restarted, and the new outcomes are recorded for future comparisons.

The number of neurons in the hidden layer of the network can be increased to give more flexibility to the system to optimize the parameters. Moreover, it is essential to gradually increase the hidden layer size to allow the network to optimize more parameters than data vectors to restrict these parameters. If the hidden layer is too large, then this can lead to under-characterization.

Nevertheless, picking a different algorithm, such as Bayesian regularization, can sometimes cause the generalization capacity to outperform the stopping early method, which is the default model for improving generalization [27]. In this study, a different number of neurons were employed to optimize the neuron number in the ANN model hidden layer considering the above-mentioned MSE andR^2 values.

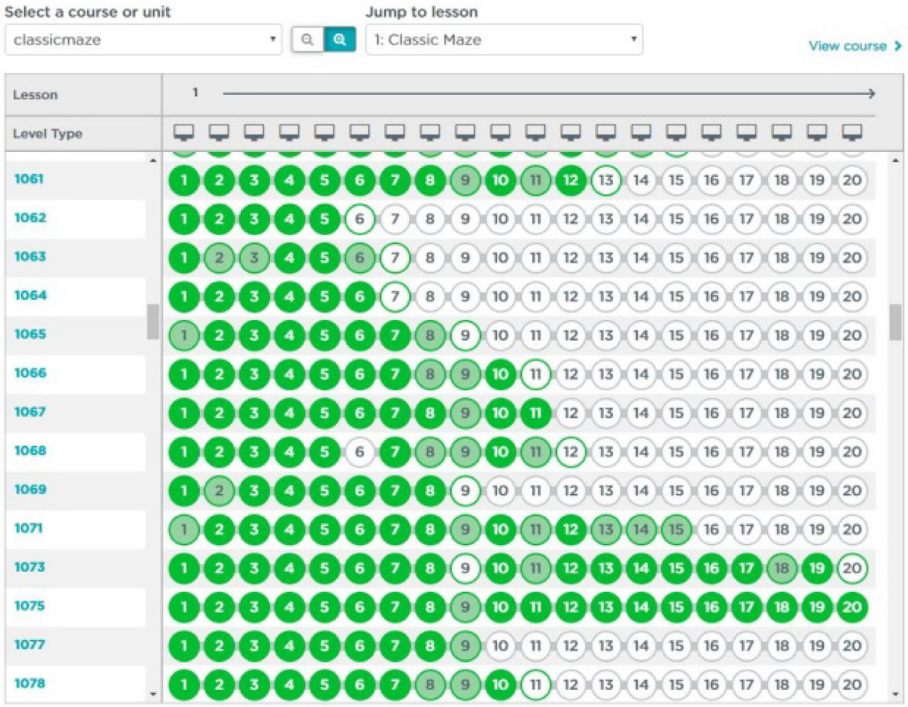

Fig. 1. Snapshot of Code.org's Classic Maze student completion level.

3. Results and discussion

ANNs are widely utilized for their ability to recognize patterns and training [23]. The Levenberg–Marquardt, Bayesian regularization, and scaled conjugate gradient algorithms are the most commonly used prediction algorithms of ANNs. The neuron number in a hidden layer is one of the important parameters of an ANN model to optimize the outputs.

The employed ANN models consisted of one hidden layer, 13 inputs, and one target (Fig. 5). Each model was executed 10 times for between 2 and 10 neurons to find the optimal number of neurons in its hidden layer. The best execution results were given in Table 2. Moreover, the best results' training, testing, and validation values are given in Figs. 2–4 for Levenberg–Marquardt, Bayesian regularization, and scaled conjugate gradient algorithms, respectively.

The error histogram graphics are given in Figs. 6–8 for the three algorithms.

The performance graphics is presented in Figs. 9–11 for the Levenberg–Marquardt, Bayesian regularization, and scaled conjugate gradient algorithms, respectively.

Among the ANN training algorithms, Bayesian regularization algorithm has been found to give the best effect with 10 neurons in the hidden layer. The results of the Bayesian regularization algorithm are as follows: Training-R = 9.72284e−1; Test-R = 9.12687 e−1, and All-R = 9.597e−1; Training-MSE = 11.29453e−0; Test-MSE = 50.71159e−0; and All-R = 9.597e−1.

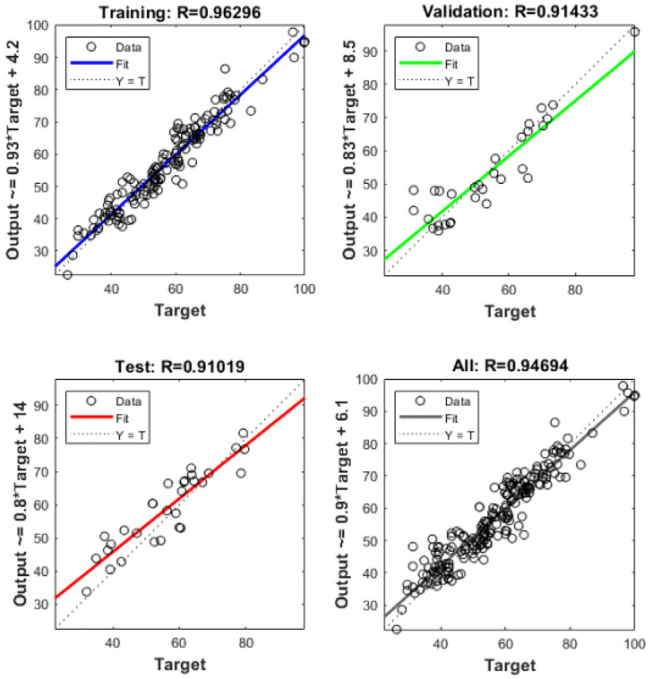

Fig. 2. Best results of the Levenberg–Marquardt algorithm's training, testing, and validation values.

Table 2
Analysis of the best results.

Algorithm	NoNHL	Training-R	Validaiton-R	Test-R	All-R	Training-MSE	Validation-MSE	Testing-MSE
Levenberg-Marquardt	5	9.65123e−1	9.24989−1	9.16369e−1	9.5197e−1	15.43640e−0	28.93511e−0	34.76528e−0
Bayesian Regularization	10	9.72284e−1	–	9.12687e−1	9.597e−1	11.29453e−0	–	50.71159e−0
Scaled Conjugate Gradient	5	9.5098e−1	9.201e−1	9.0196e−1	9.3626e−1	19.19747e−0	43.28254e−0	45.81064e-0

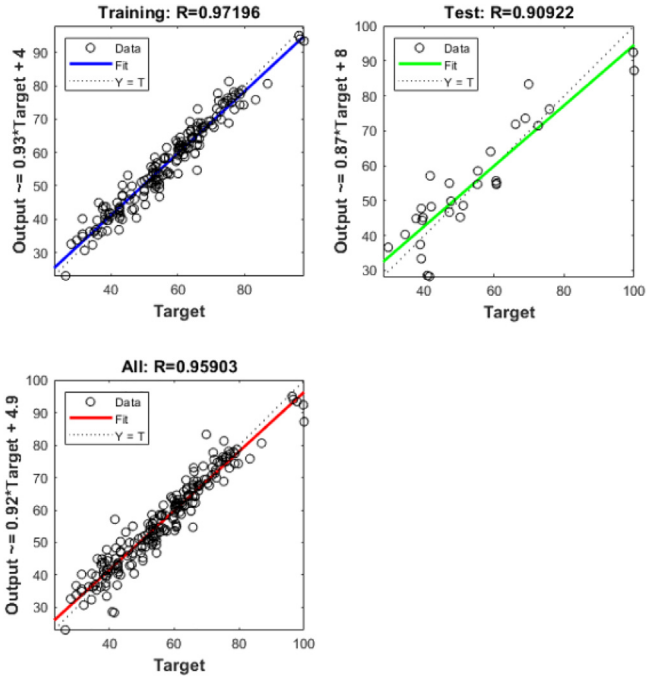

Fig. 3. Best results of the Bayesian regularization algorithm's training, testing, and validation values.

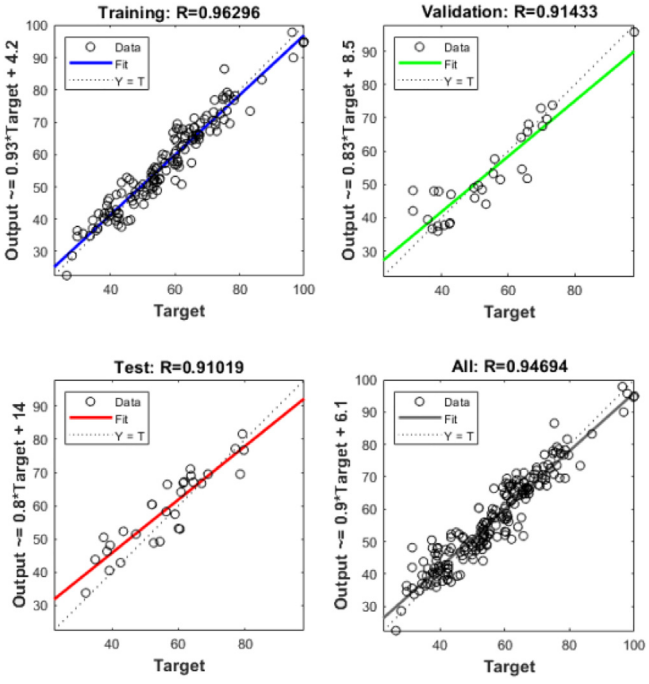

Fig. 4. Best results of the scaled conjugate gradient algorithm's training, testing, and validation values.

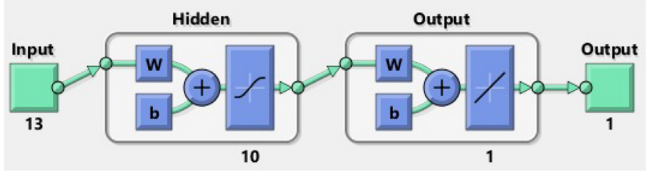

Fig. 5. Neural network diagram.

Fig. 6. Best results of the Levenberg–Marquardt algorithm's error histogram values.

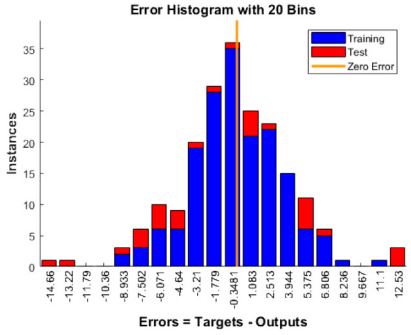

Fig. 7. Best results of the Bayesian regularization algorithm's error histogram values.

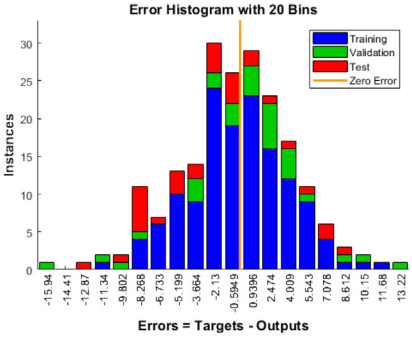

Fig. 8. Best results of the scaled conjugate gradient algorithm's error histogram values.

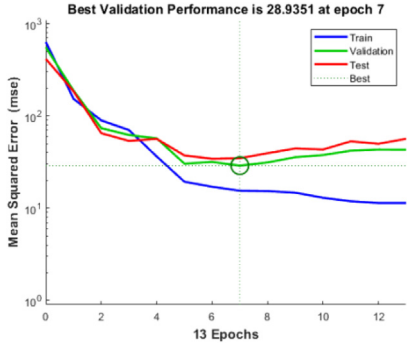

Fig. 9. Best results of the Levenberg–Marquardt algorithm's performance values.

Fig. 10. Best results of the Bayesian regularization algorithm's performance values.

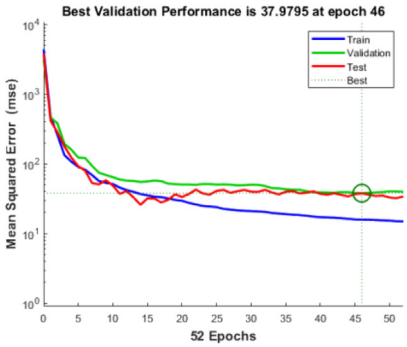

Fig. 11. Best results of scaled conjugate gradient algorithm's performance values.

The error histogram, in which the blue bars symbolize training data, the green bars stand for validation data, and the red bars represent testing data, can provide us a display of outliers, which are data points where the fit is remarkably worse than the bulk of data [27].

In the case of Levenberg–Marquardt algorithm's error histogram values, while most errors fall between − 9 and 9, several outliers can be noticed, such as − 13.73 and 15.84. As for the Bayesian regularization histogram, the bulk of errors is between −9 and 8, but a couple of outliers can be seen, such as −14.66, −13.22, and 12.53. The scaled conjugate gradient's error histogram shows the majority error values between −9 and 9, which also indicate several outlier data points. These outliers are also visible on the testing regression plot.

The employed ANN models outperformed the well-known regressions methods from the literature: rational quadratic gaussian process regression R^2: 0,86; linear regression R^2: 0,86; linear support vector machine R^2:0,84; Levenberg–Marquardt training R^2: 9.65123e-1; Bayesian regularization training R^2: 9.72284e-1; Scaled conjugate gradient training R^2: 9.5098e-1.

4. Conclusions

When the middle school students of today reach to the 2030s, after university graduation, their career will begin. The current technological momentum shows that the softwarization process will rapidly continue, and almost all of the new solutions will be code-based. Programming will be one of the necessities for a citizen to live in the future.

The prediction models of middle school students' programming talent were investigated for future career trajectories. In light of the findings of this investigation, it was concluded that the programming tendency of middle school students was predictable by the parameters used in the participant questionnaire. With the conclusion, it is observed that ANN was an appropriate machine learning method to forecast participants' skills, such as analytical thinking, problem solving, and programming aptitude.

The ANN algorithms, which were employed in this study, offer a method to allow the parents and teachers of middle school pupils to determine their pupil's programming talent and to encourage them to pursue computer science-related fields for future career trajectories. Pupils who have been told by parents or teachers that they will be good at programming are 2.5 to 3 times more likely to be interested in getting into programming in the future [28]. Thus, the study results will be shared with the participants' teachers and parents to encourage the students to pursue programming career trajectories. Even without prior programming experience, if a student obtained good results in the survey, then his parents and teacher can be advised to encourage him to pursue a programming-related career. This model should, as in other prediction models, not mean that students who are relatively unsuccessful cannot work in these jobs.

Consequently, the study indicates that ANN algorithms are an important solution for validly and efficiently predicting programming aptitude.

One limitation of this study here is the relatively small sample size of 200 students. In recent studies on ANN, small data sets have also been shown to give good results in ANN [29–31]. Beyond the sample size, this study's focus on urban state school students in one city, Konya, Turkey, presents a specific limitation on how the study findings can be extensively generalized. It is also acknowledged that the same results are unlikely without the dataset, and the data are serviceable from the authors on request.

In future works, different studies can be conducted more on prediction algorithms, such as deep learning (e.g. LSTM), machine learning algorithms and fuzzy logic methods. The questions of the participant survey can be further diversified and customized according to various cultural and demographic backgrounds. Future work should attempt to achieve a more extensive and diverse sample of students and should focus on earlier age than middle schools, such as primary school students.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

This paper is based on Ali Çetinkaya's Ph.D. dissertation. The authors would like to acknowledge the support of the Ministry of National Education Konya Provincial Directorate of Turkey.

References

[1] I. Ognjanovic, D. Gasevic, S. Dawson, Using institutional data to predict student course selections in higher education, Internet High Educ. 29 (2016) 49–62, https://doi.org/10.1016/j.iheduc.2015.-12.002.
[2] J. Zagami, M. Boden, T. Keane, B. Moreton, K. Schulz, Girls and computing: female participation in computing in schools, Aust. Educ. Comput. 30 (2015) 2.
[3] J. Bughin, E. Hazan, S. Lund, P. Dahlström, A. Wiesinger, A. Subramaniam, Skill shift: Automation and the future of the workforce, McKinsey Global Institute 1 (2018) 3–84.
[4] J.H. Maloney, K. Peppler, Y. Kafai, M. Resnick, N. Rusk, Programming by choice: urban youth learning programming with Scratch, Educ. Inf. Technol. 39th SIGCSE (2008) 367–371.
[5] D.J. Ruppert, The elements of statistical learning: data mining, inference, and prediction, Am. Stat. Assoc. 99 (466) (2004) 567, https://doi.org/10.1198/jasa.2004.s339.
[6] A.A. Kardan, H. Sadeghi, S. Ghidary, M.R.F. Sani, Prediction of student course selection in online higher education institutes using neural network, Comput. Educ. 65 (2013) 1–11.
[7] C. Piech, J. Bassen, J. Huang, S. Ganguli, M. Sahami, L.J. Guibas, J. Sohl-Dickstein, Deep knowledge tracing, J. Adv. Neural Inf. Process. Syst. (2015) 505–513.

[8] S.J. Sebastian, J. Puthiyidam, Evaluating students performance by artificial neural network using WEKA, International Journal of Computers and Applications 119 (23) (2015) 36–39, https://doi.org/10.5120/21380-4370.
[9] M.F. Sikder, M.J. Uddin, S. Halder, Predicting students yearly performance using neural network: a case study of BSMRSTU, 5th Int Conf. Informatics (2016) 524–529.
[10] E. Bahadir, Using neural network and logistic regression analysis to predict prospective mathematics teachers' academic success upon entering graduate education, Educ. Sci.: Theory Pract. 16 (3) (2016) 943–964.
[11] B. Nakhkob, M. Khademi, Predicted increase enrollment in higher education using neural networks and data mining techniques, J. Adv. Comput. Res. Q. 7 (2016) 2345–2356.
[12] I. Aydogan, Estimation of student successes by artificial neural networks and comparison of efficacy of impact models by logistic regression analysis, Ph.D. Dissertation, Yüzüncü Yil University, 2017.
[13] A. Yagci, M. Cevik, Predictions of academic achievements of vocational and technical high school students with artificial neural networks in science courses (physics, chemistry and biology) in Turkey and measures to be taken for their failures, Educ. Inf. Technol. 37 (2019).
[14] Code.org: Available on: studio.code.org/hoc/1 (2019).
[15] H. Partovi, M. Sahami, The hour of code is coming!, ACM SIGCSE Bull. 45 (2013) 4–5.
[16] C. Wilson, What's next for Code.org?, IEEE Computer. Soc. 8 (2013) 95–97.
[17] F. Kalelioglu, A new way of teaching programming skills to K-12 students: Code. org. Computers in Human, Behavior, Comput. Human Behav. 52 (2015) 200–210.
[18] M. Israel, Q.M. Wherfel, J. Pearson, S. Shehab, T. Tapia, Empowering K-12 students with disabilities to learn computational thinking and computer programming, Teach Except Child. 48 (2015) 45–53.
[19] H. Buckova, J. Dostal, Modern approach to computing teaching based on Code. org, J. ICER Proc. 1 (2017) 5091–5096.
[20] S. Fincher, B. Baker, I. Box, Q. Cutts, M. de Raadt, P. Haden, J. Hamer, M. Hamilton, R. Lister, M. Petre, R. Simon, K. Sutton, D. Tolhurst, J. Tutty, Programmed to succeed?: A multi-national, multi-institutional study of introductory programming courses, Technical Report, University of Kent, 2005.
[21] P. Byrne, G. Lyons, The effect of developments in student attributes on success in programming of management students, Int. Conf. Educ. Technol. Comput. ICETC 6 (2009) 191–193.
[22] G. White, M.A. Sivitanides, Theory of the relationships between requirements of computer programming languages and programmers' cognitive characteristics, J. Inf. Syst. Educ. 14 (2003) 409–416.
[23] J. Bennedsen, M.E. Caspersen, Revealing the programming process, ICER Proc. 16 (2015) 155–163, https://doi.org/10.1145/1047344.1047413.
[24] M.F. Moller, A scaled conjugate gradient algorithm for fast supervised learning, Neural Networks 6 (1993) 525–533, https://doi.org/10.1016/S0-893-6080(05)80056-5.
[25] M.T. Hagan, M.B. Menhaj, Training feedforward networks with the Marquardt algorithm, IEEE Trans. Neural Networks 5 (1994) 989–993.
[26] F.D. Foresee, M.T. Hagan, Gauss-Newton approximation to Bayesian learning, Proc. Int. Jt. Conf. Neural Networks 3 (1997) 1930–1935, https://doi.org/10.1109/icnn.1997.614194.
[27] Mathworks: 2019. Available on:www.mathworks.com/products/matlab.
[28] Google Inc.; Gallup Inc. (2017) 1–11.
[29] C. Greco, A. Polonioli, J. Tagliabue, Less (data) is more: why small data holds the key to the future of artificial intelligence, Int. Conf. on Data Sci. Tech. and App. DATA (2019).
[30] W. Hämäläinen, M. Vinni, Comparison of machine learning methods for intelligent tutoring systems, Lecture Notes Comput. Sci. (2006) 525–534, https://doi.org/10.1007/11774303_52.
[31] G. Leroy, T.C. Rindflesch, Effects of information and machine learning algorithms on word sense disambiguation with small datasets, Int. J. Med. Inform. 74 (2005) 573–585.