

CHAPTER 1

INTRODUCTION

Background of the Study

Living conditions in societies have been continuously evolving which, for the most part, have been greatly influenced by advancements in technology. However, the education sector is lagging behind when it comes to its development. According to a report published by The World Bank in 2018, there is a worldwide learning crisis. They further mentioned that millions of young adults still do not know even the most basic life skills that should have been learned in primary school.

With the COVID-19 pandemic, governing bodies around the world struggled to balance their duty to enforce education while assuring public health and safety (Bilefsky & Gutierrez, 2021). Opportunities in education have lowered and became ‘dramatically unequal’ because of the closure of schools. Only a handful of students were able to cope with the changes in school setting. Putting things into perspective, the UNICEF, UNESCO, and the World Bank conducted a survey which shows that less than half of the student population is adopting the remote learning setup (Saavedra, 2021).

Philippines is one of the developing countries that have been experiencing education crisis even before the pandemic. On top of that, 57% of Philippine households do not have internet access. This leads to them being more vulnerable to the impacts of school closure brought about by the COVID19 pandemic. With remote learning being the only option, these households are forced to shell out more money to accommodate the need for education. Furthermore, the transition to remote learning has negatively impacted students both on their academics and mental health. This consequently increases the risk of school dropouts, child labor, and whatnot (Bilefsky & Gutierrez, 2021).

One way to improve the education system is to make use of relevant raw data to determine areas of development. With data, governments are able to gain insights to identify what needs to be done and make timely and evidence-based decisions (Bilefsky & Gutierrez, 2021). Business Intelligence is a field that provides a technical framework for the collection, storage, and analysis of data (Frankenfield, 2021). It offers automation tools (such as for Data Mining) which can be used to extract knowledge from raw data input (Cortez & Silva, 2008).

Several studies have been conducted where data was used to predict student performance given several demographic factors as input. Minaeli-Bigdeli et al. (2003) utilized a web-based system to gather

student data. The data collected is subjected to a series of pattern classifiers to identify the classification of each student. The accuracy of the predictions/classifications are further enhanced using a genetic algorithm (GA) to identify more appropriate weighting of the features collected. A 2014 study conducted by Chen et al. concluded that the cuckoo search algorithm was able to train a neural network that can be used in predicting students' academic performance. Some of the inputs for the trained neural network were previous exam results, high school location, and gender.

More recently, a 2019 study conducted by Kaya (2019) made use of an artificial neural network (ANN) to analyze the relationship between the courses offered in a program. Further, student performance for a particular course was also investigated to determine if it can be affected by a student's performance in previously taken courses. The relationship between student performance and courses taken can be extracted from the data and was useful for predicting student performance on a particular course. Hence, the ANN was able to successfully predict student performance.

In this research, data from two private schools in Portugal will be gathered and subjected to a simple Artificial Neural Network model with three layers. The model will be used to predict student performance in Mathematics and Portuguese given the past school grades, demographic, social, and other school related data.

Statement of the Objectives

The study aims to develop a three-layered artificial neural network (ANN) that can classify students based on the input data and to determine the accuracy of the neural network. Specifically, the objectives of this research are:

1. To develop an artificial neural network to predict whether a student will pass or fail the subjects Mathematics or Portuguese
2. To calculate the accuracy of the neural network in predicting student performance

Significance of the Study

The purpose of this study is to build a three-layered artificial neural network (ANN) that is able to extract knowledge in data collected from schools. This will greatly benefit the society especially the education sector. The obtained information from the models can be used to improve the quality of education and enhance resource management (Cortez & Silva, 2008). Governments and educational institutions can implement better policies and standards of learning. This can result to an even more developed educational system.

Professors and instructors will also be able to determine which of their students need additional training or support in their courses. They will be able to make the necessary actions or adjustments so that every student is able to learn. Consequently, the learning experience for the students will become more enjoyable and fruitful.

Lastly, the researchers in this field will be able to use this research as a reference in constructing innovative models for classification and knowledge extraction from raw data. From there, the models can be used to solve various problems even outside the field of education and student performance.

Scopes and Limitations

The study will make use of several attributes that can be derived from the students' personal information (age, sex, health status, educational background, etc.), their school performance (grades in previous grading periods, number of failures, number of absences, etc.), their parents and family (educational background, jobs, etc.), relationships with friends (going out with friends, romantic relationships, etc.), and some miscellaneous information such as whether they took additional paid classes, had tutors, participated in extra-curricular activities, internet access, and etc.

This information will serve as inputs for the artificial neural network, and it should be able to classify the students whether they will pass or fail the proceeding grading period. The inputs that had a large influence of the output will be identified. Additionally, the classification accuracy of the model will also be calculated.

CHAPTER 2

REVIEW OF RELATED LITERATURE

The field of education provides an array of possible applications for data mining (Ma, Liu, & Wong, 2000). Some of these applications include student performance prediction where useful information is extracted from the automated analysis raw data (Cortez & Silva, 2008). These data can be used for determining students that needs additional support (Ma, Liu, & Wong, 2000), identifying factors that affect student performance (Cortez & Silva, 2008), as well as a decision-making tool for curriculum development (Kaya, 2019). The following related literatures are some of the studies that have been carried out in this particular field.

Several data mining methods were used in predicting student performance in a study conducted by Minaei et al. (2003). They utilized the data being recorded by the web-based educational system called LON-CAPA. The features used for the model consisted of student homework data such as the total number of correct answers in the homework and total number of tries in solving a problem. The models had three classification tasks, all of which are based on students' grades.: binary (pass/fail), 9-class (grade-specific), and 3-class (high, middle, low). Results show that the combination of multiple classifiers had the best overall performance for all three classification tasks. Further, the use of a genetic algorithm (GA) as an optimization tool for the classifiers also showed significant improvement in the performance of the different models. In effect, the information extracted from the database was able to provide insights for professors to identify which of their students are at risk of failing earlier.

This study conducted by Minaei et al. (2003) made use of features that were primarily based on the students' performance in answering homework provided that the homework is done in the web-based education system. While these data are effective in predicting student performance, student and family background also plays an important factor in student performance. The following study made use of a dataset with features that include both student academic performance as well as an extensive personal and family background.

In the study conducted by Cortez and Silva (2008) entitled 'Using Data Mining to Predict Secondary School Student Performance', several data mining (DM) models were used: Naïve Predictor (NV), Neural Network (NN), Support Vector Machines (SVM), Decision Trees (DT), and Random Forest (RF). All of these made use of the same dataset for the construction of their respective models. The performance of each model was evaluated with the Naïve Predictor as the basis of comparison. Results show that the nonlinear

function methods (NN and SVM) were outperformed by the tree-based methods. This was largely attributed to the high number of irrelevant inputs to which NN and SVM are sensitive to. Nevertheless, the neural network was still able to have a percentage of correct classification (PCC) of 88.3% for binary classification (pass or fail) and a 60.3% PCC for the five-level classification. Lastly, a 2.05 Root Mean Squared result for regression.

Having that said, the neural network that will be developed in this study will have the default configuration where all inputs/features are considered for initial testing. Further, the neural network will only be performing a binary classification (pass or fail) and the same dataset from Cortez and Silva's study will be used.

A three-layered feed-forward artificial neural network was developed by Chen and Do (2014) where they made use of two powerful search algorithms: Cuckoo Search and Gravitational Search Algorithm. These two algorithms were used to train the neural network that would predict student academic performance. The input variables were taken from the admission registration profiles which includes academic performance, personal background, and entrance exam results. The output variable is the average score of a student in the first academic year in college which ranges from 0-10. The neural network consists of only one hidden layer where it was simulated to have optimal performance with six hidden nodes. Results show that both search algorithms have satisfactory performance when it comes to predicting student performance. However, the cuckoo search algorithm had a more accurate and reliable prediction outcome. The same three-layered neural network structure will be adopted in this study.

CHAPTER 3

METHODOLOGY

Machine Specifications

The machine used in developing the model and application is an Acer Aspire 3 A315-A41G-R4BW with Windows 10 Home Single as the operating system. The system consists of an AMD Ryzen 5 2500U Processor (with Radeon Vega Mobile Gfx clocking at 2.00 GHz and eight logical processors) along with an AMD Radeon Vega 8 Graphical Processing Unit, a 12-gigabyte (GB) Random Access Memory (RAM) with only 10.9 GB that is usable, and an L1 cache with a size of 384 kilobytes (KB).

Artificial Neural Network

For this study, two neural networks were developed: 1) A three-layered 32-22-1 ANN built from scratch using Python and NumPy, a Python module that offers mathematical functions and random number generators that will be used for the project; and 2) A neural network built using Python with TensorFlow and Keras. Both of these models utilized the same dataset which will be talked about more in detail in the following section.

Scratch Network

The 32-22-1 ANN or the ‘Scratch Network’ followed the rule of thumb for the hidden nodes where it must be at around half of the number of input data features (Ranjan, 2019). With that said, shown in Equation (1) is the formula used to derive the number of hidden nodes for this network.

$$numHiddenNodes = \frac{2}{3}x + 1 ; x = numInputNodes \quad (1)$$

The program was developed with a functional paradigm. The flow of the program (see Appendix A for source code GitHub link) follows the pseudocode summarized in Figure 1. As the dataset is loaded into the program, the values of each feature data are normalized for the neural network to understand. In this case, the values will range from 0.1 to 0.99 using the function shown in Figure 2. The dataset will be then divided into three subsamples: Training, Validation, and Testing with a distribution of 40%, 30%, and 30%, respectively.

```

math_dataSet, math_headers = loadDataFiles(math)
portugal_dataSet, portugal_headers = loadDataFiles(portugal)

shuffleData()
partitionData() #40-30-30

initializeHiddenNodeWeights() # 32 weights each (corresponding to each inputnode), 22 bias weights for 22 hiddenNodes
initializeOutputNodeWeights() # 22 weights: 50% = 1, 50% = -1, biasWeight = 0.1
LearningRate = 0.4
epoch = 0
percentageCorrectClassification = 0

while (percentageCorrectClassification < 50): # train until percentageOfCorrectClassification is at least 50%

    # ! TRAINING
    trainNeuralNetwork(mathTraining, hiddenNodeWeights, outputNodeWeights, targetOutputs)
    # trainNeuralNetwork(portugalTraining, hiddenNodeWeights, outputNodeWeights, targetOutputs)

    # ! CROSS VALIDATION
    percentageCorrectClassification = validateNeuralNetwork(mathTraining, hiddenNodeWeights, outputNodeWeights, targetOutputs)
    percentageCorrectClassification = validateNeuralNetwork(portugalTraining, hiddenNodeWeights, outputNodeWeights, targetOutputs)

```

Figure 1. Pseudocode for Functional Implementation of 33-22-1 ANN

```

def normalizeValue(value, Vmax, Vmin): # normalize the value to 0.1-0.9
    Tmax = 0.9 # max value of the target
    Tmin = 0.1 # min value of the target
    Tfactor = Tmax - Tmin
    V_denom = Vmax - Vmin

    return Tmin + (((value - Vmin) / V_denom) * Tfactor)

```

Figure 2. Normalization Function for Input and Output values

The weights and bias weights for the hidden nodes were randomized with values between 0.0 and 0.1 with uniform distribution. Meanwhile, the output node weights had half of its weights initialized to 1 and the other half with -1. The bias weight was initialized to a value close to zero which is 0.1. Furthermore, the same function was used when the output of the network is encoded (see Figure 2). Since there is only one output node, its value is expected to be between 0.1 and 0.99 which corresponds to FAIL or PASS, respectively.

The sigmoid function that was used for the hidden and output node is the logistic function defined in Figure 3. Ideally, the output of each node should be kept between 0 and 1. Thus, the logistic sigmoid function was adopted since it has a lower bound asymptotic to 0 and an upper bound asymptotic to 1.

```
def sigmoidFunction(value):
    # this is dE/dz
    return 1/(1+np.exp(-value)) # logistic sigmoid function
```

Figure 3 The Sigmoid/Activation Function

During backpropagation, the error derivatives of the output node and hidden nodes were calculated (See Appendix B for the equations). The resulting values were used to adjust the weighted nodes as well as their bias weights. After training, the network was subjected to validation to calculate its accuracy. Once the accuracy reaches the threshold of 50, the training will be stopped and should be tested with the test dataset. Otherwise, a new epoch is started. The algorithms implemented in this network were largely based from the multilayer perceptron discussed by (Keim, 2020) in his article entitled ‘How to Create a Multilayer Perceptron Neural Network in Python’

TensorFlow + Keras Network

Keras is a deep learning API that is built with Python on top of TensorFlow (TF) that allows the development and training of machine learning networks (About Keras, n.d.). With these tools, developing and training an ANN became relatively easier and quicker.

The ANN built using TensorFlow and Keras in Python closely follows the same structure as the Scratch Network. However, a different encoding technique was done to categorical variables with greater than two possible values using the library ScienceKit-Learn or sklearn module. The categorical variables were encoded with the OneHotEncoder (OHE) and ColumnTransformer functions. OHE transforms the values of categorical data into binary vector streams (Mohadarkar, 2021). For instance, the feature *Mjob* (Mother’s job) can have the following values: *teacher*, *health*, *service*, *at_home*, and *other*. These values could be encoded as binary vectors: 001, 010, 011, 100, and 101, respectively. With that said, the total number of input nodes became 45 which is higher as compared to the number of input nodes in the Scratch Network. Furthermore, the target outputs were also encoded into 0’s and 1’s similar to the process done with the Scratch Network. The initial numeric variables were left as it is.

Following the rule of thumb in determining the number of hidden nodes (see Equation 1), the number of hidden nodes used for the network is 31 with the sigmoid function as the activation function. The weights are randomly generated with values ranging from -0.5 and 0.5 by Keras and TF with a uniform

distribution. The sole output node has the same activation function. The resulting ANN was trained with 32 batches and 1000 epochs for both Mathematics and Portugal dataset.

Both of the dataset is divided into 60% training and 40% testing. The testing dataset was used during the evaluation of the model using the built-in function provided by Keras and TF. For future exploration, the model can also be subjected to cross validation using the entire dataset without any partitions.

The model is optimized using the Adam algorithm which, according to the Keras website, “is a stochastic gradient descent method that is based on adaptive estimation of first-order and second-order moments”. Moreover, a loss function, Binary Cross Entropy, was used since it is ideal for binary classification applications. This loss function helps identify the quantity that the network should seek to minimize during training. Lastly, each model was trained and evaluated five times to get the average loss and accuracy percentages.

Datasets

The same datasets were used to train the models developed in this study: Mathematics and Portugal. The Scratch Network had its input normalized to values ranging from 0.1 to 0.99 whereas the output variables were converted to 0's or 1's. Shown in Appendix C are the variables in the dataset placed side-by-side with their raw and normalized values for both networks. Since the PASS/FAIL classification is based on the final grades of the students, G3 is the target output of the network since it represents the students' actual final grade for the subject. Further, G3 is set to 0 when the score is less than 10. This signifies that a score less than 10 means that the student has failed the subject. Otherwise, 1 for passing the subject.

CHAPTER 3

RESULTS AND DISCUSSION

Scratch Network

For the Scratch Network, the program ran for 1000 epochs for each dataset (Mathematics and Portugal) while monitoring the network's number of correct predictions as well as percentage of correct classifications (PCC) during cross-validation. During training, the program displayed a warning where the NumPy function (i.e., the sigmoid function) that was used had overflowed (see Figure 4).

```
C:\Users\kevin\Desktop\191 Project\Actual Project\code\neuralNet_utils.py:8:
RuntimeWarning: overflow encountered in exp
  return 1/(1+np.exp(-value)) # logistic sigmoid function
```

Figure 4 Runtime Warning for Scratch Network

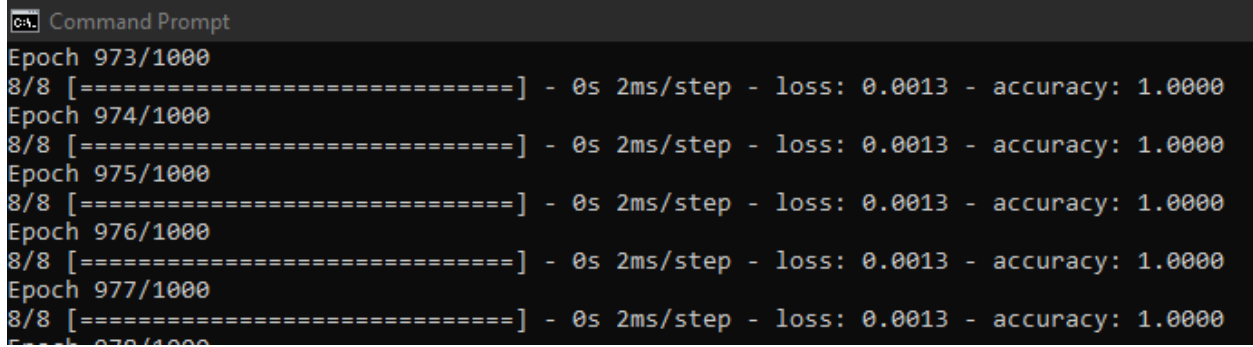
In this case, the program still continues the calculation however the values are already truncated to 0.0. That said, the model is likely to have less precise weight adjustments and upon closer inspection, the number of correct prediction count and PCC every epoch did not change as shown in Figure 5. This evidently shows that most of the weights were truncated to zeros causing the result to yield the same value every iteration. With this in mind, for further exploration, it is recommended to use other libraries that can handle infinitesimal numbers with larger precision or adjust the initial values for the weighted nodes.

```
Command Prompt
104 41 34.74576271186441
105 41 34.74576271186441
106 41 34.74576271186441
107 41 34.74576271186441
108 41 34.74576271186441
109 41 34.74576271186441
110 41 34.74576271186441
111 41 34.74576271186441
112 41 34.74576271186441
113 41 34.74576271186441
114 41 34.74576271186441
115 41 34.74576271186441
116 41 34.74576271186441
117 41 34.74576271186441
118 41 34.74576271186441
```

Figure 5 A snapshot of the ANN during training of Mathematics dataset
(from left to right: epoch-count, number of correct classifications, PCC)

TensorFlow and Keras Network

The built model also ran for 1000 epochs and each epoch logged the loss and accuracy of the model as shown in the snapshot in Figure 6. See Appendix D for the link to the document containing the entire log of training the model for Mathematics and Portugal dataset.



```
C:\> Command Prompt
Epoch 973/1000
8/8 [=====] - 0s 2ms/step - loss: 0.0013 - accuracy: 1.0000
Epoch 974/1000
8/8 [=====] - 0s 2ms/step - loss: 0.0013 - accuracy: 1.0000
Epoch 975/1000
8/8 [=====] - 0s 2ms/step - loss: 0.0013 - accuracy: 1.0000
Epoch 976/1000
8/8 [=====] - 0s 2ms/step - loss: 0.0013 - accuracy: 1.0000
Epoch 977/1000
8/8 [=====] - 0s 2ms/step - loss: 0.0013 - accuracy: 1.0000
Epoch 978/1000
```

Figure 6 Training the ANN

During trainings, the model was able to reach an accuracy of 100% as it draws near the 1000th epoch count. As summarized in Table 1, however, both models were able to achieve an accuracy of above 85% during model evaluation. This shows that both of the trained models did not overfit the training dataset as it was still able to maintain decent accuracy with the validation dataset.

Table 1 Evaluation Results of the Model (Sigmoid)

Evaluation Results (Sigmoid Function)		
Dataset	Loss (%)	Accuracy (%)
Mathematics Dataset	0.4582	0.8987
Portugal Dataset	0.6459	0.9077

The developed model was further explored with the hidden node activation function being replaced with a Rectified Linear Unit (ReLU). According to (Brownlee, 2019), the ReLU is a piecewise defined linear function that outputs the input if the input is positive, otherwise, zero. It has become the default activation function for most multilayered networks because it improves the model's training and performance. Table 2 shows the averaged evaluation results for the loss and accuracy of the models developed with the Mathematics and Portugal dataset. In comparison, it appears that there is no significant difference between the results yielded by the Sigmoid and ReLU function. Further testing with additional layers is recommended to determine which is better in terms of scaling up the model for better classifications.

Table 2 Evaluation Results of the Model (ReLU)

Evaluation Results (ReLU)		
Dataset	Loss (%)	Accuracy (%)
Mathematics Dataset	0.61686	0.8886
Portugal Dataset	0.85112	0.9008

The TensorFlow and Keras model was able to create a full-fledged model with accuracy results over 85%. For improvement, the model can be subjected to cross-validation, additional layers for the hidden network, and improved input encoding can be done. On the other hand, the Scratch Network needs improvement in terms of data preprocessing, data handling, and training algorithm. Libraries capable of handling infinitesimal numbers, optimization algorithms and properly calculated weight values are recommended for improving the model.

CHAPTER 3

CONCLUSIONS AND RECOMMENDATIONS

Generally, both models were able to yield results. However, the Scratch Network encountered overflows during training which resulted to the model not being able to improve progressively through each epoch. The same output after the first epoch remained until the 1000th epoch. Further algorithm improvements are recommended. On the other hand, the ANN built with TensorFlow and Keras yielded decent results after evaluation with accuracies greater than 85%. Additionally, the Rectified Linear Unit function was used to experiment if it would improve the performance of the model. Immediate results show that there are no significant and noticeable difference. Nevertheless, further tests are recommended especially with increased layers for the hidden nodes.

For further exploration, it is recommended to make use of additional analytical tool. In the conduct of this research, the final weights of the models were not recorded and therefore further analyses regarding those were not included. The analysis of the weights would benefit both models in determining which of the independent variables largely impacts the predicted final performance of the students. Furthermore, additionally classifications can be considered. Instead of simply classifying whether a student will PASS or FAIL, it would be better to provide more precise classes such as a four-class classification application with values: Needs Special Attention, At Risk, Satisfactory, Excellent.

Bibliography

About Keras. (n.d.). Retrieved from Keras: <https://keras.io/about/>

Bilefsky, D., & Gutierrez, J. (2021, September 13). *With Schools Closed, Covid-19 Deepens a Philippine Education Crisis*. Retrieved from The New York Times: <https://www.nytimes.com/2021/09/13/world/asia/philippines-students-remote-covid.html>

Brownlee, J. (2019, January 9). *A Gentle Introduction to the Rectified Linear Unit (ReLU)*. Retrieved from Machine Learning Mastery: <https://machinelearningmastery.com/rectified-linear-activation-function-for-deep-learning-neural-networks/>

Chen, J.-F., & Do, Q. H. (2014). TRAINING NEURAL NETWORKS TO PREDICT STUDENT ACADEMIC PERFORMANCE: A COMPARISON OF CUCKOO SEARCH AND GRAVITATIONAL SEARCH ALGORITHMS. *International Journal of Computational Intelligence and Applications*, 13(01), 1450005. doi:10.1142/S1469026814500059

Cortez, P., & Silva, A. (2008, January). Using Data Mining to Predict Secondary School Student Performance. *EUROSIS*.

Frankenfield, J. (2021, January 23). *Business Intelligence (BI)*. Retrieved from Investopedia: [https://www.investopedia.com/terms/b/business-intelligence-bi.asp#:~:text=Business%20intelligence%20\(BI\)%20refers%20to,performance%20benchmarking%2C%20and%20descriptive%20analytics](https://www.investopedia.com/terms/b/business-intelligence-bi.asp#:~:text=Business%20intelligence%20(BI)%20refers%20to,performance%20benchmarking%2C%20and%20descriptive%20analytics).

Kaya, I. E. (2019). Artificial Neural Networks as a Decision Support Tool in Curriculum Development. *International Journal on Artificial Intelligence Tools*.

Keim, R. (2020, January 19). *How to Create a Multilayer Perceptron Neural Network in Python*. Retrieved from All About Circuits: <https://www.allaboutcircuits.com/technical-articles/how-to-create-a-multilayer-perceptron-neural-network-in-python/>

Ma, Y., Liu, B., & Wong, C. K. (2000). Targeting the Right Students Using Data Mining. *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*, (pp. 457-464). doi:10.1145/347090.347184

Minaei-Bidgoli, B., Kashy, D. A., Kortemeyer, G., & Punch, W. F. (2003, December). Predicting Student Performance: An Application of Data Mining Methods With the Educational Web-based System Lon-Capa.

Mohadarkar, S. (2021, October 19). *Implementing Artificial Neural Network(Classification) in Python From Scratch*. Retrieved from Analytics Vidhya: <https://www.analyticsvidhya.com/blog/2021/10/implementing-artificial-neural-networkclassification-in-python-from-scratch/>

Nair, A. (2018, December 24). *How To Create Your first Artificial Neural Network In Python*. Retrieved from Analytics India Mag: <https://analyticsindiamag.com/how-to-create-your-first-artificial-neural-network-in-python/>

Ranjan, C. (2019, July 23). *Rules-of-thumb for building a Neural Network*. Retrieved from Towards Data Science: <https://towardsdatascience.com/17-rules-of-thumb-for-building-a-neural-network-93356f9930af>

Saavedra, J. (2021, January 5). *A silent and unequal education crisis. And the seeds for its solution*. Retrieved from World Bank Blogs: <https://blogs.worldbank.org/education/silent-and-unequal-education-crisis-and-seeds-its-solution>

The World Bank. (2018). *LEARNING to Realize Education's Promise*. Retrieved from International Bank for Reconstruction and Development/The World Bank: <https://www.worldbank.org/en/publication/wdr2018>

The World Bank. (2019, January 22). *The Education Crisis: Being in School Is NOT the Same as Learning*. Retrieved from The World Bank: <https://www.worldbank.org/en/news/immersive-story/2019/01/22/pass-or-fail-how-can-the-world-do-its-homework>

APPENDICES

APPENDIX A

Source Code

Link to GitHub repository for the ANNs built in this study:

<https://github.com/KevinGines1/neural-net-project>

APPENDIX B

The Error Derivatives

Output Node

$$\frac{\delta E}{\delta z} = z - t$$

where z is the final output of the output node and t is the value of the target output

$$\frac{\delta z}{\delta v} = z(1 - z)$$

$$p = \frac{\delta E}{\delta z} \times \frac{\delta z}{\delta v}$$

$$\frac{\delta E}{\delta b_j} = \begin{cases} p ; & \text{for } b_0 \\ py_j ; & \text{for } b_j \end{cases}$$

where b_0 is the bias weight of the output node and b_j is the weight of the output node to its corresponding hidden node connection

Hidden Node

$$\frac{\delta E}{\delta y} = \sum_{k=1}^K p_k b_k$$

where $p = \frac{\delta E}{\delta z} \times \frac{\delta z}{\delta v}$ and b_k is the weight of the hidden node

$$\frac{\delta y}{\delta u} = y(1 - y)$$

where y is the final output of the hidden node

$$q = \frac{\delta E}{\delta y} \times \frac{\delta y}{\delta u}$$

$$\frac{\delta E}{\delta a_i} = \begin{cases} q ; & \text{for } a_0 \\ q x_i ; & \text{for } a_i \end{cases}$$

where a_0 is the bias weight of the hidden node and a_j is the weight of the hidden node to its corresponding input node connection

APPENDIX C
The Dataset Variables

Feature/Target	Possible Actual Values	Scratch Network Representation	TensorFlow + Keras ANN Representation
Sex	m/f	0/1	0/1
Age	15-22	15-22	15-22
School	Gabriel Pereira/Monsinho da Silveira	0/1	0/1
Address	Urban/rural	0/1	0/1
Parents' Status (Pstatus)	Together/apart	0/1	0/1
Mother's Education (Medu)	0-4 (a)	0-4	OneHotEncoder()
Mother's Job (Mjob)	0-4 (b)	0-4	OneHotEncoder()
Father's Education (Fedu)	0-4 (a)	0-4	OneHotEncoder()
Father's Job (Fjob)	0-4 (b)	0-4	OneHotEncoder()
Guardian	0-2 (c)	0-2	0-2
Family Size (famsize)	Less than 3 (LT3)/Greater than 3 (GT3)	0/1	0/1
Family Relationship (famrel)	1-5 (1=very bad 5=excellent)	1-5	1-5
Reason for choosing school (reason)	Close to home/ school reputation/ course preference/ other	1-4	1-4
Traveltime	1 = <15min 2 = 15-30min 3 = 30min-1hr 4 = > 1hr	1-4	1-4
Studytime	1 = <2hrs 2 = 2-5hrs 3 = 5-10hrs 4 = > 10hrs	1-4	1-4
Failures	1 = 1 2 = 2 3 = 3 4 >= 4	1-4	1-4
School Support (schoolsup)	Yes/no	0/1	0/1
Family Support (famsup)	Yes/no	0/1	0/1
Extra-Curricular Activities (activities)	Yes/no	0/1	0/1
Took additional paid classes (paid)	Yes/no	0/1	0/1

Internet access (internet)	Yes/no	0/1	0/1
Took nursery (nursery)	Yes/no	0/1	0/1
Wants to take higher educ (higher)	Yes/no	0/1	0/1
Has romantic relationship (romantic)	Yes/no	0/1	0/1
Freetime	1-5 (1=very low 5=very high)	$0.1 \leq x \leq 0.99$	1-5
Goes out with friends (goout)	1-5 (1=very low 5=very high)	$0.1 \leq x \leq 0.99$	1-5
Weekend Alcohol consumption (Walc)	1-5 (1=very low 5=very high)	$0.1 \leq x \leq 0.99$	1-5
Weekday Alcohol consumption (Dalc)	1-5 (1=very low 5=very high)	$0.1 \leq x \leq 0.99$	1-5
health	1-5 (1=very bad 5=very good)	$0.1 \leq x \leq 0.99$	1-5
absences	0-93	$0.1 \leq x \leq 0.99$	0-93
G1	0-20	$0.1 \leq x \leq 0.99$	0-20
G2	0-20	$0.1 \leq x \leq 0.99$	0-20
G3 (target)	0-20	0/1	0/1

a: 0 = none, 1 = primary-4th grade, 2=5th grade to 9th grade, 3= secondary, 4=higher

b: teacher, health (health care related), service (civil service), at home, other

c: 0 = mother, 1 = father, 2 = guardian