A Comparative Study of Artificial Neural Networks and Multiple Linear Regression by Predicting Human Development Index

Ashutosh Saboo[1], Rishab Parakh[2], Parth Trivedi[3] and Dr. M. B. Potdar[4]

Abstract— Predicting Human Development Index is very crucial for every country since it's the measure of its economical and social development and measures where it stands when compared to other countries. This paper compares the Artificial Neural Network and Linear regression based approach in predicting the Human Development Index (HDI). HDI measures the county's development using four criterias as proposed by the United Nations (UN): Life Expectancy at Birth, Mean years of Schooling, Expected Years of Schooling, and Gross National Income per Capita. We outline the design of the Neural Network model and the multiple linear regression model with its salient features and customizable parameters in this study. The final test of the algorithms was on currently available HDI values, provided by the UN. The results of this experiment indicated that the neural network based approach can predict HDI with a best case accuracy of 95.3% and Linear Regression with 89.33%.

Index Terms— Artificial Neural Networks, Linear Regression, Gradient Descent, Human Development Index, Multi-layer Neural Networks, Prediction Methods, Feed-Forward Neural Networks.

1 Introduction

WE humans have a natural tendency to develop, to move forward and achieve something that has not been achieved in the past. And along with this, we humans are completive; we want to be ahead of someone, defeat someone simply because it makes us feel better. Human development is defined as a process by which we humans find a way to extend our freedom, create new and enlarge existing opportunities and in general, improve the well-being of the humans. Developing humans simply implies developing countries because as Mahbub ul Haq pointed out, a country's development should be a measure of the development of the people living in it and not just the country's economy. Countries in today's world are developing faster than they ever have in the past. And they are getting competitive about it. In this rat race for development there comes a need to measure and compare the development that nations try to achieve. Human Development Index is a measure of the economical and social development of a country. The Human Development Index takes into account three important criterias of development to classify countries into four different groups:

- Health: Life expectancy at birth is used to measure the health and access to health care facilities of the people living in the country. It uses a naive assumption that healthier people live longer.
- Education: Literacy rate is used to measure how educated the people of a nation are. Two parameters are used to measure literacy: Mean Years of Schooling and Expected Years of Schooling,
- Standard of living: This is measured by calculating the country's gross domestic product with the total population to account for the difference in population of various nations.

Haq shifted the notion of development from being totally economic to social or rather human-centric by adding the factors like Health and Education to the index that measures the development of a country as he argued that a country's economy cannot reflect the well being of the people living in them.

2 Background

2.1 Artificial Neural Networks

Artificial Neural Networks, also known as Neural Networks or Neural Nets are effective Machine Learning tools that are designed to process information in such a way that it is similar to the way the human brain processes Information, by utilizing its ability to learn new things from observations and generalization of facts from abstraction. While the human brain is a collection of about 100 billion neurons each firing signals as we think, read, write or even breathe, in an Artificial Neural Network, artificial neurons or nodes are structured and connection in hierarchical manner.

Hierarchical organization of the neural network implies that they are organized in layers. Layers are in turn made up of multiple nodes or artificial neurons. These nodes contain an activation function which carries out the mathematics involved with each Artificial Neural Network. The layers that make up a neural network are:

1. The Input Layer
2. The Hidden Layer
3. The output layer

The input pattern that we would want out neural network to learn is presented to it via the nodes in the input layer. The input layers communicate with the hidden layers where the actual processing of the input data is carried out. The hidden lay-

[1,2] *Ashutosh Saboo (ashutosh.saboo96@gmail.com) and Rishab Parakh (rishabp178@gmail.com) are students of B.E. in Computer Science and M.Sc. (Maths), respectively, in BITS, Pilani, Goa Campus, India.*

[3,4] *Parth Trivedi (prem30488@gmail.com) and Dr. M. B. Potdar (mbpotdar11@gmail.com) are Project Scientist and Project Director, respectively, at BISAG, Gandhinagar, Gujarat, India.*

ers can be linked to other hidden layers in the Neural Network. Finally the last hidden layer is linked to an output layer which gives us the final output or answer. The weights that connect these different nodes in all those layers, are constantly evaluated and modified in the training phase using some sort of a learning rule. Although there are various types of learning rules available, our Neural Network uses the delta rule. This rule is utilized by one of the most popular classes of neural network: the backpropagation neural network.

Backpropagation is short for backwards propagation of error. Here the learning is a supervised process done in cycles through forward activation flow of outputs, and the backwards error propagation of weight adjustments. Simply speaking, when a neural network is initially presented with a pattern, it randomly makes a guess on what might the weights be. Then it sees how wrong it was and then starts evaluating each and every training example to correct itself by adjusting the weights till it reaches a final satisfactory output. What backpropagation basically does is that it performs a gradient descent to reach a global minimum where the error is theoretically minimum.

2.2 Multiple Linear Regression

Multiple linear regression attempts to model the relationship between multiple independent variables and a single dependent value. It can be said that this is build up on linear regression model which is just the process of mapping an independent variable to a dependent one using a best fit line through a scatter plot spread across multiple dimensions. The most common method of doing it is called the method of least squares. This method does so by calculating the best-fit line for the scattered plot by minimizing the sum of the squares of the vertical distances from each data point to the line using gradient descent. Since these vertical distances are first squared before adding, there is no cancellation of the distances from two points one below and the other above the best fit line. What we use in our study is a special form of multiple linear regression called multiple polynomial regression. We are able to treat the polynomial regression model as a linear regression model simply because we can always consider the polynomial features to be different independent features.

Multiple times we find that the features that we use to train these multiple linear regression models are not of the same order. This is specially the case for polynomial regression. Thus, there is a need to normalize the features. The feature normalization process is carried out by normalizing each feature using its mean and its standard deviation.

Also when dealing with polynomial regression or neural networks there comes a problem of over fitting. This can be defined as fitting the given training data so perfectly that the model fails to achieve the generalization that it needs. Thus, these over fit models perform very well on the training data sets that are provided to them but fail to perform well on new test data sets. This problem can be solved by introducing a regularization term to the cost function. The weight of this regularization term is λ (lambda) also known as the regularization parameter. This λ determines how much will the cost function get penalized for over fitting.

3 Tools and Technologies

3.1 Python

From the official website "Python is a programming language that lets you work quickly and integrate systems more effectively". Python is a widely used high level programming language which follows more than one programming paradigm. It is Object-oriented, Functional and procedural. It's a dynamic programming language which means that it executes many common programming behaviors at runtime rather than at the compilation time. Python's design philosophy was for it to be readable and to achieve the same task as other languages with fewer amounts of lines of code. Python's implementation began in December of 1989 by Guido van Rossum[1]at Centrum Wiskunde & Informatica (CWI) in the Netherlands.

This study uses python along with its library openpyxl[2] for extracting the data from the excel sheets and writing it down into text files. These files are then used as input for the Neural Network and Multiple Linear Regression model to train the model and test it.

3.2 GNU Octave

GNU Octave is a high level programming language primarily intended for numerical computation. It provides very simple for various complex numerical computations. Along with this, it provides extensive graphical support for data visualization and modification. Octave is an open source version of the more intensive programming language MATLAB which is also popular for its numerical computation capabilities. Development of Octave began on 1992 by John W. Eaton[3].

We used GNU Octave in this study for the implementation of the two algorithms for multiple linear regression and artificial neural network because if its numerical computation capabilities and its graphical support for plotting the various graphs that were used in the process of developing the algorithms for the two models. Octave was chosen over other languages for the implementation of the algorithm because of the ease with which it handles matrix operations so that we could focus more on the actual working of the algorithm and less on its implementation.

4 The Models for Predicting Human Development Index

4.1 The Artificial Neural Network Model

Artificial Neural Networks (ANN) offers a very intelligent and efficient way to perform tasks similar to the human brain. ANNs comprise of the input layer, the hidden layers, and the output layer which can be configured in various ways to generate a higher accuracy.

In our ANN model for predicting HDI, as part of the input layer, we generate all possible cube root, square root, single, and squared powers of Life Expectancy at Birth (LEB), Mean ears of Schooling (MYS), Expected Years of Schooling (EYS), and Gross National Income per Capita (GNIP). In addition to this, we also add the logarithm of these terms to the input layer as well. In addition, to this we also ensure that all the input features and scaled to a common scale, and they are normalized as well, so that one feature doesn't heavily impact the hidden layer weights in comparison to the other. The input feature x is normalized using the equation - $x = (x - \mu)/\sigma$, where μ = average value of the feature x , and σ = standard deviation of the feature x.

For the hidden layer, we set a fixed number of 100 hidden neurons as part of our hidden layer. These neurons in the hidden layer perform the necessary computation, which help us in predicting the HDI values. As part of the ANN model, weights are assigned to the hidden layer, which in turn affect the values computed by the hidden layer neurons as well. Regularization is also implemented in the ANN, to prevent overfitting of graphs on the training set, so as to prevent a very high error on the test set.

ANNs are provided with basically 3 sets of data - Training set, cross validation set, and the test set. The training set is used for the ANN to predict the hidden layer weights, based on the input data provided. The Cross Validation set is used to validate and set the regularization parameter, which gives the best results. Finally, the Test set is used to test the ANN with the initialized weights, and the regularization parameter to determine the final accuracy of the ANN.

The hidden layer weights are initialized randomly, and subsequently get trained automatically by the feed-forward and back-propagation methods of the ANN, based on the input training data provided to the ANN.
As part of the output layer of our ANN model, we have a single neuron, whose value ranges from 0 to 1, and predicts the HDI values of countries, The difference between the actual HDI value and the predicted HDI value is the error in each test set provided to the ANN, and the accuracy of the ANN model is determined by the percentage of the test set errors with the range of -0.03 to 0.03.

4.2 The Multiple Linear Regression

Multiple linear Regression (MLR) is used to predict the relationship between all the input variables that are fed to the multiple linear regression model. A training set is provided to our MLR model, which helps to predict the best-curve that fits the training data, with the most minimum errors, and finally a testing set is provided, which helps us to check the value predicted by the MLR model, and also measure the accuracy. Multiple linear regression basically helps to predict the best fit curve, based on the input training data provided to it, and then extrapolates the curve to predict the value for the testing set.

So, as a part of comparing the accuracy provided by the MLR and the ANN, for consistency purposes, we provide the same input features to our MLR model as we provided to the ANN model, that is all the possible cube root, square root, single, and squared powers of Life Expectancy at Birth (LEB), Mean years of Schooling (MYS), Expected Years of Schooling (EYS), and Gross National Income per Capita (GNIP), along with their logarithms as well. In addition, we also ensure that all the input features are scaled to a common scale, and they are normalized as well, so that one feature doesn't heavily impact the weights in the hypothesis function, in comparison to the other.

The input feature x is normalized using the equation – $x = (x - \mu)/\sigma$ where μ = average value of the feature x, and σ = standard deviation of the feature x.

5 Training, Cross-validating and Testing the Model

5.1 The Artificial Neural Network Model

So, we set up our ANN model, then we needed to train and cross-validate it, with existing HDI values of countries over the past few years, so that our ANN model can predict HDI values with the best possible accuracy.

Now for training, cross-validating, and testing purposes, we first extract the HDI values of all countries over the past 4 years, provided by the United Nations (UN). This forms our entire set of available data for supervised learning. The entire data is randomly re-arranged, and 60% of it is made the training set, which will be used to train our ANN model. The next 20% of our total available data is used as a cross-validation set, and the next and final 20% is used as a testing set, which helps us to finally predict the accuracy of our ANN model.

So, we start with making our ANN model learn in the supervised way, by initializing the weights in the hidden layers of our ANN randomly. Then, our ANN model starts to get trained as we provide our "training set" to our ANN model. The weights of our ith hidden layer our called where, m is the mth unit in the i-1th layer, and n is the nth unit in the ith hidden layer. These hidden layer weights, which connect from the mth hidden layer neuron in the i-1th layer to the nth hidden layer neuron in the ith layer help us in calculation the values in the hidden layer neurons in the following way-:

i) The values that get calculated in the hidden layer neurons are represented by certain variables - s_j^i and a_j^i , where j represents the jth hidden layer neuron, and i represents the ith hidden layer.

ii) Now, s_j^i is calculated using the following formula,
$s_j^i = \Sigma\theta_{ij} a_j^{i-1}$.
iii) Also certain activation functions are part of our ANN Model, which help us in computing the values generated by the hidden layer neurons. These are (by convention) represented by g(z). Also, $a_j^i = g\left(s_j^i\right)$.

iv) As a part of our ANN model, we use the hyperbolic tan function as our activation function, i.e., $g(z) = \tanh(z)$, which ranges from (-1,+1). Now, backpropagation is also used as a part of our ANN model, so that the hidden layer weights improve every iteration. As a part of training process for our ANN for predicting HDI, we train our ANN with the "training set" (specified above) for maximum 1000 iterations.

In addition to this, we also implemented regularization for our ANN, which is necessary to prevent over fitting of curve, in the training process, and hence, produce a lesser error with the testing set. The regularization parameter λ is initially initialized to 0, and then progressively increased by every 0.01, and is increased to maximum 1. This is where we feed the cross-validation set to our ANN, to validate the weights predicted in the training phase, as well as to confirm the regularization parameter λ best for our neural network. For every λ, the ANN calculates the cost using the weights calculated in the training phase, on the cross-validation set. In the process, the λ which produces the least error on the cross-validation set is chosen as the final λ for the testing phase. Now, as a part of the testing phase, the test set is fed to the ANN, and further the ANN calculates the cost on the test set using the already initialized weights in the training phase, and the final regularization parameter, which was calculated in the cross-validation phase. This further gives us the error of each test example, as the difference between the actual HDI value and the predicted HDI value, and the final accuracy of our ANN model is determined by the percentage of test set example errors within the range on [-0.03,0.03]. This is how our ANN is tested to determine the final accuracy.

5.2 The Multiple Linear Regression Model

As we set up our MLR model, it is also essential to train it, so that the hypothesis function gives the minimum possible-cost on the test set.

For consistency purpose, the entire training set along with the cross-validation set, as specified in the ANN model, form the testing set for our MR model, that is, 80% of the total available HDI values of all countries over the span of 4 years. Similarly, the testing set of our ANN model, remains the same, and is used as the testing set for our MR model, again for consistency purpose, so that we can easily compare the results of both the models at the end.

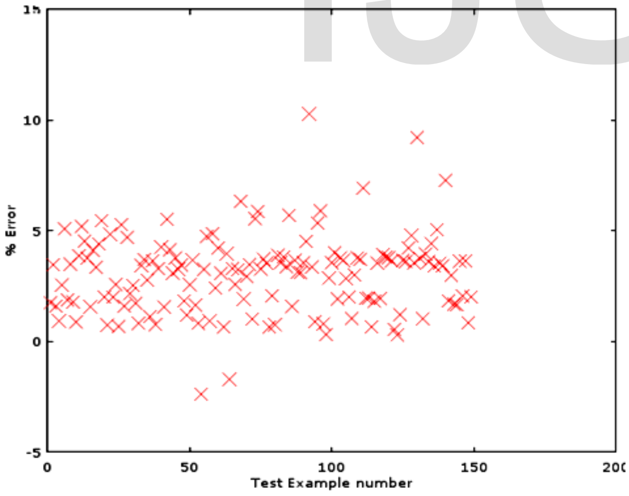


Figure 1: Percentage errors of the Artificial Neural Network model.

The MR model has a hypothesis function associated to it - $h_\theta(x) = \Sigma\theta_i x_i$. These θ values are basically the weights associated with each input feature xi, in the hypothesis function. The value of these θ values are predicted by our MLR model, in the training phase. So, we feed the training set to our MLR model, and the model uses gradient descent to calculate the best possible values of the θ values, such that the hypothesis function best-fits the training data.

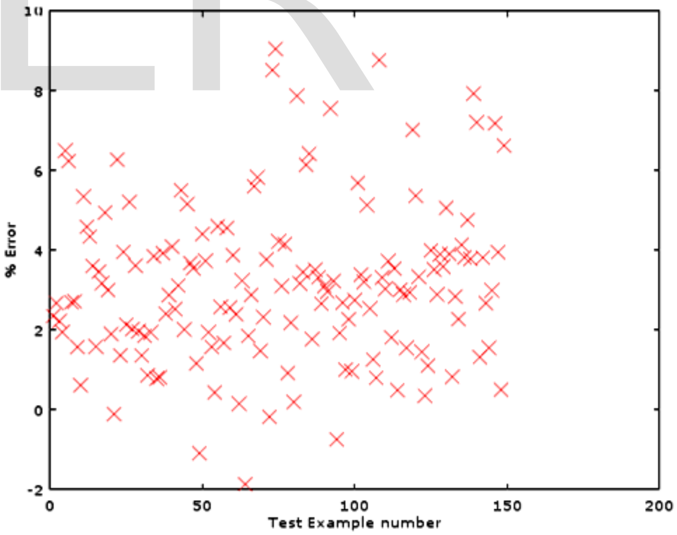
$$\theta_j = \theta_j - \alpha \frac{\partial}{\partial\theta_j} J\left(\theta_0,\theta_1,\theta_2,...,\theta_n\right)$$

For our training process, to establish consistency, again, we train our training set with the MR model for a maximum number of 1000 iterations, which is same as the ANN model.

α is known as the learning rate of the gradient descent function, and it needs to be specified by the user. Typically it's not chosen way too large, and neither way too less. For our model, we basically iterate the entire training phase on a range of values of α, from 0 to 0.01 over an increment of every 0.001, and then choose the value of α that gives us the least cost on the training set. For our own model, α = 0.009 gives us the least possible cost on our training set.

This is how the MR model is trained, and once the hypothesis function is calculated in the training phase, then the testing phase begins. So, the test set is fed to the MR model, and our MR model, calculates the cost on the testing set, by using the hypothesis function (calculated in the training phase). The error in each test example is determined by the difference in the actual HDI value and the predicted HDI value by our MR model. Again, for consistency purposes, the accuracy of the MR model is determined by the percentage of test set example errors within the range of [-0.03, 0.03], just like the ANN model, alue anc

Figure 2: Percentage error for the Multiple Linear Regression model.



6 Conclusion

The Human Development Index is calculated by the United Nations, and its formula changes periodically after several years, depending on the current level of world economy, and various other factors as well in a very theoretic way.

Automating the Human Development index is indeed a real-world prediction problem, and can also help in saving a lot of theoretical work.

Table 1: Comparision of the predicted HDI values with the actual HDI values

Country (Year)	Actual HDI	HDI Predicted by ANN	HDI Predicted by MLR
Belize (2011)	0.69882	0.71777	0.71098
Belgium (2013)	0.88078	0.89467	0.91121
Palau (2011)	0.78156	0.79946	0.79399
India (2014)	0.60869	0.60686	0.63097
Zambia (2012)	0.44772	0.46548	0.45909
Germany (2013)	0.91143	0.92559	0.94191
Paraguay (2011)	0.66473	0.68442	0.67712

We successfully modeled our Artificial Neural Network (ANN), as well as our Multiple Linear Regression (MLR) to predict the Human Development Index of any given country in the upcoming years, once the required input features are provided correctly to our program. Our ANN model achieves a best-case accuracy of 95.3%. On the other hand, our MLR model only achieves a best-case accuracy of 89.933%.

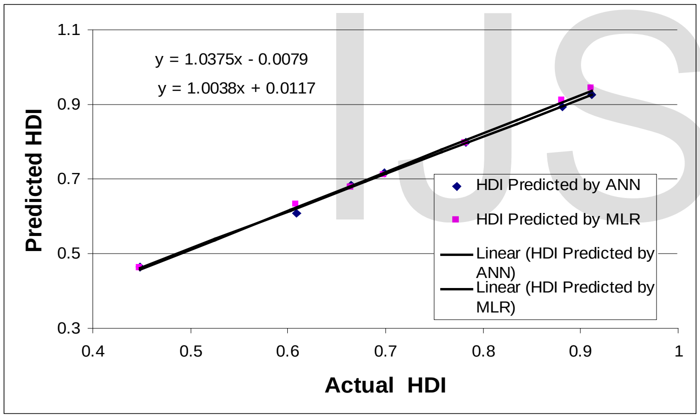


Figure 3: Comparision of Predicted HDI with the actual HDI based on the ANN and MLR techniques.

All in all, our Neural Networks representation of predicting the HDI helped us to predict the same, in a more accurate way than the Multiple Linear Regression model, giving us error ranges majorly in the range of -0.03 to 0.03 only.

8 Data Providers

Majorly all the data we got was in the format of excel sheets which had to be converted into text format for our algorithm.

[1] http://hdr.undp.org/sites/default/files/hdr14_statisticaltables.xls

[2] http://hdr.undp.org/sites/default/files/2015_statistical_annex_tables_all.xls

[3] http://www.pnud.org.br/atlas/ranking/HDR_2011_Statistical_Tables.xls

[4] http://www.iser.uaa.alaska.edu/people/colt/personal/econ650_su03/hdi.

7 References

[1] Ms. Sonali. B. Maind and Ms. Priyanka Wankar 2000. Research Paper on Basic of Artificial Neural Network

[2] Lippmann, R.P., 1987. An introduction to computing with neural nets. IEEE Accost. Speech Signal Process. Mag., April: 4-22.

[3] N. Murata, S. Yoshizawa, and S. Amari, ―Learning curves, model selection and complexity of neural networks,‖ in Advances in Neural Information Processing Systems 5, S. Jose Hanson, J. D. Cowan, and C. Lee Giles, ed. San Mateo, CA: Morgan Kaufmann, 1993, pp. 607-614

[4] Pauziah Mohd Arsad, Norlida Buniyamin, Jamalul-lail Ab Manan 2014. Neural Network and Linear Regression Methods for Prediction of Students' Academic Achievement

[5] Predrag Ćosić, Dragutin Lisjak, Dražen Antolić, Regression Analysis And Neural Networks As Methods For Production Time Estimation.

[6] Carlos Gershenson, "Artificial Neural Networks for Beginners", United Kingdom.