# Linear Regression and Multi-Layer Perceptron Models for predicting Happiness Score

Pranav Bajoria
CIDSE
Arizona State University
Tempe, Arizona, USA
p.bajoria@asu.edu

Naga Sai Teja Vinnakota
CIDSE
Arizona State University
Tempe, Arizona, USA
nvinnako@asu.edu

***Abstract*— A World Happiness Report was released by the UN in 2017, which ranked 155 country based on their happiness scores. This data can be useful for statistical analysis in the domain of economic growth, Psychology, foreign affairs and the overall country's progress. Thus, predicting a country's happiness score at any given point can prove to be quite valuable for a country's progress as it can help them make key decisions. In this report we apply and compare two different machine learning regression models on the world happiness dataset taken from Kaggle to predict happiness score of a country based on its features like GDP, Life expectancy, corruption, generosity, freedom and family.**

## I. Introduction

The world happiness dataset was prepared based on a survey where people were asked to answer to a poll which had the main life evaluation question. The scores are given from 0 to 10 for 0 being the worst possible life and 10 being the best possible life. This evaluation criterion is known as the Cantril Ladder. The features of the dataset are scores driven by factors like GDP, life expectancy, social support, freedom, absence of corruption and generosity. Each feature's score estimates the extent to which that factor contributes to the life evaluation being higher than they are in a hypothetical country whose scores are lowest national averages for each of these fields.

The hypothetical country is called Dystopia. There is yet another important column to note in the dataset which is Dystopia residual. This value is the Dystopia's happiness score summed with unexplained components of each country, which reflects the extent to which the six factors fail to explain the evaluations. The summation makes sure that all the values remain positive for cases when the residual is a result of over-explanation.

## II. Method / Approach

### A. *Data Pre-processing*

The available dataset has 3 sheets of data of the 155 countries for each year from 2105-2017. Since machine learning models perform better for larger data, we can combine the 3 by swapping columns and dropping obviously unnecessary columns like Country, Happiness Rank, Whisker High, Whisker Low, Lowe confidence Interval, Upper Confidence Interval, Region, standard error. This is done in such a way that all the three sets of data have the same number of columns in the same order. Then the three sets of data are appended together to form a single data-frame. The Pandas library was used to perform these operations. The final size of the data was 470 X 8.

### B. *Selecting the optimum features for training the models.*

From the data it is evident that the sum of all the fields following the happiness score results to the happiness score since the data is well setup and normalized for use. Hence the models trained using all the 7 feature columns might be unreliable to do prediction. For prediction, the residual feature can not be calculated as it is simply a post processed factor. Hence to tackle this, we followed 2 approaches. First, completely dropping the Dystopia residual column and then performing Machine Learning to predict score. Second approach was to perform statistical analysis of this information and trying to utilize it.

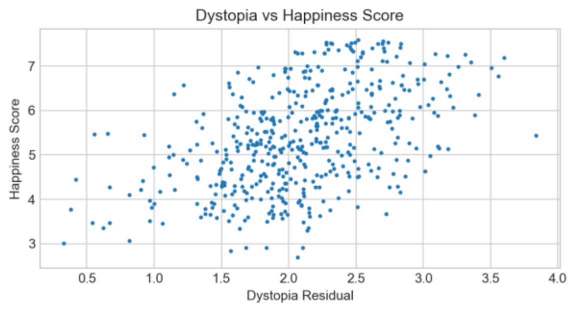


*Figure 2: Dystopia vs Happiness Score plot*

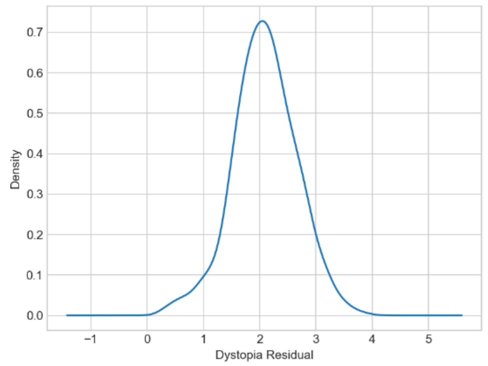


*Figure 1: PDF plot of Dystopia Residual*

Upon plotting the Dystopia residual score for each country with respect to Happiness score, it was found in Figure 2 that the scores seemed to follow a normal distribution, i.e. most of the countries seemed to have the same score. This is like that of the estimated happiness score for the hypothetical country Dystopia. So, subtracting the mean of the dystopia residual column from happiness scores of each country will not affect the result, but help the model perform better. The model thus learnt can predict a value and then the dystopia residual mean can be added to get the value to it to get the final happiness score. This was our approach 2 to see if the model performed better.

There are also possible scenarios where all the features are always not available for predicting the happiness scores. To perform further analysis on the features that can be selected in such cases, we plotted their covariance matrix to understand how each factor was correlated with the happiness score and each other.

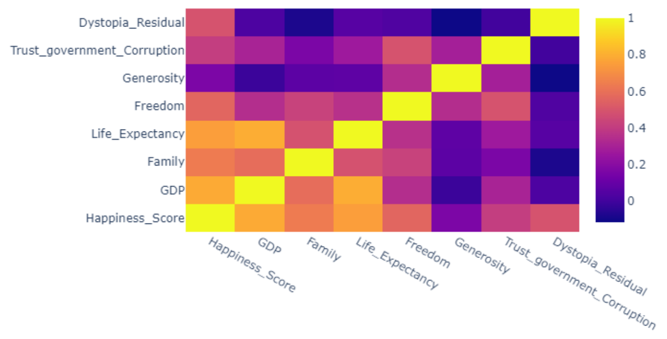


*Figure 3: Covariance matrix of all the features*

To understand the correlation of each feature with respect to happiness, scattered plots were generated as shown in Figure 4.

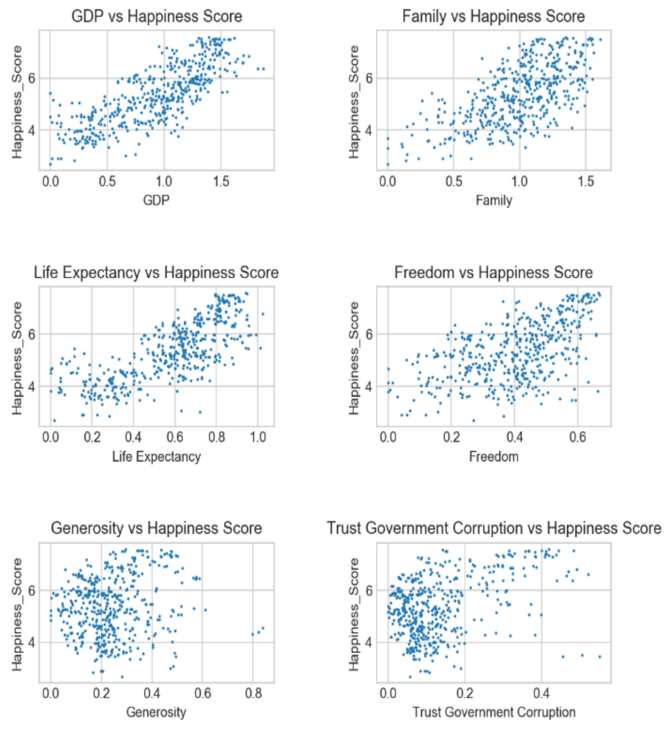


*Figure 4: Scatter plots to show feature correlation with label*

We can observe that Happiness score varied in high correlation with GDP and Life expectancy, moderately with freedom and family. Happiness score had very little affect of the change in generosity and trust Government corruption, which is quite strange. Since most people were seen to have little trust in the government, their happiness is less affected by this factor.

To demonstrate this, we train the Linear regression and MLP models by picking each of these feature pairs for prediction. The expected results should be least RMSE for GDP-Life expectancy, higher for Family-Freedom and highest for Generosity-Trust government.

## C. *Machine Learning*

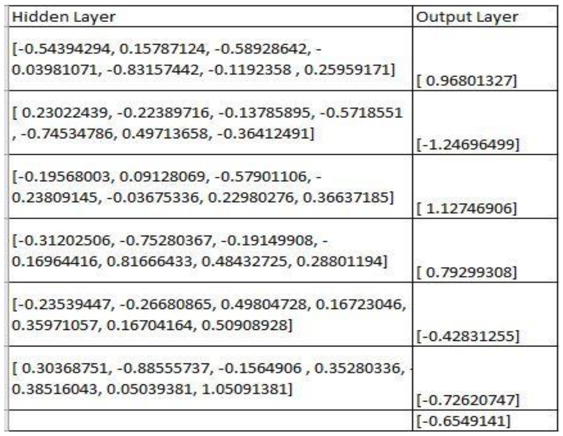
On this dataset, we perform analysis and comparison of the two machine learning tools – Linear Regression Model and a Multi-Layer Perceptron Model (MLP). To measure their performance, we have used Root Mean Square Error as the metric, since it is a prediction problem. Since the data is highly corelated, Linear Regression model is expected to perform good, but our aim was to make the MLP model to perform at least as good.

Since MLP can act like a linear regression in certain conditions, we were positive to get the RMSE of the MLP to be as less as the Linear Regressor. To achieve this, we fine-tuned the parameters of the MLP, like changing the number of neurons in the hidden layer, activation function and the number of iterations. The optimum parameters of MLP when trained on all the features except the Residual score is given in the Table 1.

*Table 1 Optimum parameters for MLP*

| **Parameters** | **Values** |
|---|---|
| No. of neurons in the layer | 7 |
| Learning rate | 0.01 |
| No. of iterations | 2500 |
| Activation function | “Tanh” |

*Table 2 Weight vectors of the parameters learnt in MLP*



| Hidden Layer | Output Layer |
|---|---|
| [-0.54394294, 0.15787124, -0.58928642, -0.03981071, -0.83157442, -0.1192358 , 0.25959171] | [ 0.96801327] |
| [ 0.23022439, -0.22389716, -0.13785895, -0.5718551 , -0.74534786, 0.49713658, -0.36412491] | [-1.24696499] |
| [-0.19568003, 0.09128069, -0.57901106, -0.23809145, -0.03675336, 0.22980276, 0.36637185] | [ 1.12746906] |
| [-0.31202506, -0.75280367, -0.19149908, -0.16964416, 0.81666433, 0.48432725, 0.28801194] | [ 0.79299308] |
| [-0.23539447, -0.26680865, 0.49804728, 0.16723046, 0.35971057, 0.16704164, 0.50908928] | [-0.42831255] |
| [ 0.30368751, -0.88555737, -0.1564906 , 0.35280336, -0.38516043, 0.05039381, 1.05091381] | [-0.72620747] |
|  | [-0.6549141] |

After training the models, the linear regression model learns 7 parameters as 6 coefficients and a bias.

$$Y_p = W_1X_1 + W_2X_2 + W_3X_3 + W_4X_4 + W_5X_5 + W_6X_6 + b$$

The learnt parameters are as follows:
W: [1.015, 0.6725, 1.251, 1.524, 0.358, 0.8585], b: 0.1269

Since the number of neurons in the hidden layer is 7 and the features are 6, each neuron gets 7 parameters, the total number of parameters learnt in the hidden layer is 7*7 = 49. In the output layer, it is 7 weights learnt from the 7 neurons of the hidden layer. The Table 2 shows the weight vectors of each layer.

## *D. Model Selection*

The main aim of this procedure was to tune the hyper-parameters like the number of neurons in the hidden layer in the hope that the RMSE would further decrease and we would get the best multi-layered perceptron model for prediction of happiness score. This process was done individually by both the authors. I, (author 1), have presented my findings as below.

I used cross validation to test the RMSE of different values of number of hidden neurons. Changing this would change the complexity of the model. Cross validation is a key method used when selecting the right parameters because when a model is run on the real data for prediction, it should neither be too biased and nor be too complex. A good model is that works better on the real time data. Therefore, it is vital to have a separate test data-set to check whether the selected hyperparameters work good enough. The most conventional way of doing this is by splitting the available data into train and test portions.

For dividing the train and test data, I used the train_test_split method from the sklearn.modelselection library. I chose an optimum 70-30% split for training and testing.

Cross validation involves training different models based on different hyper-parameters and testing the loss function computed on the same test data. This process "validates" the error on the same rubric.

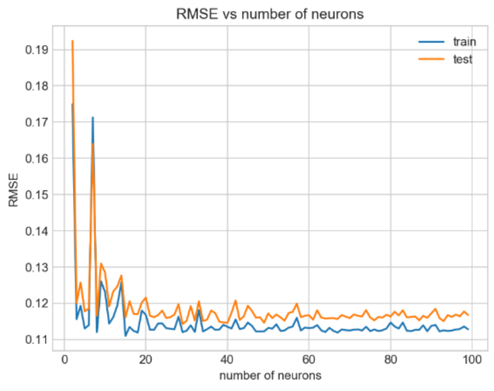


*Figure 5: RMSE on test Data for models trained on different number of neurons for Approach 1*

In the current case, I implemented cross validation to tune the number of neurons. In a loop, I kept the number of neurons value increasing from 2 to 100 and trained each model on the test-data portion of the split. For this to work, it is important to keep the rest of the set-up constant. For eg. all the other parameters like learning rate, activation function, number of iterations, random_state should not change. In each of these iterations, I tested the performance of each model on the same test data by measuring the RMSE in each iteration.

Each iteration's RMSE and number of neurons value was stored in a list to be plotted later. The plots are shown in the figures **Error! Reference source not found.** & Figure 6.

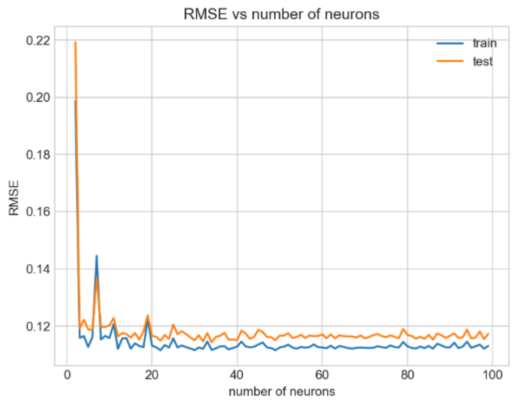


*Figure 6: RMSE on test Data for models trained on different number of neurons for Approach 2*

From the plots, we could see that after a certain value of the number of neurons, the RMSE did not change much for both the training and testing data. I selected the number of neurons for which the RMSE was the least in the range.

The results can be shown below:
For approach 1:
optimum number of neurons = 29, RMSE = 0.11428
For approach 2:
optimum number of neurons = 34, RMSE = 0.1143

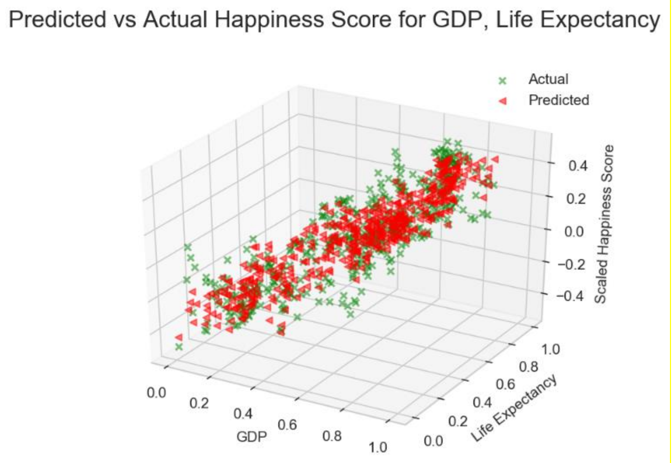


*Figure 7: Selected Model's performance for approach 1*

The model's performance can be seen in the predicted vs actual happiness values plots in the figures.

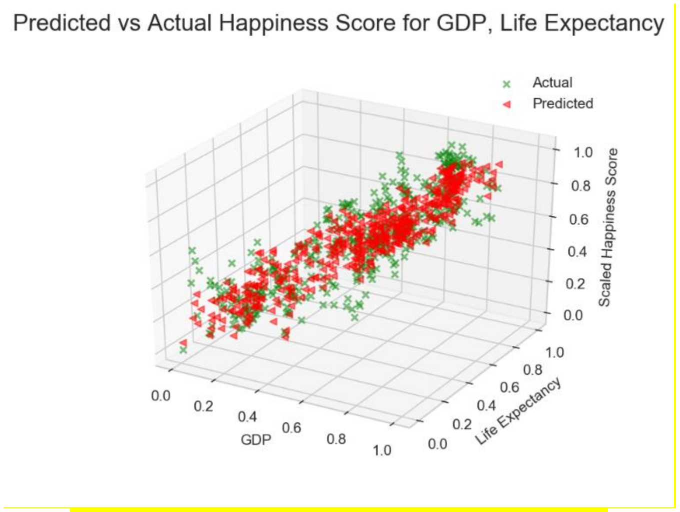


*Figure 8: Selected Model's performance for approach 2*

It can be observed that the model performed better after the model selection procedure by cross validation as RMSE decreased from 0.5 to 0.1.

## III. RESULTS

The results of the performance of the model is plotted and shown in the figures Figure 9, Figure 10. The RMSE values can be found in tables Table 2 and Table 3.

*Table 3: RMSE for the two approaches*

| Features Selected | RMSE | |
|---|---|---|
| | Linear Regression | Multi Layer Perceptron |
| All Features | 0.00028 | 0.032 |
| All Features except Dystopia Residual | 0.5525 | 0.538 |
| All features with modified happiness score | 0.5525 | 0.5273 |

### *A. Approach 1- Models trained Without Dystopia Residual*

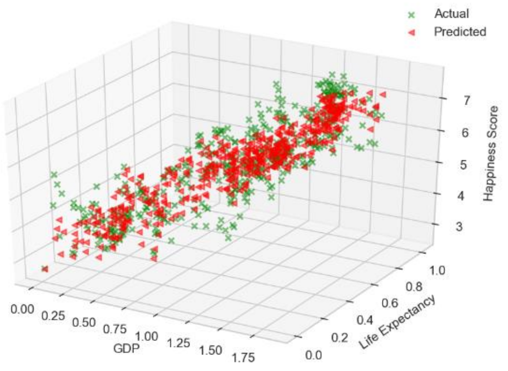


*Figure 9: Predicted vs actual happiness score*

### *B. Approach 2 – Models trained to fit modified happiness score (Score – mean of Dystopia Residual).*

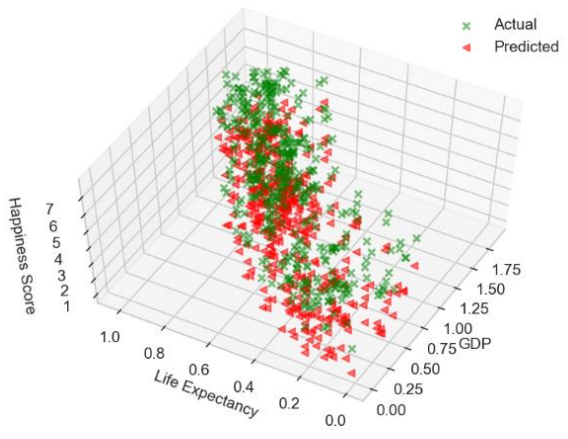


*Figure 10: Results comparison for modified Score*

### *C. Model trained with other pairs of features.*

| Features Selected | RMSE | |
|---|---|---|
| | Linear Regression | Multi Layer Perceptron |
| GDP, Life Expectancy | 0.6628 | 0.6634 |
| GDP, Family | 0.6579 | 0.6629 |
| GDP, Generosity | 0.6743 | 0.6758 |
| Generosity, Trust Government Corruption | 1.0363 | 1.0365 |
| Family, Freedom | 0.797 | 0.7972 |
| Generosity, Dystopia Residual | 0.9578 | 0.958 |

## IV. DISCUSSIONS & CONCLUSION

The most important objectives of the study were to efficiently architect and implement Linear Regression and Multi-Layer Perceptron models on the World Happiness Dataset. The key inferences drawn were, i) number of features in case of both linear regression and MLP are important for the model's performances; ii) tuning the hyperparameters of a multi-layer perceptron can result in a much better performance than default parameters; and iii) the selection of a Machine Learning model depends highly on what the data's size and distribution.

Secondly, the results show that the linear regression and the Multi-Layer Perceptron models have a similar Root Mean Square Error. This could be the case due to the fact that the data is well laid out, i.e. it is normalized, cleaned, and ready for use. The MLP could have performed better than the Linear Regression model if the data was larger in size or not perfectly linear.

## REFERENCES

[1] World Happiness Report, Kaggle, June 14, 2017, URL: https://www.kaggle.com/unsdsn/world-happiness

[2] Predicting World Happiness, Kaggle Kernel by Manali Shinde, URL: https://www.kaggle.com/mshinde10/predicting-world-happiness

[3] World Happiness Report Visualization, blog by Eric Adlard, URL: https://nycdatascience.com/blog/student-works/world-happiness-report-visualization