Using Machine Learning to predict World Happiness Index

Abstract— **In this assignment we created a linear regression model for the happiness score and recorded the Root Mean Square Error of the model. We then trained a multilayer perceptron model and tried to get the RMS error equal to the one obtained in Linear Regression. We used the Kaggle Happiness Dataset to train and test our model.**

Keywords—Linear Regression, MLP, RMSE Error

I. INTRODUCTION

The dataset that we have chosen is happiness 2017 dataset, one of Kaggle's dataset. This dataset gives the happiness rank and happiness score of 155 countries around the world based on seven factors including family, life expectancy, economy, generosity, trust in government, freedom, and dystopia residual. Sum of the value of these seven factors gives us the happiness score and the higher the happiness score, the lower the happiness rank. So, it is evident that the higher value of each of these seven factors means the level of happiness is higher. We can define the meaning of these factors as the extent to which these factors lead to happiness. Dystopia is the opposite of utopia and has the lowest happiness level. Dystopia will be considered as a reference for other countries to show how far they are from being the poorest country regarding happiness level. The idea is fit a multivariate linear regression model to the data and see which features have the most impact on Happiness.

II. GENERAL BACKGROUND

A. Perceptron

A perceptron is a linear classifier; that is, it is an algorithm that classifies input by separating two categories with a straight line. Input is typically a feature vector x multiplied by weights w and added to a bias b: y = w * x + b.

A perceptron produces a single output based on several real-valued inputs by forming a linear combination using its input weights (and sometimes passing the output through a nonlinear activation function). Here's how you can write that in math:

$$y = \phi\left(\sum_{i=1\ to\ n} w_i x_i + b\right) = \phi(w^T x + b)$$

A multilayer perceptron (MLP) is a deep, artificial neural network. It is composed of more than one perceptron. They are composed of an input layer to receive the signal, an output layer that makes a decision or prediction about the input, and in between those two, an arbitrary number of hidden layers that are the true computational engine of the MLP. MLPs with one hidden layer are capable of approximating any continuous function.

Multilayer perceptrons are often applied to supervised learning problems3: they train on a set of input-output pairs and learn to model the correlation (or dependencies) between those inputs and outputs. Training involves adjusting the parameters, or the weights and biases, of the model in order to minimize error. Backpropagation is used to make those weigh and bias adjustments relative to the error, and the error itself can be measured in a variety of ways, including by root mean squared error (RMSE).

B. Linear Regression

A regression is basically where we are trying to predict the values of a function at a value of an input. In a linear regression, we try to fit a line to the data given (input features and output class labels/predictions) and predict what values we have for different inputs.

$$y = \theta_0 + \theta_1 x$$

The values θ_0, θ_1 are called the parameters of the regression. We use gradient descent to find the parameter values which minimize the cost function i.e minimize the distance between the predicted values vs the actual values.

C. Root Mean Square Error

The Root Mean Square Error or RMSE is a type of cost function which helps us understand how well the data fits our prediction. Lesser the error, the greater the accuracy of our hypothesis/model. It calculates the average of the squares of the distances between the predicted values and true values for the training data. The RMSE can be mathematically written as:

$$J(\theta) = 1/n \sum_{i\,=\,1\ to\ n} (pred_i - x_i)^2$$

$$= 1/n \sum_{i\,=\,1\ to\ n} (\theta_0 + \theta_1 x_i - x_i)^2$$

III. METHOD AND APPROACH.

To build our Linear Regression and Multi-Layer Perceptron models we used python based sklearn library . For the data, we ran multiple tests- on data separated year wise, as well as, combined data from all years. Though we may fit the smaller dataset well, there could be errors during generalization over new data . This can be reduced by training the models on more data. Hence, combining the data from all the different years may be beneficial in this regard.

Libraries Used: NumPy, sci-kit, sk-learn, Pandas, seaborn

The features we used were : Economy (GDP per Capita), Family, Health (Life Expectancy), Freedom, Trust (Government Corruption), Generosity, Dystopia Residual. Here, we only have those features that were common to all the 3 years so that we could combine the datasets effectively. We have also normalized the 7 input features to bring them into a common scale.

To understand the features given better we did a quick analysis of correlation between different features . We did this by plotting a correlation matrix using the Python library -

Seaborn. Here the most correlated correspond to 1 and the least correlated ones correspond. We do this because in a multiple regression model - Multicollinearity- where one predictor variable can be linearly predicted from the others with a high degree of accuracy, can lead to skewed or misleading results.

Here we can observe that none of the features we used for our LINEAR REGRESSION and MLP modelled are highly correlated . Happiness scores are highly correlated to Economy(GDP per capita) and Health (Life Expectancy). But , this does not affect us as we are not considering Happiness score as a feature space.

A. *Fitting values for Linear Regression and MLP*

We could easily fit linear regression models using the default parameters provided by the sklearn library.

To train our MLP models to data, we tried changing the different parameters available to us and repeating the fitting process. 2 of the most important hyperparameters that play an important role in fitting MLP models are the number of layers and the number of nodes in each layer. In our work we could get good results by modifying the number of nodes itself.

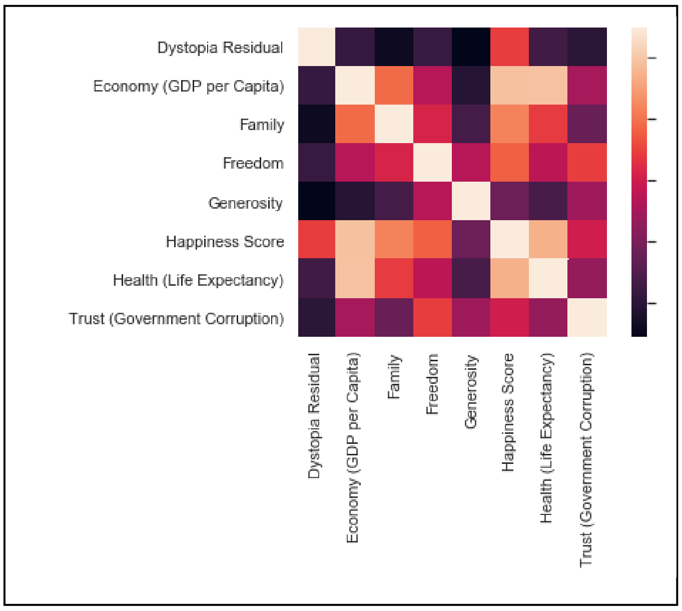


Fig: Plotting a heatmap (correlation matrix) between features and Happiness Score, we see that health and Economy are most correlated with Happiness.

During this process , we observed that the least square error from our MLP model decreased as we increased the number of nodes in the hidden layers we used even though we kept the number of hidden layers fixed at 1 and we got best results at 100 nodes in the hidden layer. We also tried changing the random state of our algorithms and tests. This helped us get slightly better results. Here, we did not change other parameters and used the default values from the sklearn library itself as the model converged without changing them.

IV. Results

Set 1: We got best results for following parameters values for the different datasets:

Activation Function	ReLU
Solver	ADAM

1) For 2017:

LINEAR REGRESSION Score :0.9991124250190904

MLP Score :0.9782680407323764

Mean squared error LINEAR REGRESSION: 0.03

Mean squared error MLP: 0.16

Features are : Economy (GDP per Capita), Family, Health (Life Expectancy), Freedom, Trust (Government Corruption), Generosity, Dystopia Residual

LINEAR REGRESSION Weights : [1.93858898 1.57298228 0.85721453 0.66032029 0.85100486 0.434130392.73542981]

MLP Params :

Hidden Layer Nodes	50
Max Iterations	1000
Random State	0

2) For all 3 years combined:

LINEAR REGRESSION Score :0.9922133231511345

MLP Score :0.9903180813036601

Mean squared error LINEAR REGRESSION: 0.10

Mean squared error MLP: 0.11

Features are : Economy (GDP per Capita), Family, Health (Life Expectancy), Freedom, Trust (Government Corruption), Generosity, Dystopia Residual

LINEAR REGRESSION Weights : [1.84496879 1.30329517 0.92689753 0.62912033 0.59595285 0.68374014 2.94324781]

MLP Params :

Hidden Layer Nodes	100
Max Iterations	1000
Random State	5

Set 2: In this set of results we present the results for parameters :

No of hidden layers =1

No of hidden layer nodes =100 .

Activation Function	ReLU
Solver	ADAM

Hidden Layer Nodes (MLP)	100
Max Iterations (MLP)	1000
Random State (MLP)	5,1

3) For 2016:

LINEAR REGRESSION Score :0.9999999336049087

MLP Score :0.8689587676689297

Mean squared error LINEAR REGRESSION: 0.00

Mean squared error MLP: 0.41

Features are : Economy (GDP per Capita), Family, Health (Life Expectancy), Freedom, Trust (Government Corruption), Generosity, Dystopia Residual

LINEAR REGRESSION Weights : [1.82425337 1.18328866 0.95262959 0.6085824 0.50505877 0.81985295 3.01967172]

4) For all 3 years combined:

LINEAR REGRESSION Score :0.9922133231511345

MLP Score :0.9875227005156413

Mean squared error LINEAR REGRESSION: 0.10

Mean squared error MLP: 0.13

Features are : Economy (GDP per Capita), Family, Health (Life Expectancy), Freedom, Trust (Government Corruption), Generosity, Dystopia Residual

LINEAR REGRESSION Weights : [1.84496879 1.30329517 0.92689753 0.62912033 0.59595285 0.68374014 2.94324781]

PHASE II

For this phase we experimented with networks having with 2 hidden Layers. Below are the best results for datasets having data from years 2015,2016,2017 and all years combined.

1)For 2015:

Features are : Economy (GDP per Capita), Family, Health (Life Expectancy), Freedom, Trust (Government Corruption), Generosity, Dystopia Residual

Mean squared error MLP: 0.18

MLP Parameters :

Hidden Layer Nodes	100
Max Iterations	1000
alpha	0.0001
Activation Function	tanh
Solver	sgd

2) For 2016:

Features are : Economy (GDP per Capita), Family, Health (Life Expectancy), Freedom, Trust (Government Corruption), Generosity, Dystopia Residual

Mean squared error MLP: 1.12

MLP Parameters :

Hidden Layer Nodes	100
Max Iterations	1000
alpha	0.0001
Activation Function	tanh
Solver	sgd

3) For 2016:

Features are : Economy (GDP per Capita), Family, Health (Life Expectancy), Freedom, Trust (Government Corruption), Generosity, Dystopia Residual

Mean squared error MLP: 0.19

MLP Parameters :

Hidden Layer Nodes	100
Max Iterations	1000
alpha	0.0001
Activation Function	tanh
Solver	sgd

4) For all 3 years combined:

Features are : Economy (GDP per Capita), Family, Health (Life Expectancy), Freedom, Trust (Government Corruption), Generosity, Dystopia Residual

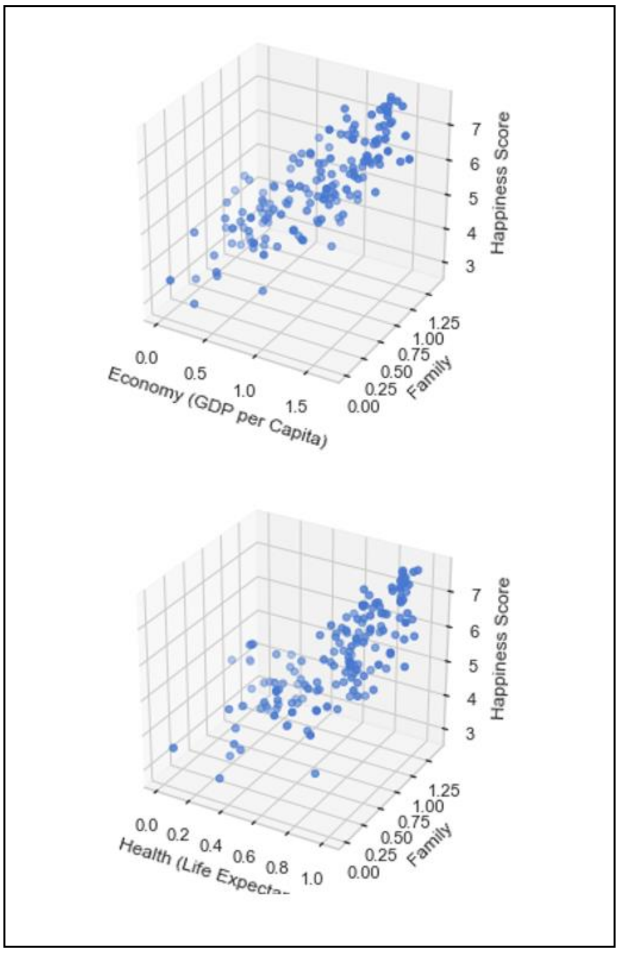
Mean squared error MLP: 0.16

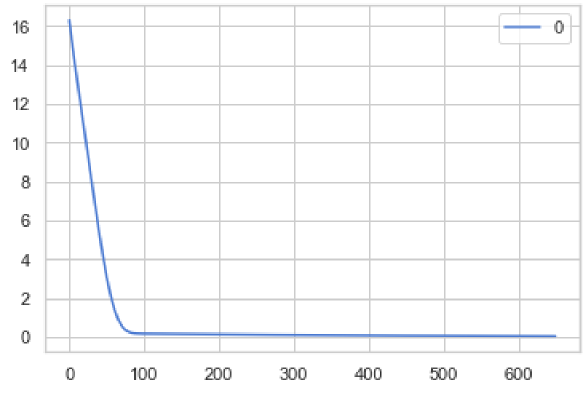
MLP Parameters :

Hidden Layer Nodes	100
Max Iterations	1000
alpha	0.0001
Activation Function	identity
Solver	sgd

Plots and Analysis

We have plotted the happiness score vs 2 feature parameters to visualize and cross check whether our results make sense. As predicted, we see that the Economy and Family features are the ones which fit a line best.





Loss Function decreasing with equation

A. *Discussion and Conclusion*

We were successful in training a Multilayer regression model to fit the given data set of happiness index for different countries for 3 different years. We have been able to match the results obtained by the MLR model to the one obtained by Linear regression model successfully. We achieved the best results when we combined the 3 datasets to form one unified dataset to train our MLP model. Our results improved as we increased the number of layers in our hidden layer. We also observed that different random state values - that which signifies different initialization values gave our different but also in many cases improved results.

Acknowledgment

The authors of this paper would like to thank Prof. Jennie Si for giving us direction for maximal progress and always being there when we faced difficulties of any sort.

References

[1] World Happiness Report dataset: https://www.kaggle.com/unsdsn/world-happiness/downloads/world-happiness-report.zip/2

[2] https://www.kaggle.com/javadzabihi/happiness-2017-visualization-prediction