Predicting Students Performance on Math Exams

Daniel Janis

April 2020

Contents

1 Introduction **2**

2 Dataset **3**
2.1 Dataset Description 3
2.2 Output Data Plots 4

3 Data Processing **5**
3.1 Normalization of Data 5
3.2 Splitting of Validation and Training Sets 5
3.3 Randomization of Data 6
3.4 Performance Comparison 6
3.5 Model 1 - Single Neuron 7
3.6 Model 2 - Two neurons 8
3.7 Model 3 - Three neurons 9

4 The Chosen Model **10**
4.1 Future Work 11
4.2 Results 11

Chapter 1

Introduction

Test scores in school have always been a good indicator of performance in certain areas of subjects for both students and teachers. Predicting the outcome of how a student may score on certain exams based on factors such as if their parents obtained a higher education, or if the student had test preparation or no test preparation at all. Even factors like if the students meals were free may have had a correlation with the scores which I had hoped to find out with a model. As a student, I am interested in trying to determine if there is a link between these influences on a students performance and those students math exam grades.

Chapter 2

Dataset

The dataset used in this analysis is titled "Students Performance in Exams" which can be found on kaggle.com authored by the user Jakki [1]. This set contains data concerning students test performance on various exams along with statistics including parental education level, lunch availability, race/ethnicity, and test preparation.

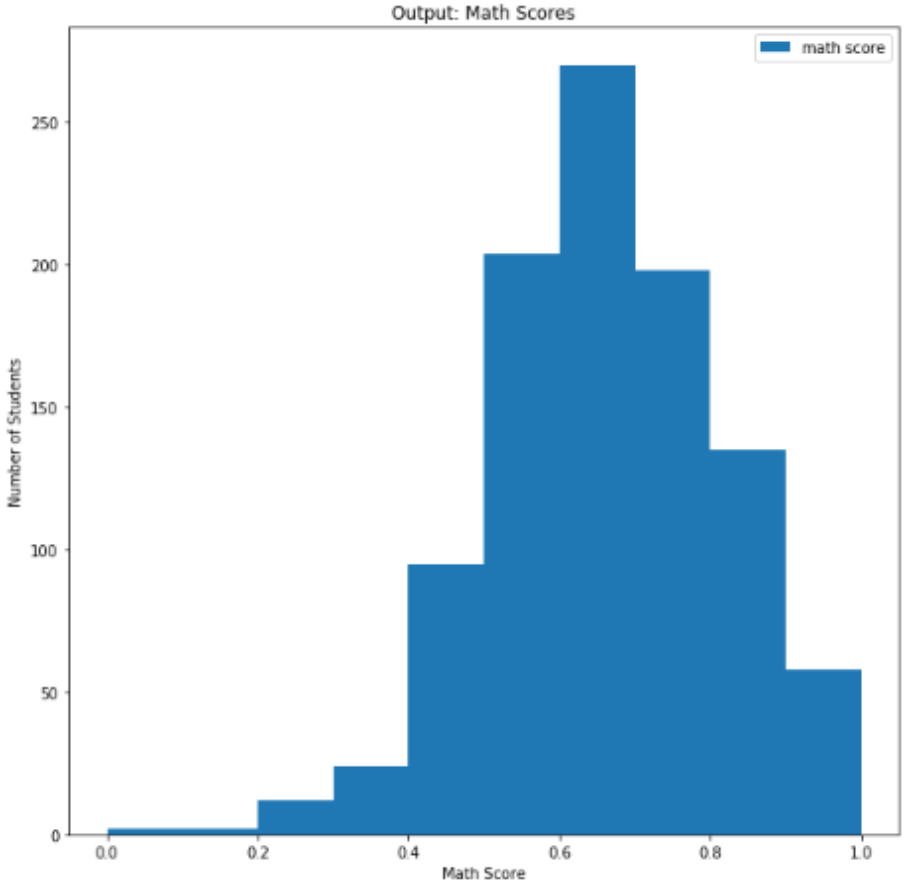
2.1 Dataset Description

In total there are 8 columns and 1000 rows for this dataset. For this project I chose to use columns 1-6, excluding the columns 7 and 8 (the reading and writing scores which are covered on the last page because I was interested). For each student, I have selected the columns regarding gender, race/ethnicity, parental education level, lunch, and test preparation course, along with the math score as the output. The data I am working with will consist of 6 columns and 1000 rows in total (along with the additional extra analysis at the end).
The input columns used are as follows:

- gender
- race/ethnicity
- parental level of education
- lunch
- test preparation course

2.2 Output Data Plots

The output column has been plotted showing its distribution below:



This plot is balanced, as there aren't too many extreme cases (0.0 or 1.0's) and most of the students scores hover around the 0.6 mark, or 60%.

Chapter 3

Data Processing

3.1 Normalization of Data

For this part I had used re-scaling to turn the values of the math score column into an integer between 0 and 1. This was done utilizing the following equation:

```
# Rescaling the Math Score Column
colMathscore = data['math score']
colMathscore = colMathscore / colMathscore.max()
data['math score'] = colMathscore
data.head()
```

Basically, this equation took every score and divided it by 100. The reason this was done is because the maximum math score was 100 and the minimum math score was 0. This meant that I could simply divide each value of the column by the maximum minus the minimum (100 - 0 = 100). The result meant that each of my math scores were now in the range from 0 to 1.

3.2 Splitting of Validation and Training Sets

Validation data is used for training the neural networks presented below. The reason for splitting the data is so that we can train our model and look at the contrast between the training and validation sets.

```
################################################
# Split the data into training/validation sets   #
################################################

# Index for 30%
index_30percent = int(0.3 * len(dataset[:, 0]))
print(index_30percent)
print(int(len(dataset[:, 0])))
# Split into training and validation
XVALID = dataset[:index_30percent, [0, 1, 2, 3, 4]]
YVALID = dataset[:index_30percent, 5]
XTRAIN = dataset[index_30percent:, [0, 1, 2, 3, 4]]
YTRAIN = dataset[index_30percent:, 5]
```

3.3 Randomization of Data

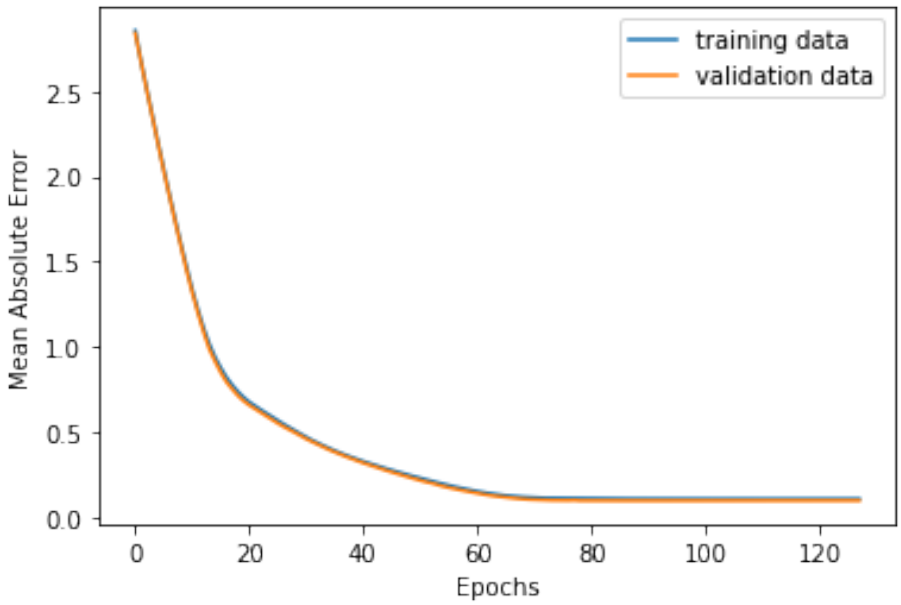
This was a rather simple part to implement, basically it randomizes the order of the rows so that we are getting a unique answer and it helps test the model accurately.

3.4 Performance Comparison

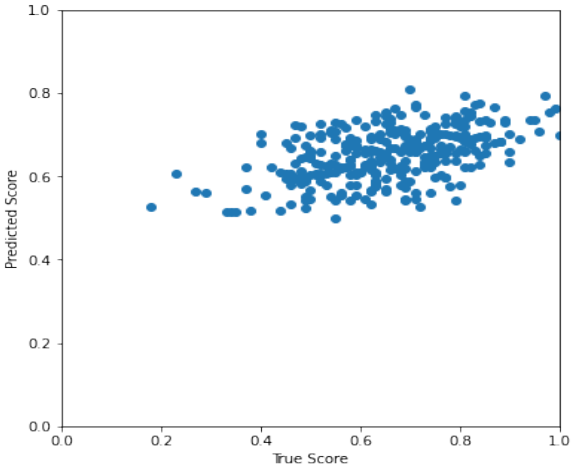
Models will be selected based on how closely the training and validation data compare to each other. The lower the Mean Absolute Error the better. Basically, if we have a low mean absolute error (MAE) than that would mean that there are very few errors (aside from the few outliers).

For the following models I took a look at the Mean Absolute Error differences between each model. The goal is to minimize Mean Absolute Error (MAE) as the model runs through Epoch's while training.

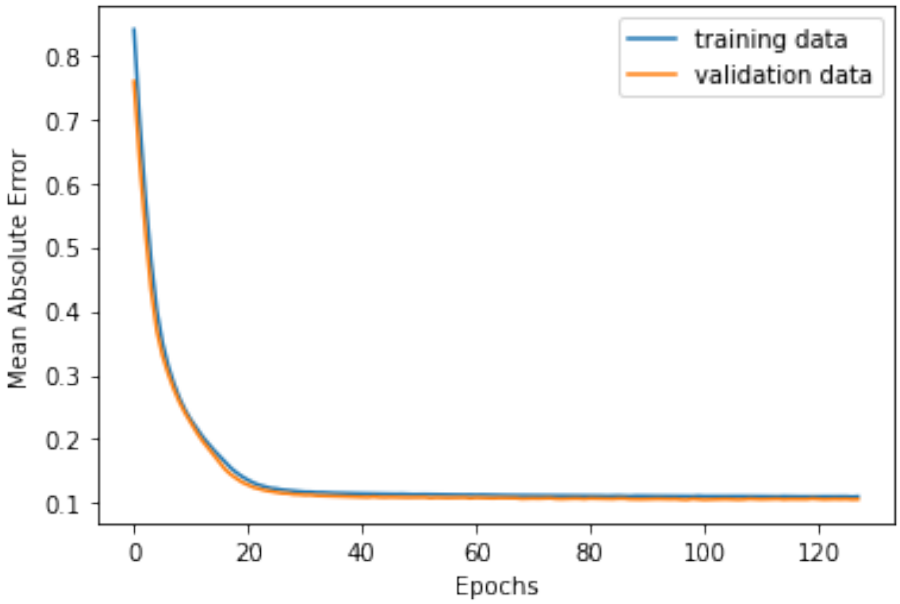
3.5 Model 1 - Single Neuron



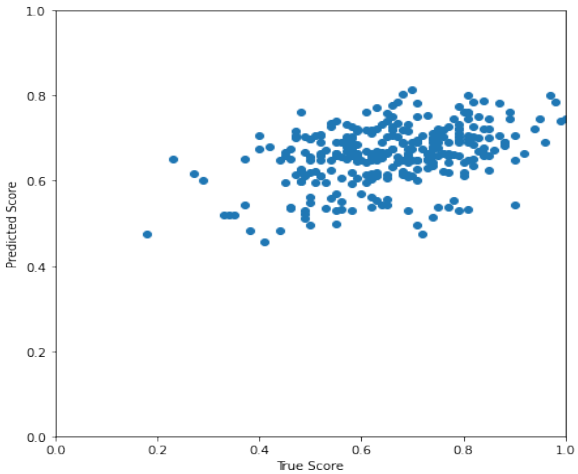
This model is known as a multivariate linear regression model. It involves a single neuron with 5 input columns. The plot above is the Mean Absolute Error vs. Epoch's graph for the model after running for 128 epoch's. What this plot shows is that as the model progresses and passes the inputs through the tensorflow.keras.models/layers functions, it becomes less prone to make errors in its predictions. This model gave me an MAE of 10.211 which is very low considering all of the different models I've tested on this data set. The predicted scores vs. true scores plot is shown below.



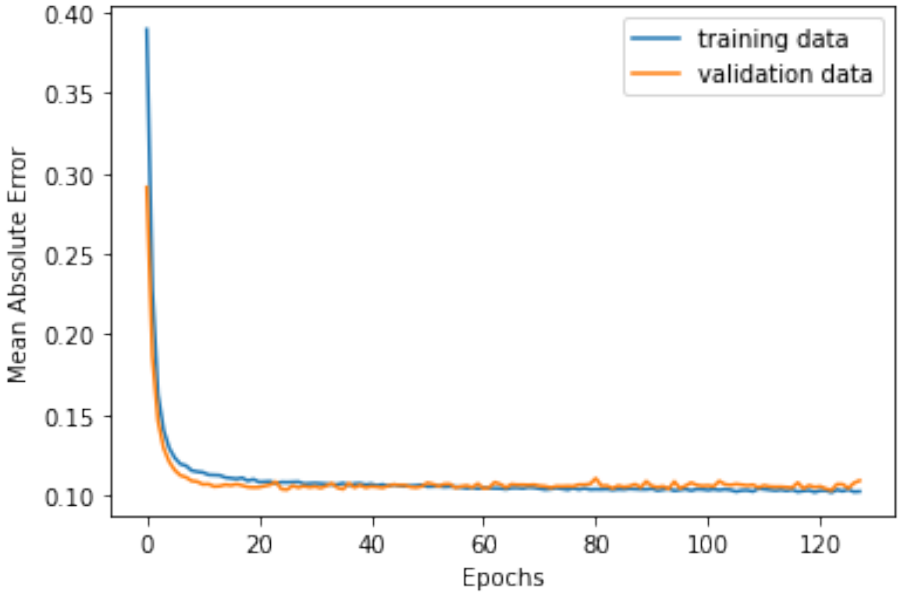
3.6 Model 2 - Two neurons



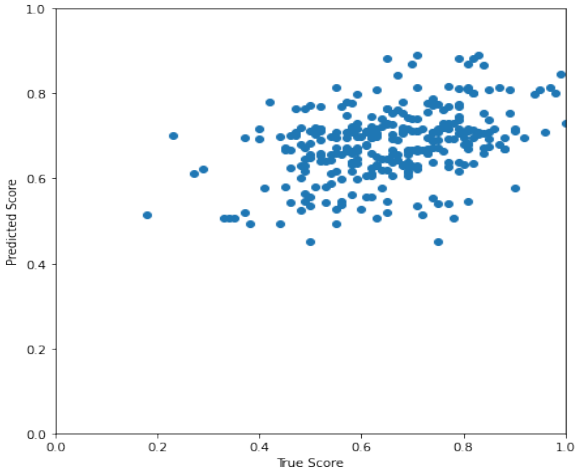
This model is a copy of the single neuron with an added neuron layer that has a density of 7 and the same 5 input columns. After 128 epoch's the graph above shows that the MAE is constantly decreasing because as the model progresses, the line continues to decrease. The MAE for this model was 10.626 which is also a good MAE on average. After running this model many times, 10.6 is around the lowest I could get the MAE (besides the 10.2 above for the single neuron model). Predicted vs. true scores plot is shown below.



3.7 Model 3 - Three neurons



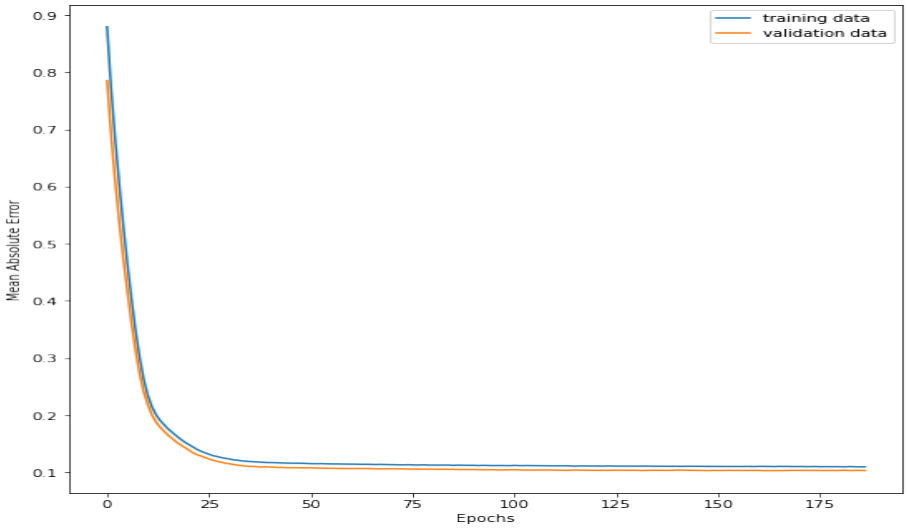
This model is a copy of the two neurons model with an added neuron layer that has a density of 42 and the same 5 input columns. After 128 epoch's the graph above shows that the MAE is worse than the two models above it, with an MAE of 10.875. Along with having a bad MAE, this plot also shows a lot of wiggles which means it was trying to match the data perfectly but actually ends up doing a worse job at predicting values than the two neuron model. The distribution of predicted vs. true scores is also worse and more scattered apart.



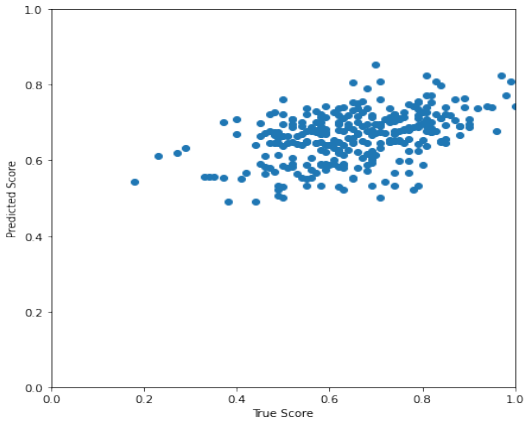
Chapter 4

The Chosen Model

I decided to stick with the Two Neuron Multivariate Linear Regression model that utilized each input column available due to how vast the differences were each time I tried to look at feature importance and reduction. After much trial and error, it seemed that the best model always included all of my inputs to aid in predictions. The models MAE vs. Epoch's plot after early stopping and model check pointing is shown below:



The early stopping allows for the model to stop training once the MAE hasn't decreased within the last 25 epoch's meaning that it trains until it determines that it will not benefit in training further. This plot is constantly decreasing, and it shows that the MAE on average was 10.314 which tells me that the predictions this model could make are within a single Grade level. For example, if a student had scored a 70/100 on the math exam, this model should be able to predict that score within the ballpark of around +/- 10.314 (give or take the few outliers).



4.1 Future Work

If I were to add to this model I would likely change the math scores to a pass/fail system rather than a grade from 0-100 which would allow me to use a binary classification model. I believe this would be able to increase the accuracy of predicting if a student will pass or fail a math exam based on the given 5 inputs. I would also try to predict the other scores (reading and writing) if I had more time to focus on all of these differences. Gathering more data from different sources may be a way I could get around the issues with being unable to shrink the mean absolute errors I kept getting during my modeling of this small data set. Basically, the more columns and rows of information relevant to students and their grades, the better.

4.2 Results

I believe that my model somewhat succeeded at what I was trying to use it for. It is not good but it is not ridiculously bad. Basically, this model works better than a coin flip given the data set that I used in predicting a students grade based on the 5 input columns that were provided. In my eyes it is a success in that now I know that it is a lot harder than I had anticipated to predict a student's grade on an exam given such a small amount of data/inputs. The final model that I chose ended up being close enough to what I was attempting to determine at the start of this project: is there a link between a students performance on a math exam and the input column influences described earlier in this report. It turns out that yes, it may be possible to get close but as of now this data set is too small and there are too few columns to predict the math scores with high precision. It is a fun model and shows that it is making predictions that are better than random chance but I feel like I could make a better model still.

Bibliography

[1] Jakki. *Students Performance in Exams*. Mar. 2019. URL: https://www.kaggle.com/spscientist/students-performance-in-exams. (accessed 2/23/2020).