Predicting Wine Quality Using a Neural Network

Katarina Sperduto
Department of Computer Science
Florida Southern College
ksperduto@mocs.flsouthern.edu

***Abstract*— Machine Learning is a fast growing area of study. One of the main algorithms used in this area of study is the Neural Network. A neural network is a supervised machine learning algorithm which allows a computer to "think". This paper goes into the design of a neural network used to predict the quality of wine.**

I. INTRODUCTION

Neural Networks are one of the most popular machine learning algorithms used today. These models are inspired by the human brain and are used to solve complex problems. Each network design is based on the creator of the network and what information is being passed through it. Each network goes through a training and a testing phase. Being a supervised learning algorithm, a dataset with both data and labels is required. This study was focused on the creation of a neural network for a complex dataset.

II. THE PROJECT

A. The Data

For this project, I wanted to create a neural network to predict the quality of wine. I reached for a dataset high contained a good amount of interesting elements when deciding how a bottle of wine tastes. The dataset used for this project was downloaded from the website Kaggle. This set included the following for each of the samples: type, fixed acidity, volatile acidity, citric acid, residual sugar, chlorides, free sulfur dioxide, total sulfur dioxide, density, pH, sulphates, alcohol, and quality.

For this project I used python. The python file reads in the dataset then separates it into two categories, the data and the labels. In this case the data excluded the type column, purely because they were strings, and the quality column, because this column is used as the labels.

B. The Neural Network

The neural network is a sequential model with three hidden layers, eleven input elements and seven possible outputs. In other words I have a 11-100-100-100-7 neural network.

The input layer takes in the eleven elements for each test sample, both the input and hidden layers use a relu activation function, and the output layer classifies the data into one of seven categories using the sigmoid activation function. The model was compiled using the mean squared error as the loss value, the Adam function as the optimizer, the accuracy as the metric, a batch size of 50, 100 epochs, and a verbose of 2. The reason why the Adam optimizer was used instead of a set learning rate was due to the range of data. Since the data had such a large range of points, I normalized the data. This allowed the network to more easily compare features of that data which were common in a certain quality of wine. After the model was trained, I ran it on the test data, and keep track of the predictions made by the network. This information is then placed into a confusion matrix and printed to the screen

C. Training and Testing

I used the same dataset to train and test the network. I originally had plans to separate the data by type and use different data to train and test each type however, because there was such a large variety among the dataset where some categories had five sample and some have over a thousand, it became difficult to separate the data . In addition to the data not separating the way I wanted, the overall performance of the neural network was not the best. The overall accuracy of the network was about 55.3% with a loss of about 8.1%. The confusion matrix was as follows:

```
[[   0    2    0   16   11    1    0]
 [   0    0    9  129   69    7    0]
 [   0    0    2  991 1095   40    0]
 [   0    0    3  367 2228  222    0]
 [   0    0    0   22  706  346    0]
 [   0    0    0    8  107   77    0]
 [   0    0    0    0    3    2    0]]
```

Figure 1: Confusion Matrix

Out of 6,464 samples, only 1,152 were predicted correctly. After getting these results I looked further into the data set to try and figure out the reason why the network did so poorly. What this research resulted in was very interesting. In fact the network wasn't actually doing that bad, the main issue was the data.

III. RESULTS

A. *Issues with the Data*

I examined the dataset by graphing specific elements compared to the quality. Below is a graph with the acidity on the y-axis and quality on the x-axis. Fixed acidity values are the blue points, volatile acidity values are the green points, and the citric acid values are the orange points.

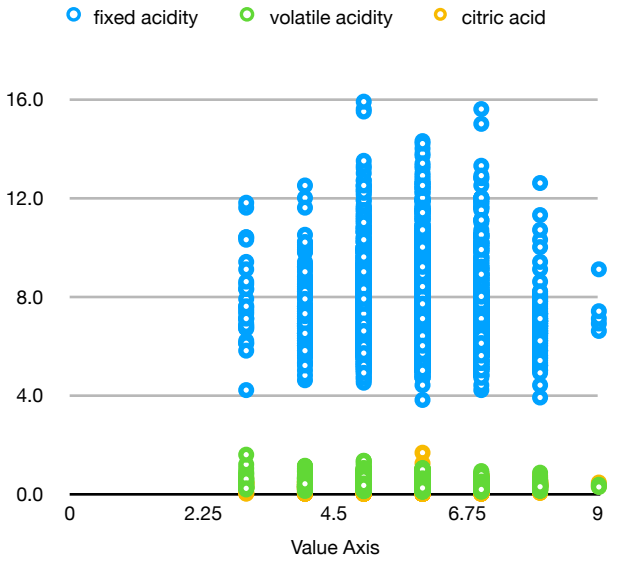

Figure 2: Acidity vs Quality Graph

As shown in the graph, this data is very difficult for a neural network to read because of the variety in each of the quality categories. There is not really any clusters where the network is able to separate values and claim them unique to a certain quality.

B. *Overall Results*

Since this data is so difficult fro the network to predict, I decided to do another test where I added an additional hidden layer and increased the batch size and amount of epochs. I re-ran the data through this 11-100-100-100-100-7 neural network with a batch size of 100 and 2000 epochs. The results are displayed the the confusion matrix below:

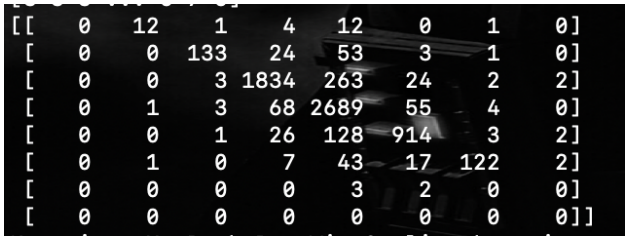

Figure 3: New Confusion Matrix

The accuracy increased to about 88% with loss decreasing to about 2.8%.

C. *Conclusion*

The overall conclusion I came to was that increasing the epochs and batch size led to a better overall accuracy. I feel if I was to increase the epochs even more, the network would do even better. This would be interesting to try in the future.