Miniproject 2

Group 3

Ashley Iannuzzelli

Joshua Ayers

Javier Mendez

Andy Senda

**Problem 1)**

**Neural Net to classify Avila Data**

**a) How many parameters are estimated in this network (including constants)?**

Total number of parameters is **4192**

For supporting work refer to *Attachment I - Project 2 - Non Software.xlsx,* tab titled *Question 1.*

**b) Calculate the value at the proposed node after the sigmoid function is applied.**

The value at the node is **0.690516**.

For supporting work refer to *Attachment I - Project 2 - Non Software.xlsx,* tab titled *Question 1.*

**c) Calculate the outputs after a softmax function is applied and estimate class probabilities for instance provided.**

| **Class** | **Softmax Output / Class Probability** |
| --- | --- |
| A | **0.05866384015** |
| B | **0.1069702251** |
| C | **0.1249018871** |
| D | **0.06068706183** |
| E | **0.0514246961** |
| F | **0.02683482612** |
| G | **0.05956061455** |
| H | **0.1141222824** |
| I | **0.1010444507** |
| W | **0.07818372658** |
| X | **0.1246021082** |
| Y | **0.09300428115** |

For supporting work refer to *Attachment I - Project 2 - Non Software.xlsx,* tab titled *Question 1.*

**d) For the instance in part (c), what class is assigned?**

The class assigned is **C**, which is the max output of the softmax, **0.12490**

For supporting work refer to *Attachment I - Project 2 - Non Software.xlsx,* tab titled *Question 1.*

**e) For the instance in part (c), suppose that the actual target value for this instance is Class A. Calculate the contribution to the cross entropy loss function from this instance.**

| **Softmax** | **ln(A)** | **Entropy loss** |
| --- | --- | --- |
| 0.0586638 | 2.835931753 | **2.835931753** |

For supporting work refer to *Attachment I - Project 2 - Non Software.xlsx,* tab titled *Question 1.*

**Problem 2)**

**Consider a neural net to classify 128 x 128 for each network below. Calculate the number of parameters (including constants).**

**a) A fully connected network is used with 3 hidden layers and 50, 40, 30 nodes in these layers.**

The total number of parameters is: **822830**

For supporting work refer to *Attachment I - Project 2 - Non Software.xlsx,* tab titled *Question 2.*

**b) A network with local connectivity (but not weight sharing), 2 hidden layers, 1 8x8 filter between input and hidden layers...**

The total number of parameters is: **65175**

For supporting work refer to *Attachment I - Project 2 - Non Software.xlsx,* tab titled *Question 2.*

**c) A network with local connectivity, and weight sharing (including a shared constant weight) (convolutional network)...**

The total number of parameters is: **500**

For supporting work refer to *Attachment I - Project 2 - Non Software.xlsx,* tab titled *Question 2.*

**d) A network with local connectivity, and weight sharing (including the constant weight) (convolutional network)...**

The total number of parameters is: **32346**

For supporting work refer to *Attachment I - Project 2 - Non Software.xlsx,* tab titled *Question 2.*

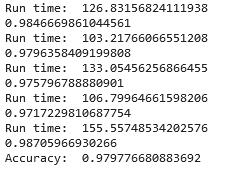
**Problem 3)**

**Software implementation of neural net for the Avila dataset.**

**Submit your final code, your output, and your estimate of generalization error.**

For final code refer to **Attachment 2 - Project 2.py***.*

The output using 5-fold cross validation had an average accuracy of **0.97978**. Details for each fold to follow.



The next step would be to train this model on the entire data set and release that model to production.

**Submit a brief description of what models you explored and how you estimated generalization error.**

The models explored included several nets ranging from 1 to 5 layers and 40 to 400 nodes. Initially, the models were developed using a randomized 80/20 data split, and the final model was validated using cross validation.

To determine the optimal model prior to running cross validation, a series of DOE type analysis was performed on the 80/20 data. The output of interest was accuracy, but run time was also taken into consideration.

First Experiment

|  |  |
| --- | --- |

The result showed lack of linearity based on the center point (red dot on left chart). The optimization plot (right chart) showed that the center point was an optimal solution.

To confirm this observation a two factor three four level experiment was completed. The motivation was to understand the curvature and assess points between the original corner point and the center point.

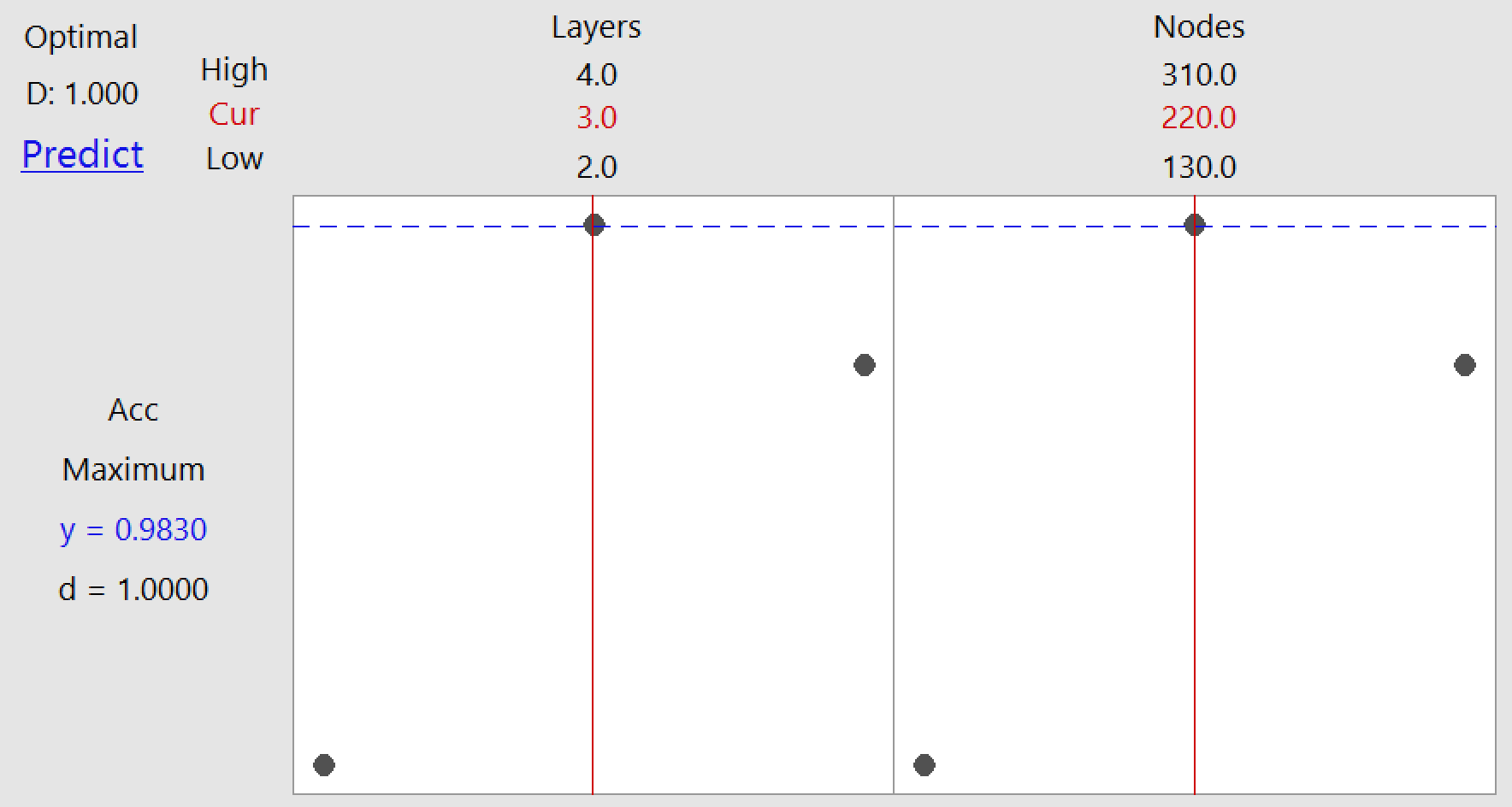
Second Experiment

|  |  |
| --- | --- |

The result showed significant gains from 1 to 2 layers and from 2 to 4 layers, but not from 4 to 5 layers. In this experiment 4 layers weakly dominated 5 layers when also taking run time into account. The experiment showed that 400 nodes dominated all other nodes, but if the analysis accounts for the interaction of nodes and layers, 400 nodes do not seem to provide a significant advantage past 2 layers.

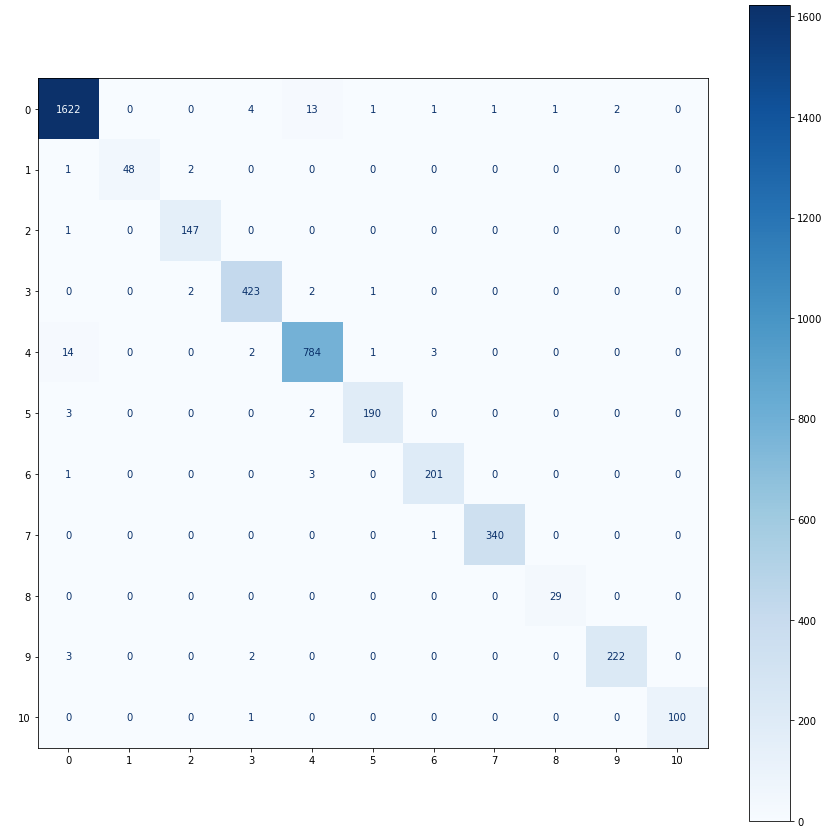
A final experiment was run to compare 4 layers, 400 nodes with the original mid-point of 3 layers, 220 nodes.

Third Experiment



The result showed that 3 layers, 220 nodes continued to be an optimal point based on accurrancy and particularly considering that it also has a much lower run time.

The confusion matrix for this model is provided below.



**Submit a clear statement of architecture and any parameters changed from the SKLEARN defaults.**

The architecture that provided the best model was a neural network that has 3 hidden layers each with 220 nodes. The number of hidden layers and nodes was changed from the default SKLEARN parameters of 1 hidden layer and 100 nodes.

**Attachments**

*Attachment I - Project 2 - Non Software.xlsx*

*Attachment 2 - Project 2.py*