Lab 6 – MLPs

MACHINE LEARNING

SAUMAY AGRAWAL
16BCE1151

EXPERIMENT

- Try the MLP algorithm for your dataset.
- Plot the training error over the number of epochs.
- Explore the possibilities of Mulitple Neurons/node, multiple layers, multiple output nodes depending on the problem and report the results in tabular form. Write your inference for your dataset.
- If possible, try an almost 'unclassifiable' dataset to see what are the challenges for this MLP Algorithm

ALGORITHM

In an earlier lab exercise, we have worked with perceptron units. At that time, we found out that a perceptron is capable of creating a single decision boundary. Because of this it was only limited to the cases when the data is linearly separable. However, in most cases, the data we get is not linearly separable and the decision making expected is far more complex. Thus, a network of perceptron units is used to achieve a combination of multiple decision boundaries, which helps us in achieving more complex decision making models. This network is called Multi-layer perceptrons (MLP).

The network has multiple layers of various perceptron units. Among these, there is one input layer, at least one hidden layer, and an output layer. Generally, input layer contains nodes equal to the number of input features and output layer contains the number of nodes equal to the number of unique class labels in the target vector. There could be arbitrary number of hidden layers, each containing arbitrary number of nodes. Each node is fully connected to every node of the previous and next layers, but there is no connection between the nodes of the same layer.

CONFUSION MATRIX

A confusion matrix is nothing but a mathematical representation of the comparison of predicted vs actual output values. It is a square matrix of dimension equal to the number of class labels in the target feature. The element C(i,j), denotes the number of transactions

having class label i, classified with class label j by our model. So C(i,j) where i=j, ie diagonal elements of matrix, denotes the number of right classifications. Meanwhile, other elements denote the number of wrong classifications by our model.

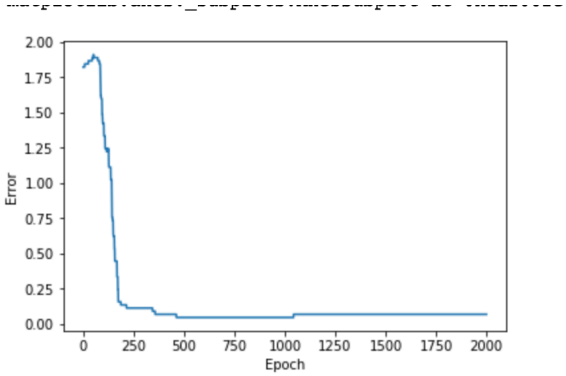
CLASSIFICATION REPORT

It showcases the various evaluation metrics like precision, recall, and f1-score for all the class labels present in the target feature.

OBSERVATIONS

IRIS DATASET

- All the variables like Petal length, Petal width, Sepal length and sepal width, were taken as input parameters. The 'species' attribute was set as the target feature.
- On setting a single hidden layer having 8 nodes, I was able to achieve an accuracy of 1.00 right away.
- The algorithm with same layer structure, didn't converge until 1157 iterations. The graph of MSE vs the number of iterations is given below for this dataset.

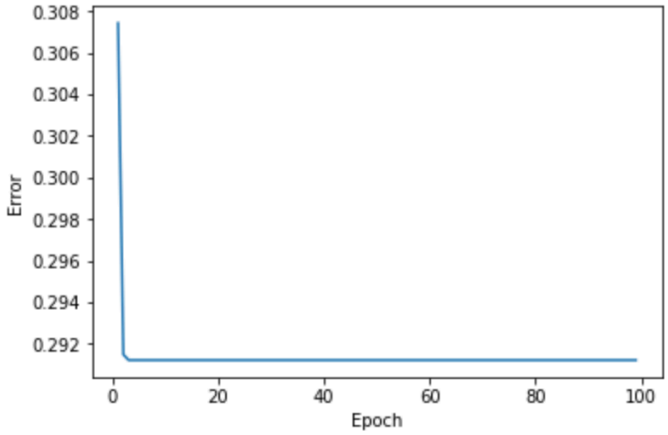


- The results for various layer structures are tabulated below.

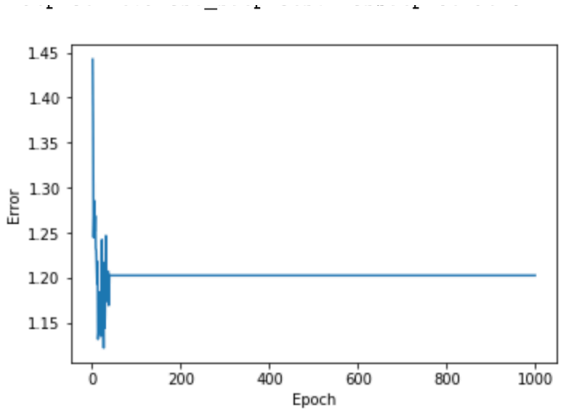
OLYMPICS DATASET

- Only relevant attributes like Height, Weight etc were kept as input features, and others like 'Name', 'ID' were discarded. The categorical values were then label encoded, and scaled for better performance. The 'medals' attribute was set as a target feature.

- By setting a single hidden layer of 5 nodes, I was able to achieve an accuracy of roughly 85%. However, the confusion matrix shows that almost all of the transactions were classified to ‘none medal’ class label.
- Thus it was found that, high accuracy was a result of fact that the number of transactions with label ‘none medal’ exceeds far more than the others combined.
- As a result, the MSE vs number of iteration graphs showed immediate convergence for a dataset that is so unclassifiable.



- To get around this, I removed all the entries having medal attribute as ‘none’, and repeated the same steps on the dataset. The accuracy of model dropped to 38.3%. The graph below is obtained as a result for MSE error vs epochs.



- The results obtained by altering the number of hidden layers and number of nodes per layer are tabulated below.

[HL – Number of hidden layers, NPL – neurons per layer]

ACCURACY COMPARISON TABLE

(HL, NPL)	Iris data	Olympic (20 years)	Olympic (40 years)	Olympic (120 years)
LOW, LOW	0.933	0.490	0.457	0.438
LOW, HIGH	0.933	0.534	0.502	0.519
HIGH, LOW	0.956	0.397	0.395	0.394
HIGH, HIGH	0.933	0.545	0.330	0.522

COMPUTATION TIME COMPARISON TABLE

(HL, NPL)	Iris data	Olympic (20 years)	Olympic (40 years)	Olympic (120 years)
LOW, LOW	347ms	6.51s	11.8s	26.1s
LOW, HIGH	264ms	1m 16s	21.1s	9m 29s
HIGH, LOW	723ms	2.55s	3.97s	11.7s
HIGH, HIGH	327ms	12m 43s	3m 29s	1h 14m

INFERENCE

- MLP algorithm is good for smaller datasets, but as the dataset grows, it becomes inappropriate in terms of computational power required.
- Computation time and performance are hardly affected by change in layer structure in smaller datasets.
- For larger datasets, low number of hidden layers and high number of nodes per layer is a good way to achieve balance between computation time spent and performance achieved.

LIMITATIONS/CONS OF MLP

- It is time consuming to get the right number of layers and nodes, suitable for dataset being used. It takes a lot of experimenting and intuition.
- It might be difficult to find a balance between overfitting and limit of classification of data, as we tweak the layers in order to increase the accuracy.
- The algorithm doesn't converge permanently at global minimum after a particular number of epochs. Instead it starts to diverge, thereby reducing the accuracy.
- Since there are so many connections between each and every node of the network, the algorithm is very intensive computationally.