CENG414-Introduction to Data Mining

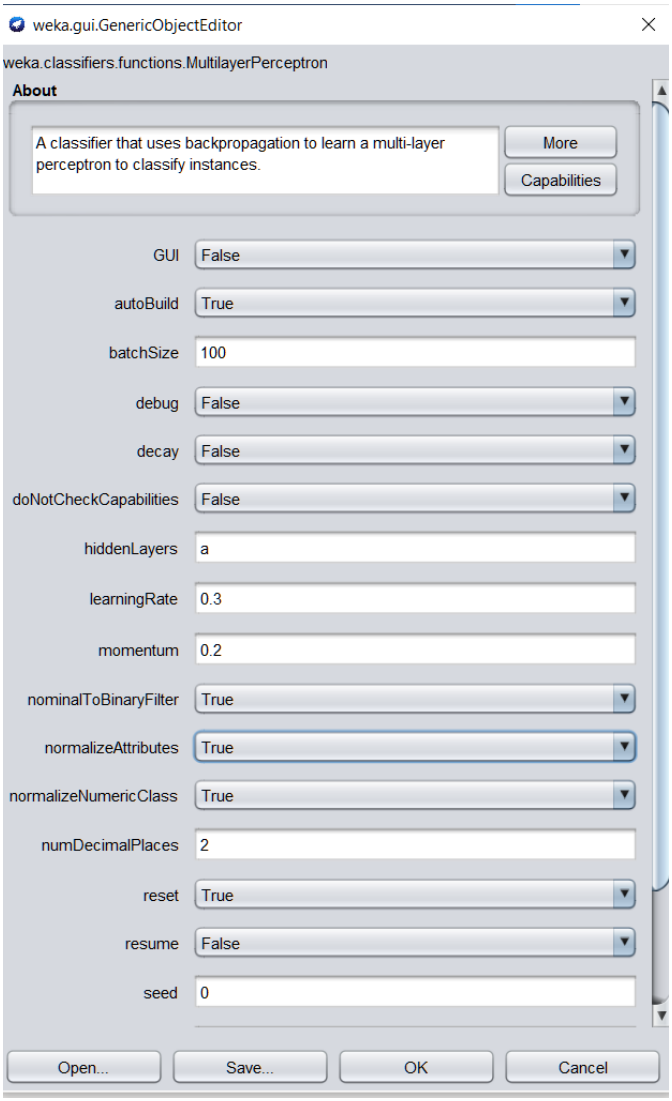
Programming HW1

Task 1-Multilayer Perceptron

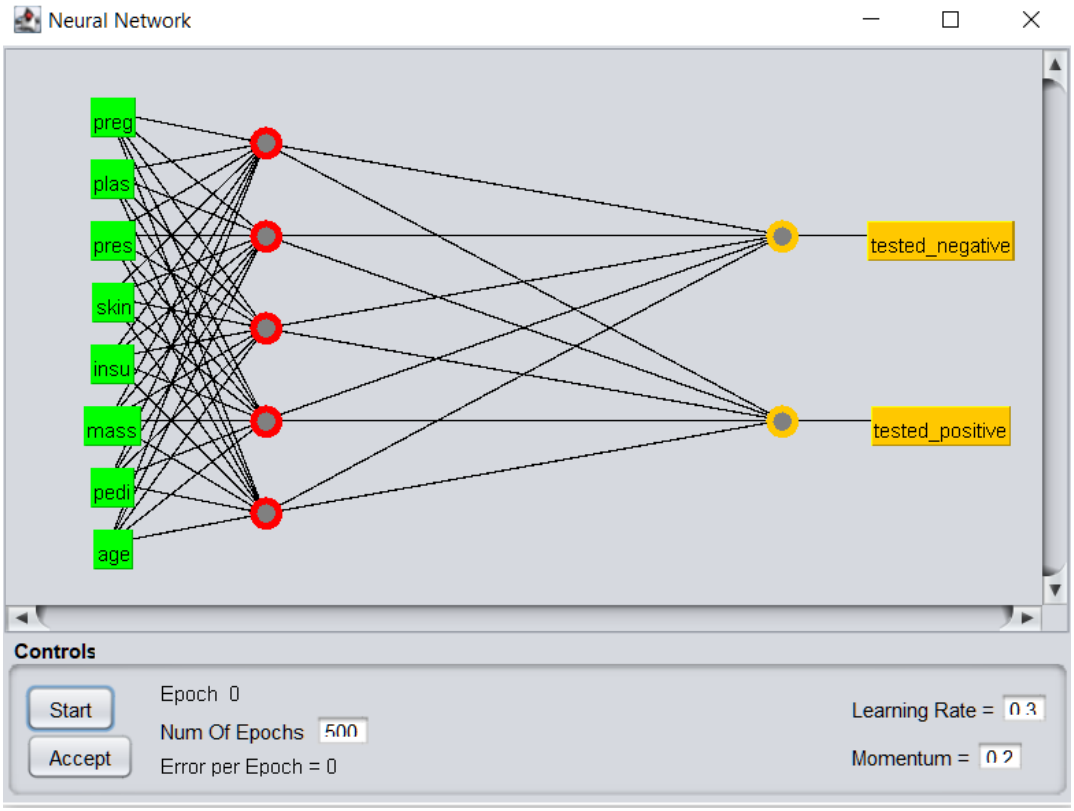
Zeynepsu Kesim

2292340

Default Parameters:



1-) How many hidden layers and hidden nodes created?



As we can see, there is one hidden layer with 5 hidden nodes (the red ones) → mostly Weka decides it by taking the mean of input and output nodes which is (8+2)/2 = 5 as we have.

Moreover, there are one input layer with 8 nodes: for each attribute there is one node (green ones)

One output layer with 2 nodes: for each class there is one node (yellow ones)

2-) Did Weka normalize the attributes? What is the effect of normalizing the attributes?

Yes, it did normalize the attributes since the default parameter for 'normalizeAttributes=True'. Normalization is useful because we have a dataset with attributes in different scales. Therefore, for each attribute min, max and more importantly the mean and standard deviation values would be in different scales, and thus hard to compare. Moreover, it will lead the model to be poor and less accurate. Some attributes would unnecessarily dominate the model because of the difference in the scales. When we normalize the data, thus the attributes, we would have the data in the same scales. All the means and standard deviations would be rescaled without distorting differences in the ranges of values, and comparable; thus, the model would be more effective and accurate. When we compare two MLP models one with normalized attributes and

the other one with no normalization, we would see that the model with normalized attributes have a higher accuracy rate. When normalizeAttributes=True, accuracy rate is about 75%. When normalizeAttributes=False, accuracy rate is about 68% in the given dataset.

3-) What is the benefit of splitting the dataset as training set and test set? Why don't we just train our model with whole data?

We split the dataset because we want to test the model with values that the model has never seen before. So that we can understand the true accuracy of the model in a better way. Therefore, we use the training data to train the model and test the model with new values, which is the test set.

4-) Which halting strategy did MLP use?

Early stopping-stop when reach the value of max epoch.

5-) What is the detailed accuracy table by class of the run?

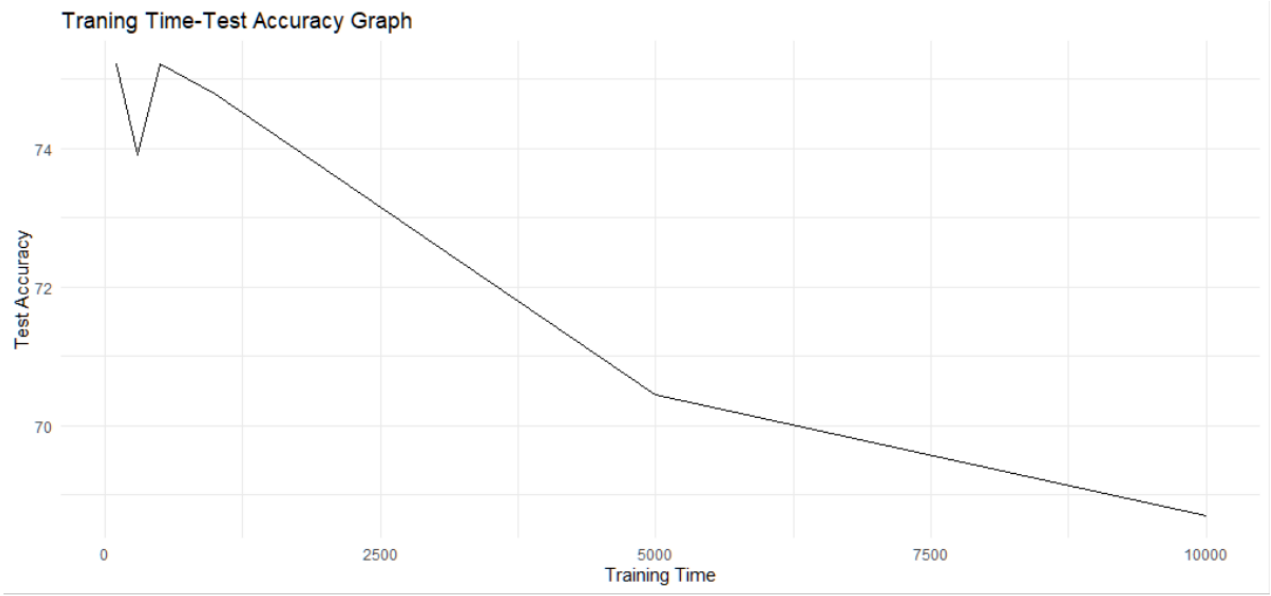
```
=== Detailed Accuracy By Class ===

                 TP Rate   FP Rate   Precision   Recall   F-Measure   MCC      ROC Area   PRC Area   Class
                 0,804     0,361     0,830       0,804    0,817       0,435    0,772      0,859      tested_negative
                 0,639     0,196     0,597       0,639    0,617       0,435    0,772      0,680      tested_positive
Weighted Avg.    0,752     0,309     0,757       0,752    0,754       0,435    0,772      0,803
```

Now change the configurations of the MLP by clicking on the name of the classifier. Run the classification task with different training times (100, 300, 500, 1000, 3000, 5000, 10000) while keeping other variables same and plot the training time-test accuracy plot. Interpret the accuracy results as; what is the relation between accuracy and training time (epoch count)? What may cause this situation?

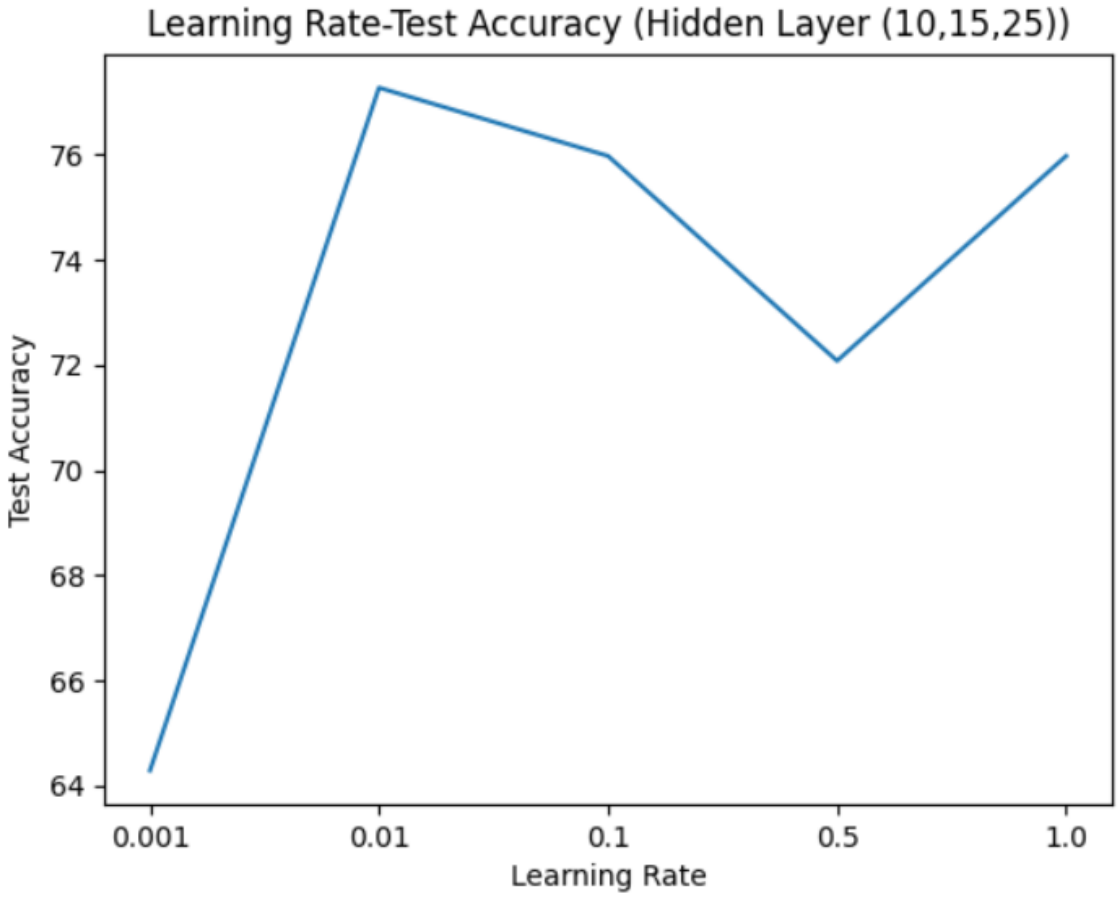
Training Times	Test-Accuracy
100	75.2174 %
300	73.913 %
500	75.2174 %
1000	74.7826 %
3000	72.6087 %
5000	70.4348 %
10000	68.6957 %

The test accuracy increases at some point, but then started to decrease, after the training time 500.

After training time 500, the model starts to overfit the data, and does not generalize well to independent test set. Therefore, as the training accuracy increases while training time increases (when we use the overall training set to test the model). Since the model overfits the differences, noise and even the outliers in the training data (as training time increases), the test accuracy (test set independent of the training set) would be lower and lower because now the model cannot interpret well the values that it has not seen before.



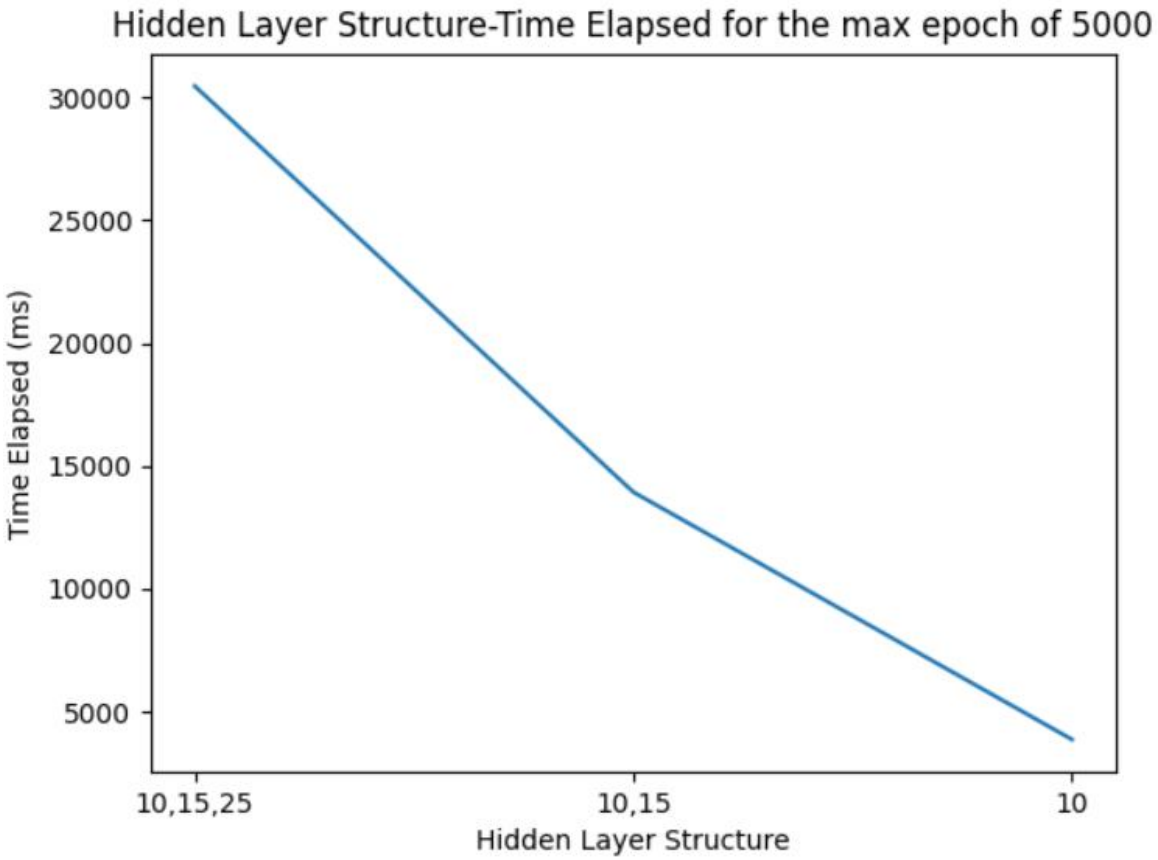
Plot the learning rate-accuracy plot for the hidden layer structure (10,15,25). Comment on the trend of the accuracy; what might result in this?



It can be seen that from the above graph, the best accuracy (about 78%) we have is at learning rate = 0.01. Beside this one, we can see that accuracy rate fluctuates. At learning rate = 0.001, since the learning follows a really slow pattern, the model cannot converge to the optimal at the given epoch value. Therefore, we observe a relatively slow accuracy rate. When the learning rate = 1, learning follows a faster pattern, and thus, while it is trying to converge, it is possible that it can miss some local minimas. For learning rate = 1, as it gets close to the convergence value, say 70%, it might oscillate. At learning rate = 0.01, the model can be trained in a more regular way, and thus it gradually increase and reached to 75%.

Plot the hidden layer structure-time elapsed plot for the max epoch of 5000. Comment on the time elapsed and how it is changed by hidden layer structure.

For learning rate = 0.01 because it is the optimal learning rate.



As the hidden layers increase (and thus nodes in the model increase too), the model becomes more and more completed. As the model becomes more complicated, the time spending on training increases. For each learning rate, the relation is the same as above one. For all learning rates, it follows the same pattern.