**Results report for Mini-Project Number 1.**

**Made by** Ariel Yechezkel(211356449), Almog Amiga(211543285) and Dvir Sadon(322438938)

**Introduction:**

In this report we will elaborate on the code we wrote in the previous questions, about the codes' results and outcomes. Also, we will talk about the process of writing and using the algorithms while providing visual support like graphs and charts to better explain our work and its results.

In the assignment we were asked to implement the following algorithms:

Adaline, which is a single layer neural network where nodes have multiple inputs but only one output.

Backpropagation, which computes the gradient in weight space with respect to a lost function for feedforward neural networks (only).

The algorithms above are used on a dataset of breast cancer cases in Wisconsin U.S. (1).

When encountered with missing data we decided to remove the corresponding cases from the dataset. We decided that because even though it's not very scientific, It’s the best and easiest (Least complicated and most efficient runtime and implantation time wise) way that we found to solve the problem of missing data. And because the cases where data was missing were very rare, we felt that we can afford to remove the cases without damaging the outcome too badly.

**First Question - Adaline algorithm:**

When dividing the data to training and testing, we selected 66% of the data to be used for training the model and 33% of it to be used for testing the model.

1. The time it took to train the model in seconds: 3.7

and in minutes: 0.05

The time it took for the entire program to execute in seconds:4

(These measurements are the time it took after all cross-validation processes.Also, this was measured on my laptop, and may depending on the computer).

1. Early on we came to the realization that the learning rate variable is very much connected to the ecops variable when it comes to the result of the algorithm. And we still think that the learning rate shouldn’t change by a lot and the ecops should. And so, the ecops and learning rate are easy to change but hard to optimize and get exactly right.

The architecture is not very easy to alter, but we do believe that it is possible and not extremely hard to do.

1. The performance of the model is partly detailed in chart (1) which shows the confusion matrix for the algorithm. We were able to achieve an average accuracy of 69%.
2. We think that the dataset was lacking enough data for the model to learn from. If the dataset had more data, we believe that the results would improve.

We had some problems with optimizing the ecops variable, which might have created a few inaccuracies.

**Second Question - Backpropagationalgorithm:**

When dividing the data to training and testing, we selected 66% of the data to be used for training the model and 33% of it to be used for testing the model.

1. The time it took to train the model in seconds: 3.75

and in minutes: 0.05

The time it took for the entire program to execute in seconds: 4

(These measurements are the time it took after all cross-validation processes. Also, this was measured on my laptop, and may depending on the computer).

1. A lot like we mentioned in the Adaline section, we think that it wouldn’t be very easy but manageable to change the architecture of the model.

We also think that it would be very easy to change all the other variable (Like the learning parameter and ecops).Abit tougher to optimize though.

1. We were able to achieve 73% average success rate. The confusion matrix for this algorithm is displayed in chart (2).
2. We had a few problems. One of them was that we couldn’t harness a very promising looking package which could have saved us a lot of time (Torch).

Also, like mentioned in the Adeline algorithm section, more data from the dataset would have made a difference.

**Summary:**

To sum up, we implemented the Adaline (69% success, for confusion matrix see chart (1)) and Backpropagation (73% success, for confusion matrix see chart (2)) algorithms and used them on a dataset of breast cancer cases.

We were hoping that the dataset would be bigger. That way we could make better graphs and train a better model.

**Graphs and Charts:**

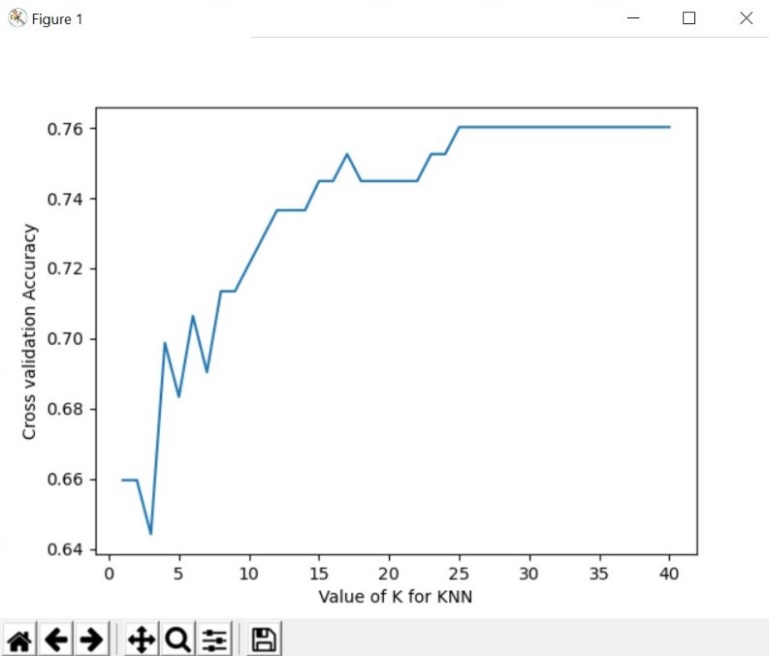
1. Confusion matrix for the evaluation of the Adaline algorithm:

|  |  |  |
| --- | --- | --- |
|  | Predicted: yes | Predicted: no |
| Actual: yes | 0.015625 | 0.0625 |
| Actual: no | 0.25 | 0.671875 |

1. Confusion matrix for the evaluation of the Backpropagation algorithm:

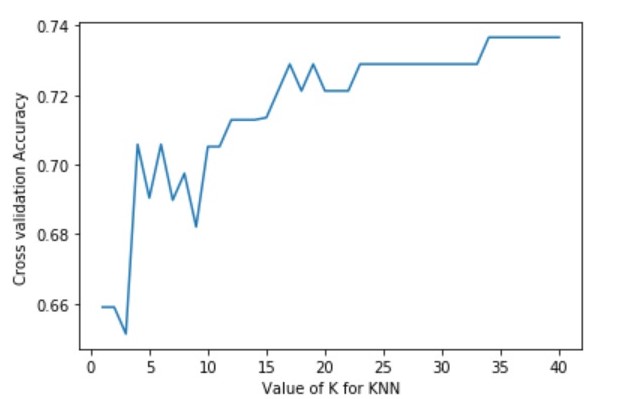
|  |  |  |
| --- | --- | --- |
|  | Predicted: yes | Predicted: no |
| Actual: yes | 0 | 0 |
| Actual: no | 0.265625 | 0.734375 |

1. Adeline cross validation accuracy against value of K for KNN:



We can see that the higher number (up to 40) that we choose, the better the accuracy of the model.

1. Backpropagation cross validation accuracy against value of K for KNN:



Just like the last graph we can see that up to a point (35), we get the best accuracy.

**For additional graphs and visualizations, run the code.**

**References:**

1. The database used in the assignment – Breast cancer cases in Wisconsin U.S:

https://archive.ics.uci.edu/ml/machine-learning-databases/breast-cancer-wisconsin/wpbc.data

1. Explanation regarding the database:

<https://archive.ics.uci.edu/ml/machine-learning-databases/breast-cancer-wisconsin/wpbc.names>

1. Used article for understanding Adeline:

https://machinelearningmastery.com/understand-the-dynamics-of-learning-rate-on-deep-learning-neural-networks/

1. Github of the full project:

<https://github.com/almogun9963/nn1>

**Thank You Very Much for Reading!**