

MSc in Software Engineering and Database Technologies

CT621 Artificial Intelligence

**Student Name:** Cristina Borges

**NUIG ID Number:** 20234955

**Course:** CT621 Artificial Intelligence

**Workshop No:** Week 6

**Assignments:** Week 6

**Date of Submission:** 27/06/2021

Contents

1.0 Assignment 2

1.1 Question 1 3

Forward pass on the network 3

Reverse pass 3

Another forward pass and comment on output 3

2.1 Question 2 5

Algorithm selected 5

Dataset used 5

Accuracy of algorithm 6

Learning curve for algorithm performance 7

Other Comments 7

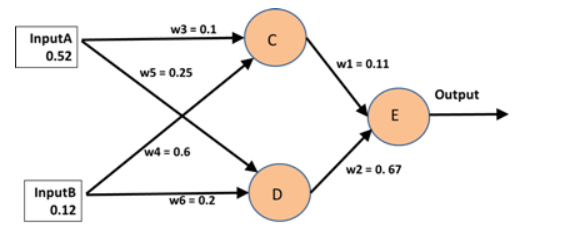
Appendix 1 8

Appendix 2 9

# Assignment

**Question 1.**

Consider the neural network depicted in the image. Assume that all neurons have a Sigmoid activation function. Also, assume that the target output for the network is 0.5.



Apply the Back Propagation algorithm to this example by performing the following steps:

1. Perform a forward pass on the network.
2. Perform a reverse pass once (the target output is 0.7)
3. Perform another forward pass and comment on the new output.

Please show all your workings.

**Question 2.**

At the beginning of Week 6 your facilitator will provide you with a dataset. You should select a Machine Learning algorithm (perhaps the one identified in research) and evaluate the performance of this algorithm (with varying parameters) on the dataset. Students must agree in advance with the facilitator which algorithm they will use. Using the WEKA software package, you should evaluate the performance of the algorithm.

Each student must submit a report that contains at least the following:

1. A description of the chosen algorithm.
2. An analysis of the overall accuracy of the algorithm (including a breakdown of the confusion matrix).
3. A learning curve for the algorithm performance on the chosen dataset.

After all assignments are submitted, the facilitator will compile a short document from them for discussion during Workshop 7.

## 1.1 Question 1

## Forward pass on the network

The general formula can for an input to neuron i is sum of inputs to neuron i x weights for path to neuron i.

Output is calculated as 1/(1 + exp(-input to neuron i))

Input to neuron C = (0.52 x 0.10) x (0.12 x 0.60) = 0.124

Output C = 1 / (1 + exp(-0.124) = 0.5309

Input to neuron D = (0.52 x 0.25) x (0.12 x 0.20) = 0.154

Output D = 1 / (1 + exp(-0.154)) = 0.5384

Input to neuron E = (0.11 x 1 / (1 + exp(-0.124)) + (0.67 x 1 / (1 + exp(-0.154)) = 0.41915

Output E = 1 / (1 + exp(-0.41915)) = 0.60328

Based on the weightings and inputs provided, the output of the neural network is 0.60328 – this compares with the target output 0.7 (I have assumed the 0.5 initially mentioned in the question is a typo).

## Reverse pass

For the reverse pass, we must calculate the output error and then hence adjust the weightings accordingly

Error = (Target output – Output E) x (Output E) x (1 - Output E) =

Error = (0.70 – 0.60328) x (0.60328) x (1 – 0.60328) = 0.023148

wi(new) = wi x (Error x Input)

w1 new = 0.11 + (0.023148 x 0.5309) = 0.12229

w2 new = 0.67 + (0.023148 x 0.5384) = 0.68246

we now step back a level and calculate the new error terms for each input:

Error2 = 0.023148 x 0.12229 x 0.5309 x (1 - 0.5309) = 0.000705

Error3 = 0.023148 x 0.68246 x 0.5384 x (1 – 0.5384) = 0.003926

w3new = 0.10 + 0.000705 x 0.52 = 0.100367

w4new = 0.60 + 0.000705 x 0.12 = 0.600085

w5new = 0.25 + 0.003926 x 0.52 = 0.252042

w6new = 0.20 + 0.003926 x 0.12 = 0.200471

## Another forward pass and comment on output

Input A = 0.52 (as before)

Input B = 0.12 (as before)

Input C = 0.52 x 0.1000367 + 0.12 x 0.600085 = 0.124029

Output C = 1 / (1+ exp(-0.124029)) = 0.53097

Input D = 0.52 x 0.252042 +0.12 x 0.200471 = 0.15512

Output D = 1 / (1 + exp(-0.15512)) = 0.5387

Input E = 0.53097 x 0.12229 + 0.5387 x 0.68246 = 0.43257

Output E = 1 / (1+exp(-0.43257) = 0.6065

New error = (0.7 – 0.6065) x (0.6065) x (1 – 0.6065) = 0.02231

So, the output error reduces.

If we repeat the process of performing a reverse pass using the weights calculated followed by a new forward pass, the Output E would increase to 0.60958 and the error would further reduce to 0.021519. If we continue to perform this process, we would expect the output E to converge with our target output of 0.70 and for the error term to reduce as the output approaches our target.

## 2.1 Question 2

## Algorithm selected

The algorithm I have selected is the linear regression machine learning algorithm.

The linear regression model is a predictive model and results given have discrete values – it would not be appropriate for classification of data.

Linear regression is used to model the relationship between a series of variables, for example the relationship between area and length.

The linear regression model can be expressed as:



Where Yi describes the output variable being predicted, β is the weighting for each variable and x is the input variable ᵋ is the random noise error term.

The reason I have selected this algorithm is that linear regression would be best suited to model the relationship between variables and allow us to make predictions about another variable.

Before examining the data a few points to consider are that:

1. Linear regression can over fit the data if the inputs are correlated – in the case of this dataset, there are a number of inputs that are correlated
   1. Area is a function of length and width so it will be correlated with those input variable
   2. Perimeter is a function of length and width so it will be correlated with those input variable
   3. Compactness is calculated with reference to Area and Perimeter so it will be correlated to these variables (and length and width)
2. Outliers may have a disproportionate effect on the algorithm

## Dataset used

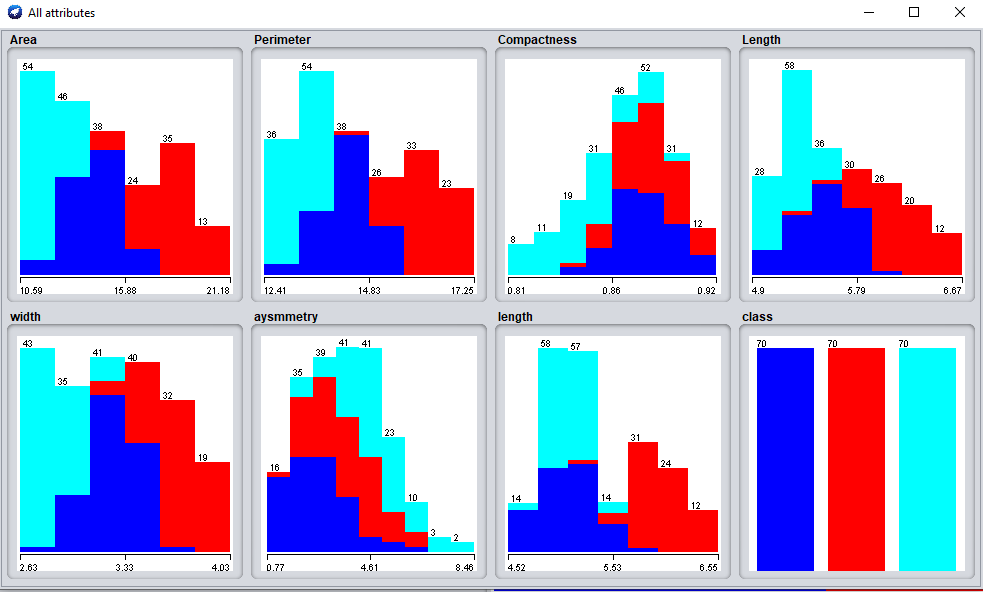
The dataset has been sourced from the link provided and before converting to a .arff, the following steps were taken:

1. Fix format for rows where tab spacing was out of line
2. Add the required header information i.e. @Relation and @attribute

The dataset also includes results for 3 different varieties, so I have initially classified my data by variety

When we have split the data by type a few immediate observations are that each variety has very distinct, so it makes sense to include a classification when performing the regression.

When ignoring the distinct classes, you can see that area, perimeter, length and width are positively correlated.



## Accuracy of algorithm

I firstly looked at the algorithm with just area, length and width.

This produced a linear relationship of:

Area = 2.9012 x length + 4.5502 x width – 16.3091

I ran the actual data through excel using this linear regression model output and found that all results were within +/-5% which intuitively seems like a reasonable outcome.

Next, I looked at the relative absolute error and mean squared error, both of which seemed reasonably low at 0.074833 and 0.1867 which appear reasonably low.

This analysis could be continued for all combinations of factors to assess the accuracy of the algorithm when predicting values of one variable relative to another.

In the appendix I have included the results for each of the parameters and have summarised the key information in the following table:

|  |  |  |  |
| --- | --- | --- | --- |
| Output | Correlation coefficient | Relative absolute error | Root Mean squared error |
| Area | 0.9994 | .031658 | 0.102 |
| Perimeter | 0.9993 | .034475 | 0.049 |
| Compactness | 0.6794 | .728035 | .0173 |
| Length | 0.9922 | 0.117485 | 0.0552 |
| Width | 0.9953 | 0.085798 | 0.0365 |
| Asymmetry | 0.5779 | 0.77933 | 1.2242 |
| Groove length | 0.9777 | 0.193425 | 0.103 |

As I am not using a classification algorithm, the accuracy of the algorithm is somewhat subjective.

With the exception of asymmetry and compactness, all other outputs are highly correlated with the inputs. They also have the highest values for the error terms.

## Learning curve for algorithm performance

I was unable to generate learning curves due to an error persisting which said access denied.

To generate learning curves, I would need to access the Advanced mode in experimenter, set up the destination (not in weka folder due to permissions), specify the results generator (regression split evaluator), add the dataset, specify generator properties and run – it is at this stage that I am getting the error. .

## Other Comments

For this data set, a classification algorithm such as the logistic regression may have been more appropriate.

## Appendix 1

**Assignment 1**

**References**

CT621 Artificial Intelligence (2021) Machine Learning (Part 2)

Available at: NUIG Blackboard

Accessed 26th June 2021

CT621 Artificial Intelligence (2021) Machine Learning (Part 3)

Available at: NUIG Blackboard

Accessed 26th June 2021

Moawad A (2018) Neural networks and back-propagation explained in a simple way.

Available at: <https://medium.com/datathings/neural-networks-and-backpropagation-explained-in-a-simple-way-f540a3611f5e>

Accessed 26th June 2021

Data source available at: <https://archive.ics.uci.edu/ml/datasets/seeds>

Multiple Linear regression

Available at: http://www.stat.yale.edu/Courses/1997-98/101/linmult.htm

Accessed: 26th June 2021

Dave P (2020) Linear Regression

Available at: https://medium.com/swlh/linear-regression-models-dc81a955bd39

Accessed: 26th June 2021

Kassambara (2018) Regression Model Accuracy Metrics

Available at: [http://www.sthda.com/english/articles/38-regression-model-validation/158-regression-model-accuracy-metrics-r-square-aic-bic-cp-and-more/](https://urldefense.proofpoint.com/v2/url?u=http-3A__www.sthda.com_english_articles_38-2Dregression-2Dmodel-2Dvalidation_158-2Dregression-2Dmodel-2Daccuracy-2Dmetrics-2Dr-2Dsquare-2Daic-2Dbic-2Dcp-2Dand-2Dmore_&d=DwMFaQ&c=3NBXXUKukgVIjVXwt0Rin6h0GAxIKZespWWvcJx4w9c&r=PyhisvEo9RBKGEQtQ1eYI0TixWoSLkcudJP2T2Qw1sANtEbc-bl34GM05l5SMjzK&m=tvB6EtNmUtEY31lX5uMsluta_ZQYO-7589DpYnCWqqI&s=qEpDDDlg8-9nVfZjAYWmk1rlwB7tx_XG7eDrf56IHG0&e=)

Accessed: 26th June 2021

## Appendix 2

**Perimeter Result**

=== Run information ===

Scheme: weka.classifiers.functions.LinearRegression -S 0 -R 1.0E-8 -num-decimal-places 4

Relation: SeedDataSet

Instances: 210

Attributes: 8

Area

Perimeter

Compactness

Length

width

aysmmetry

length

class

Test mode: evaluate on training data

=== Classifier model (full training set) ===

Linear Regression Model

Perimeter =

0.4164 \* Area +

-6.4852 \* Compactness +

0.2256 \* Length +

0.1383 \* width +

0.1008 \* class=1,2 +

12.2375

Time taken to build model: 0 seconds

=== Evaluation on training set ===

Time taken to test model on training data: 0 seconds

=== Summary ===

Correlation coefficient 0.9993

Mean absolute error 0.0388

Root mean squared error 0.049

Relative absolute error 3.4475 %

Root relative squared error 3.7605 %

Total Number of Instances 210

**Area Result**

=== Run information ===

Scheme: weka.classifiers.functions.LinearRegression -S 0 -R 1.0E-8 -num-decimal-places 4

Relation: SeedDataSet

Instances: 210

Attributes: 8

Area

Perimeter

Compactness

Length

width

aysmmetry

length

class

Test mode: evaluate on training data

=== Classifier model (full training set) ===

Linear Regression Model

Area =

1.8555 \* Perimeter +

12.0567 \* Compactness +

0.168 \* Length +

0.7321 \* width +

-0.1838 \* class=1,2 +

0.1171 \* class=2 +

-25.917

Time taken to build model: 0 seconds

=== Evaluation on training set ===

Time taken to test model on training data: 0 seconds

=== Summary ===

Correlation coefficient 0.9994

Mean absolute error 0.079

Root mean squared error 0.102

Relative absolute error 3.1658 %

Root relative squared error 3.5156 %

Total Number of Instances 210

**Length Result**

=== Run information ===

Scheme: weka.classifiers.functions.LinearRegression -S 0 -R 1.0E-8 -num-decimal-places 4

Relation: SeedDataSet

Instances: 210

Attributes: 8

Area

Perimeter

Compactness

Length

width

aysmmetry

length

class

Test mode: evaluate on training data

=== Classifier model (full training set) ===

Linear Regression Model

Length =

0.0371 \* Area +

0.2689 \* Perimeter +

-1.7339 \* Compactness +

-0.251 \* width +

0.2771 \* length +

0.0651 \* class=1,2 +

-0.1478 \* class=2 +

1.9973

Time taken to build model: 0 seconds

=== Evaluation on training set ===

Time taken to test model on training data: 0 seconds

=== Summary ===

Correlation coefficient 0.9922

Mean absolute error 0.0444

Root mean squared error 0.0552

Relative absolute error 11.7485 %

Root relative squared error 12.4856 %

Total Number of Instances 210

**Width Result**

=== Run information ===

Scheme: weka.classifiers.functions.LinearRegression -S 0 -R 1.0E-8 -num-decimal-places 4

Relation: SeedDataSet

Instances: 210

Attributes: 8

Area

Perimeter

Compactness

Length

width

aysmmetry

length

class

Test mode: evaluate on training data

=== Classifier model (full training set) ===

Linear Regression Model

width =

0.1046 \* Area +

0.0681 \* Perimeter +

3.6098 \* Compactness +

-0.1236 \* Length +

0.0068 \* aysmmetry +

-0.0551 \* length +

-1.4599

Time taken to build model: 0 seconds

=== Evaluation on training set ===

Time taken to test model on training data: 0 seconds

=== Summary ===

Correlation coefficient 0.9953

Mean absolute error 0.0277

Root mean squared error 0.0365

Relative absolute error 8.5798 %

Root relative squared error 9.6736 %

Total Number of Instances 210

**Compactness Result**

=== Run information ===

Scheme: weka.classifiers.functions.LinearRegression -S 0 -R 1.0E-8 -num-decimal-places 4

Relation: SeedDataSet

Instances: 210

Attributes: 8

Area

Perimeter

Compactness

Length

width

aysmmetry

length

class

Test mode: evaluate on training data

=== Classifier model (full training set) ===

Linear Regression Model

Compactness =

-0.0202 \* length +

0.0301 \* class=1,2 +

0.0223 \* class=2 +

0.9527

Time taken to build model: 0.01 seconds

=== Evaluation on training set ===

Time taken to test model on training data: 0 seconds

=== Summary ===

Correlation coefficient 0.6794

Mean absolute error 0.0138

Root mean squared error 0.0173

Relative absolute error 72.8035 %

Root relative squared error 73.3756 %

Total Number of Instances 210

**Aysmmetry Result**

=== Run information ===

Scheme: weka.classifiers.functions.LinearRegression -S 0 -R 1.0E-8 -num-decimal-places 4

Relation: SeedDataSet

Instances: 210

Attributes: 8

Area

Perimeter

Compactness

Length

width

aysmmetry

length

class

Test mode: evaluate on training data

=== Classifier model (full training set) ===

Linear Regression Model

aysmmetry =

0.9774 \* class=2,3 +

1.1436 \* class=3 +

2.6674

Time taken to build model: 0.02 seconds

=== Evaluation on training set ===

Time taken to test model on training data: 0.01 seconds

=== Summary ===

Correlation coefficient 0.5779

Mean absolute error 0.9562

Root mean squared error 1.2242

Relative absolute error 77.933 %

Root relative squared error 81.6134 %

Total Number of Instances 210

**Groove Length Result**

=== Run information ===

Scheme: weka.classifiers.functions.LinearRegression -S 0 -R 1.0E-8 -num-decimal-places 4

Relation: SeedDataSet

Instances: 210

Attributes: 8

Area

Perimeter

Compactness

Length

width

aysmmetry

length

class

Test mode: evaluate on training data

=== Classifier model (full training set) ===

Linear Regression Model

length =

0.0375 \* Area +

0.8837 \* Length +

-0.3074 \* width +

0.2475 \* class=3,2 +

0.1033 \* class=2 +

0.6792

Time taken to build model: 0 seconds

=== Evaluation on training set ===

Time taken to test model on training data: 0 seconds

=== Summary ===

Correlation coefficient 0.9777

Mean absolute error 0.0816

Root mean squared error 0.103

Relative absolute error 19.3425 %

Root relative squared error 21.0141 %

Total Number of Instances 210