Instituto Tecnológico y de Estudios Superiores de Monterrey Campus Querétaro

Neural Networks: Implementation of Perceptron

Laboratory Report

Authors:

Christian Ricardo Solís Cortés A01063685

Raúl Mar A00512318

Professor: Ruben Stranders

Artificial Intelligence

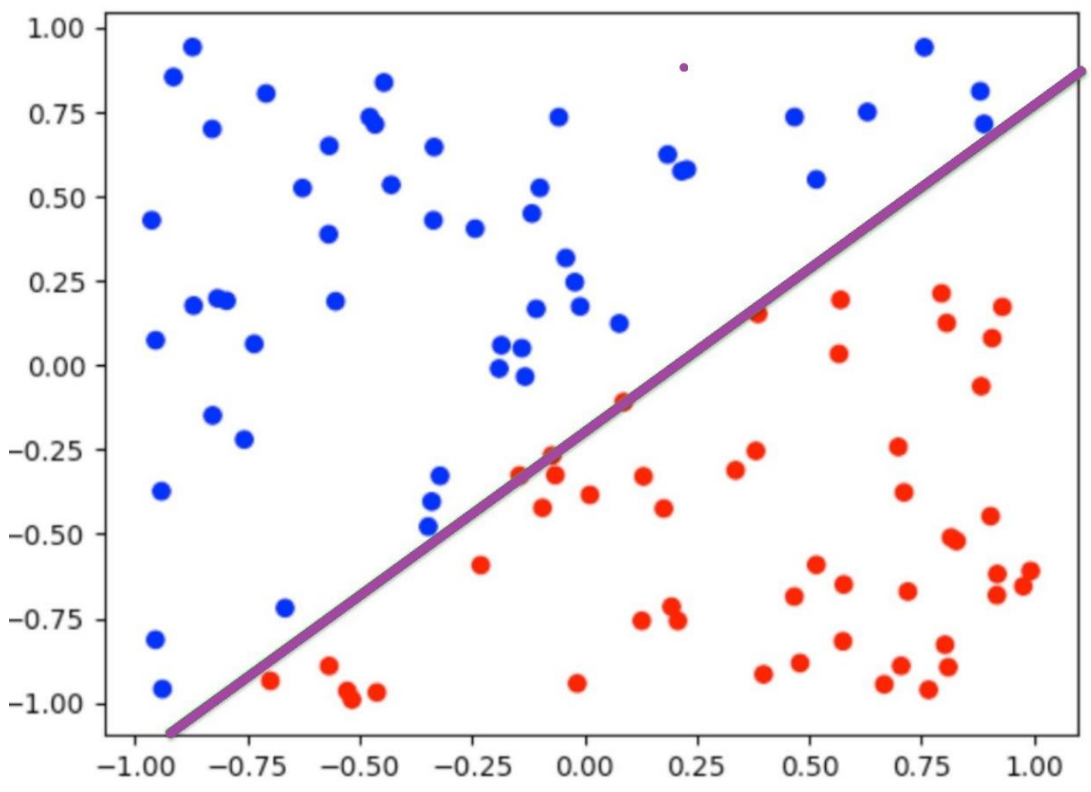
Santiago de Querétaro, Querétaro, México. Delivery Date: April 20th, 2018

Report: Implementation of Perceptron

Part I

In this laboratory, we had to implement the most basic algorithm for Neural Networks; the perceptron. After that, we had to train it with our given examples to get some predictions, or in the case of the data to be non-linear separable, return no solution found.

As requested, the following is a *scatter plot* of the training set that corresponds to the **linearly separable** example of the laboratory:

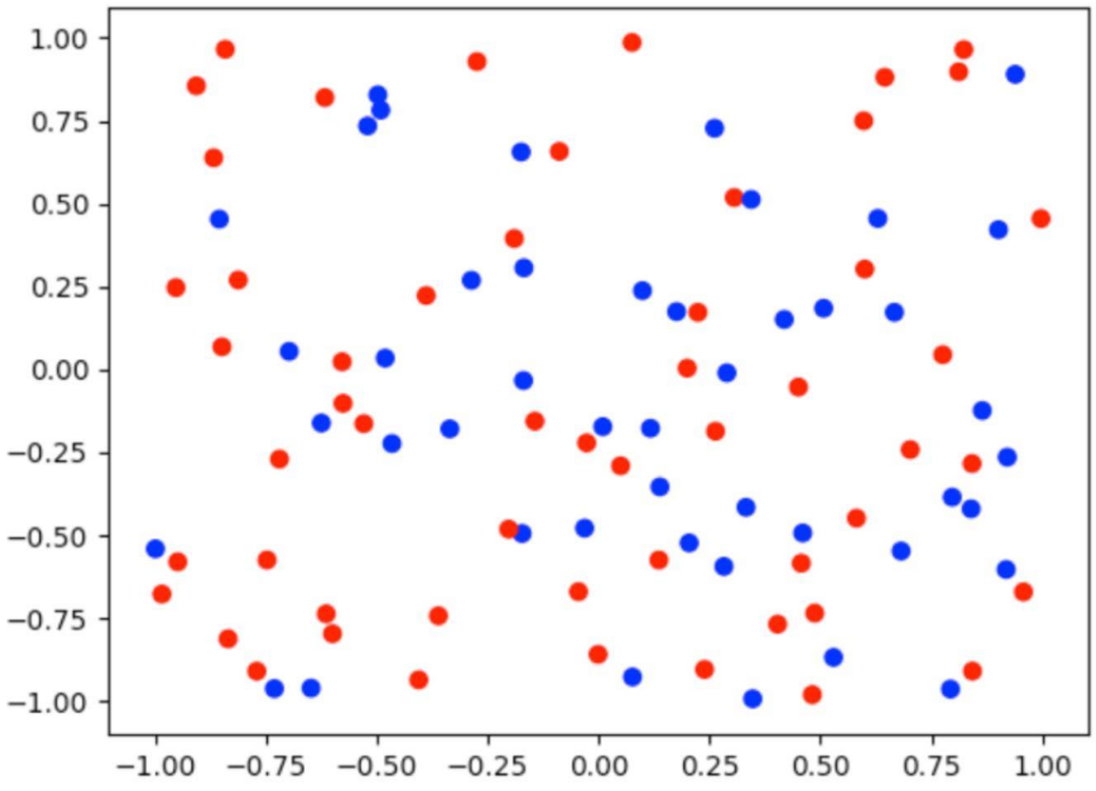


The representation of the line is possible with the following equation (it's an approximation):

$$f(x) = 1\ if\ (x[0] * 0.0124 + x[1] *\ -0.014255) > thresehold$$

$$else\ 0$$

The following is a *scatter plot* of the training set that corresponds to a **non-linearly separable** example of the laboratory. The difference in the plot is that there is no way to separate both colors.



Part II

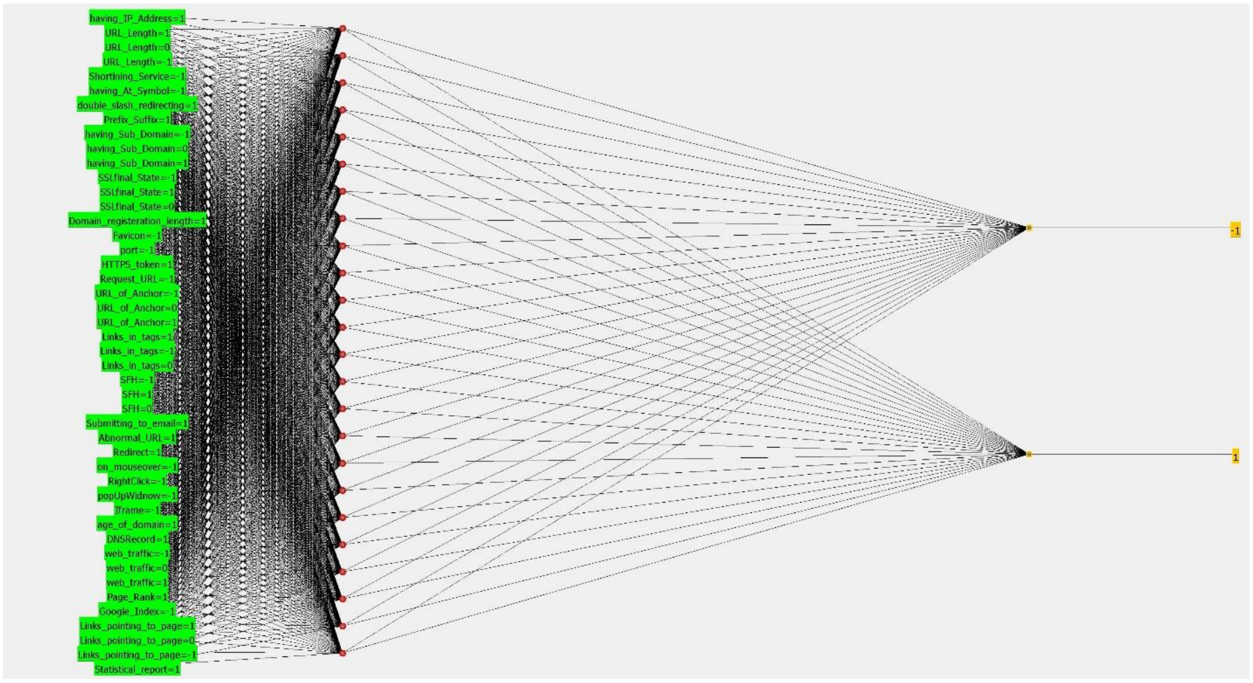
For the second part of the report we used the tool called WEKA and the data from a dataset found in a collection of datasets page, previously provided. The dataset is called Pishing Websites Data Set and it can be found here: https://archive.ics.uci.edu/ml/machine-learning-databases/00327/Training%20Dataset.arff

The intention of this dataset is to play with the attributes of an Artificial Neural Network, such as hidden layers, nodes, etc.

The following are some of the ANN created:

Example 1: Original ANN with no changes

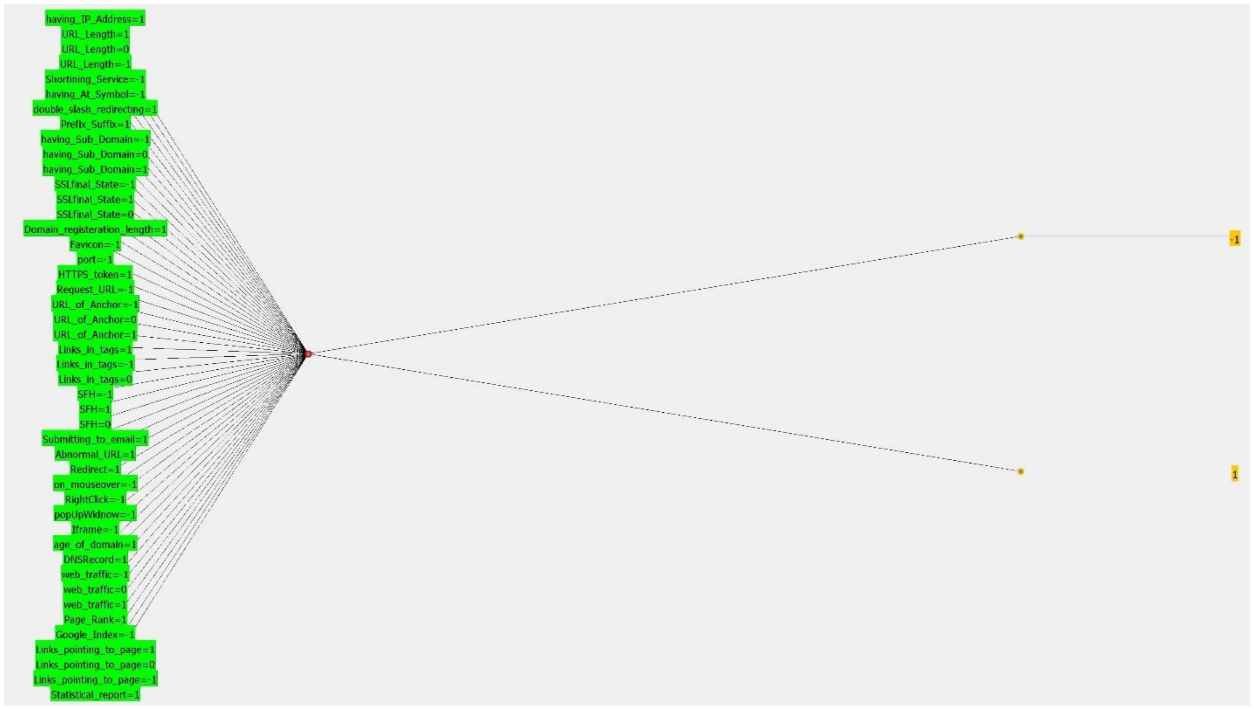
Hidden Layers	1
Learning Rate	0.3
Momentum	0.2
Nodes	24
Time taken to test model on training data	0.2 sec
Number of Epochs	500
Correct Classify Instances	100%



Example 2:

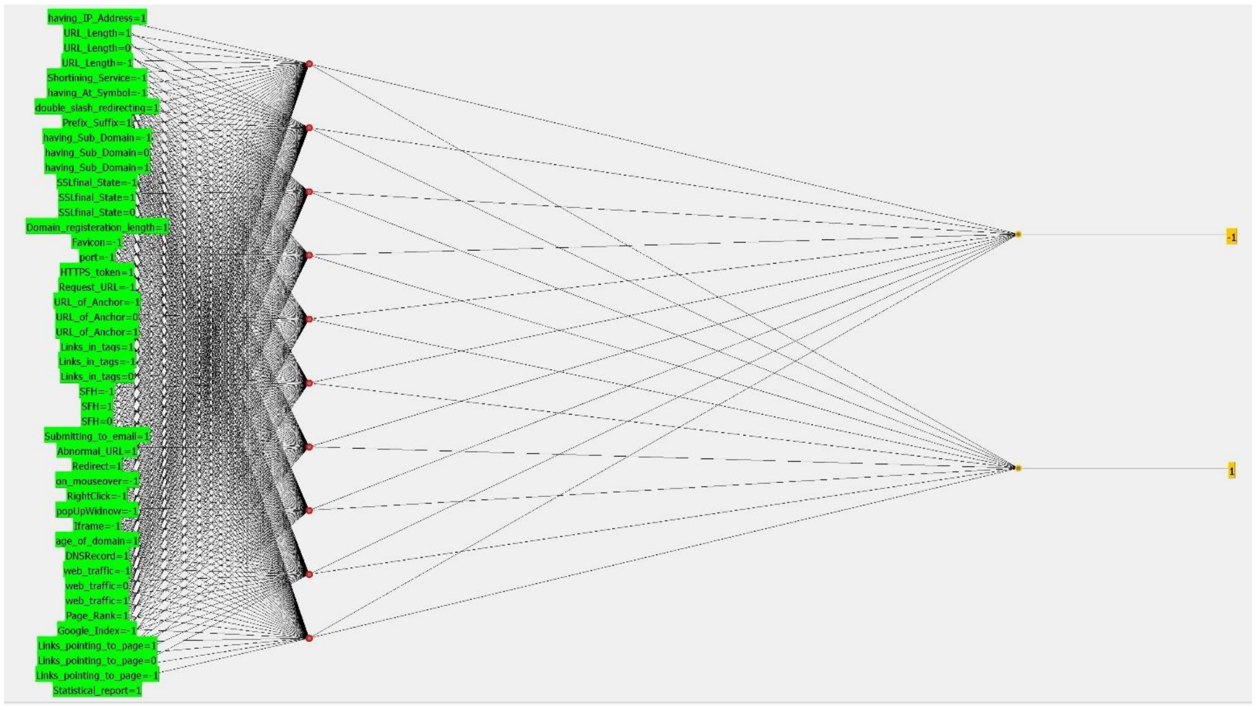
Hidden Layers	1
Learning Rate	0.6
Momentum	0.4
Nodes	1
Time taken to test model on training data	0.03 sec

Number of Epochs	250
Correct Classify Instances	100%



Example 3:

Hidden Layers	*1*
Learning Rate	***1***
Momentum	***1***
Nodes	***10***
Time taken to test model on training data	***0.1 sec***
Number of Epochs	***500***
Correct Classify Instances	***100%***



Summary by WEKA:

Example 1:

```
Time taken to build model: 1036.36 seconds
=== Evaluation on training set ===
Time taken to test model on training data: 0.2 seconds
=== Summary ===
Correctly Classified Instances       10916               98.7427 %
Incorrectly Classified Instances       139                1.2573 %
Kappa statistic                       0.9745
Mean absolute error                      0.0166
Root mean squared error                    0.0955
Relative absolute error                  3.3672 %
Root relative squared error                19.2166 %
Total Number of Instances            11055
```

Example 2:

```
Time taken to build model: 118.9 seconds
=== Evaluation on training set ===
Time taken to test model on training data: 0.03 seconds
=== Summary ===
Correctly Classified Instances      10423              94.2831 %
Incorrectly Classified Instances      632               5.7169 %
Kappa statistic                    0.8834
Mean absolute error                  0.1183
Root mean squared error                0.2323
Relative absolute error             23.9685 %
Root relative squared error           46.7555 %
Total Number of Instances           11055
```

Example 3:

```
Time taken to build model: 117.52 seconds
=== Evaluation on training set ===
Time taken to test model on training data: 0.1 seconds
=== Summary ===
Correctly Classified Instances       6157              55.6943 %
Incorrectly Classified Instances     4898              44.3057 %
Kappa statistic                    0
Mean absolute error                  0.4992
Root mean squared error                0.4993
Relative absolute error            101.1563 %
Root relative squared error          100.5102 %
Total Number of Instances           11055
```

What we notice?

When we represent the networks with the tool, we notice that there are big changes when we move the parameters in the tables. To prove this, we can notice the difference between the original ANN with its original parameters, vs the third one with less than half the nodes.

There's another variation found, is in terms of the *Root Relative Error* and the *Relative Absolute error.* There are variations because the complexity of the network is changed

What are the ANNs good for?

Artificial Neural Networks tend to copy the function of the neurons of the brain. This lead to be useful for organic learning. ANNS are good to approximate functions. They are useful to solve problems like recognition of images of any kind. For this, their value is enormous and is widely used on big data analysis.

Where would you use them?

ANNs have lots of applications due to their classifier characteristics, some of the most used applications are pattern recognition, prediction of air flows, load-forecasting, hand-writing recognition, speech recognition, hospital patient stay length prediction, breast cancer cell image classification, among others…

Are they worth the effort implementing, or not?

Yes. This is one of the most useful techniques of machine learning because it's very accurate, even when its speed is not fast.

What kind of problems do they not solve?

As we cannot really know what happens inside the network, we don't get much information.

Nevertheless, problems like understanding natural language and answer a set of questions are some of the problems ANNs can´t solve yet.