# Breast cancer classification using Neural Network Approach

Shakir Ullah

14031222

shakir2014@namal.edu.pk

03 December 2017

## Table of Contents

## 1. Abstract

Breast cancer is a fatal disease mostly founded in women. There are so many factors that causes the breast cancer and there is a large amount of breast cancer data. Because of that it may be impossible for humans to analyze this data and to find out what are the most common factors that play important role in causing breast cancer. In this situation, it is important to examine that machine can classify the cancer as either benign or malignant based on cell description gathered by microscopic examination. So in this project I will be evaluating machine learning technique (Feedforward Neural Network) on the dataset provided to us.

## 2. Introduction

The purpose of this report is to evaluate machine learning technique (Feedforward Neural Network) on the given data to classify the cancer as either benign or malignant.

Breast cancer is most common type of cancer in women and the second prominent cause of cancer related deaths, next to lungs cancer. Although men can also get breast cancer, cases of male breast cancer account for less than 0.05% of all breast cancer cases diagnosed. If eight women live to the age of 85, at least one of them will develop breast cancer in her lifetime. Two third of the women diagnosed with breast cancer are over the age of 50, and the majority of the remaining women diagnosed with breast cancer are between the age of 39 and 40 [1].

In this project, Feedforward neural network with different architectures was evaluated and the results of these architecture has been recorded.

## 3. Background

Before moving further into the depth of the problem and its solution, it is important to know the basics of domain so that it can provide us a better understanding of the problem and the project.

### 3.1 Breast Cancer

In this project, I dealing with breast cancer, then what is breast cancer? Breast cancer is caused when abnormal tissue in the breast begins to multiply uncontrollably. These cancerous cells can travel to other locations in the body and cause further damage [1]. The breast cancer is classified as benign and malignant. This classification is made using cell description by microscopic examination.

### 3.2 Artificial Neural Network

Artificial neural network (ANN) is a popular machine learning technique, it has made on the model of a human brain. There are three kinds of layer in artificial neural network; input layer, hidden layer and output layer. The number of neurons and hidden layers can be different in distinct layers and networks, respectively. The basic architecture of artificial neural network is shown below.

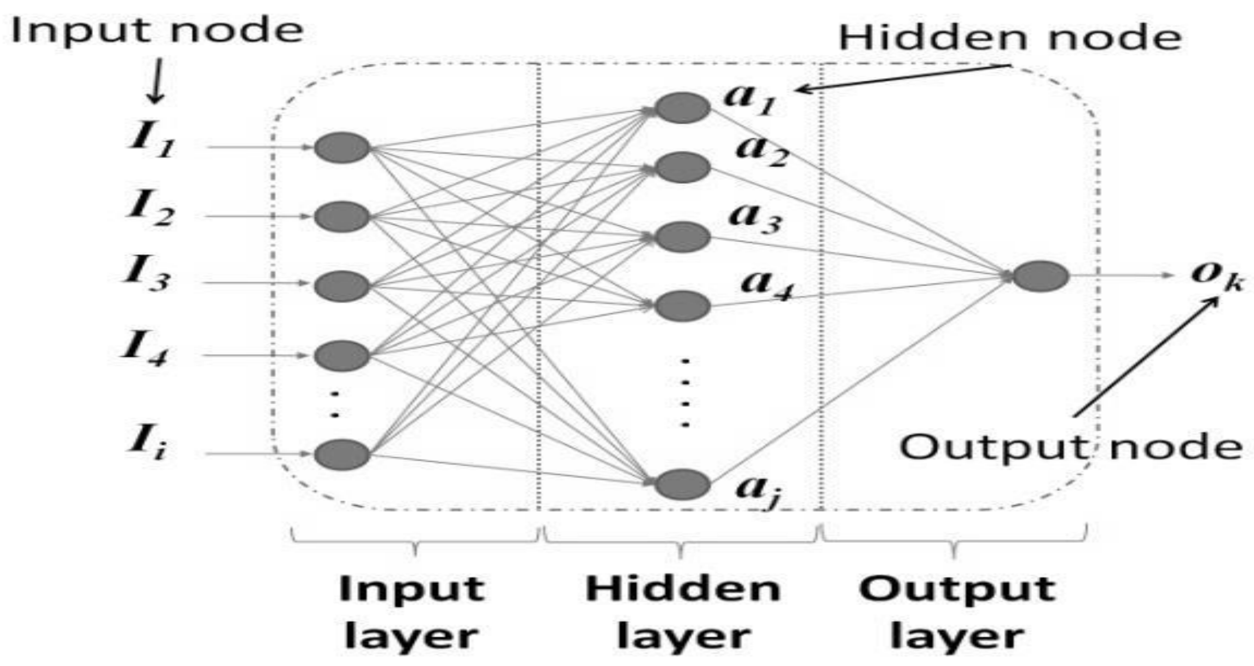


*Figure 1: Basic Architecture of ANN [4]*

### 3.3 Feedforward Neural Network

As its name suggest, it is a type of neural network in which inputs are feed forwarded into the layers starting from the input layer, passing on to the hidden layers and then ultimately output is produced at the output layer. This network is named as feedforward because all the layers are forwarded connected and there is no back connection in any layer.

## 4. Approaches and Experiments

As there are many machine learning techniques to start with, but I have evaluated only a single technique with several architectures. The technique I have evaluated is feedforward neural network.

### 4.1 Architecture

I have briefly explained feedforward neural network in section 3.3. The network architecture I have used for this problem is 9-5-5-1. It has 9 neurons in the input layer, 5 neurons in first hidden layer, 5 neurons in second hidden layer and as I have to predict either its benign or malignant, thus 1 neuron in the output layer. I got the idea of choosing numbers of hidden layers and neurons in each layer from an article [2]. The figure below further explain the structure of the network.

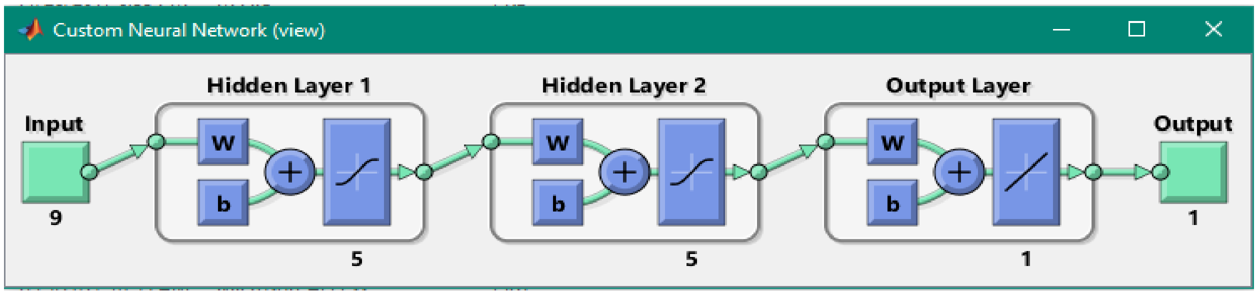


*Figure 2: Structure of the network*

## 4.2 Network Parameter

As there are a lot of training functions and transfer function but I have choose some of them. The parameters of the network on which it was built and then trained accordingly by providing the inputs and desired outputs are shown below.

1. Maximum number of epochs to train the network to train was set to 1000.
2. Performance goal for the network to achieve was 0.01.
3. Training Algorithms used was: Gradient descent with momentum and adaptive learning rate backpropagation (traingdx), Conjugate gradient backpropagation (traincgb) and Bayesian regularization backpropagation (trainbr).
4. Activation functions for the layers were 'tansig', 'logsig' [3].
5. Maximum fail for the network was set to 200.
6. Learning rates for the network were 0.01, 0.1.

Rest of the required parameters were used as default.

## 4.3 Feature set

The feature set for the given problem was obtained, based on cell description gathered by microscopic examination. The input feature set is consist of Clump thickness, Uniformity of cell size, Uniformity of cell shape, Marginal adhesion, Single epithelial cell size, Bare nuclei, Bland chromatin, Normal nucleoli and Mitosis. The domain of these ranges from 1-10. The output is either 2 for benign or 4 for malignant. Some of the values of the input features were missing, so I replaced these by taking the mean of the domain range of these values. I calculated the mean as, I sum the numbers from 1 to 10 and divide the sum by 9 i.e.

$$sum = 1+2+3+4+5+6+7+8+9+10$$

$$sum = 55$$

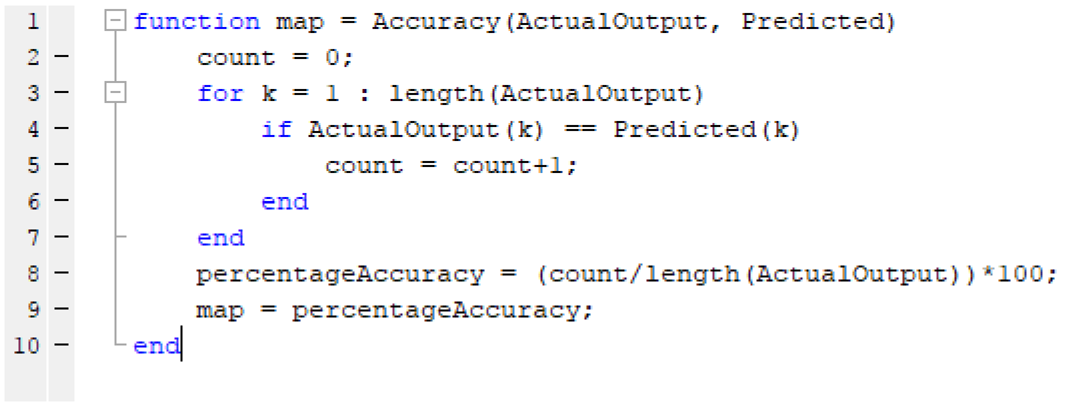
$$mean = \frac{sum}{9}$$

$$mean = \frac{55}{9} = 5.5$$

As the range of the feature parameters are whole numbers from 1 to 10, so I take the mean (5.5) as a whole number which 5.

## 4.4 Accuracy

To find that how our network will perform, I have written a function for this, which compare the actual output and predicted output and return the percentage accuracy of the network. The matlab function for this is shown below,



```
function map = Accuracy(ActualOutput, Predicted)
    count = 0;
    for k = 1 : length(ActualOutput)
        if ActualOutput(k) == Predicted(k)
            count = count+1;
        end
    end
    percentageAccuracy = (count/length(ActualOutput))*100;
    map = percentageAccuracy;
end
```

*Figure 3: Accuracy function*

## 4.5 Experiment # 1

In all the experiments, I made the assumptions on data distribution only and with that data I am experimenting the different network structures.

### 4.5.1 Hypothesis

For the first experiment, my hypothesis is that if we train a network with a high number of training data then the performance will be better. For example if we train the network on 70% data and test it on 30% data, its performance will be better than to train it on 60% and test it on 40% data and so on. For evaluating this, the distribution I choose was **Training: 75%** and **Testing: 25%.**

### 4.5.2 Training

I train the network by providing 75% of the total data as Training input and its respective actual output. After training the network, I tested the network providing only the training input and testing input separately.

### 4.5.3 Testing on Training input

For testing the network first I provided the training inputs only and record the predicted output. After that, I calculated the accuracy of each network. Experiments that I have carried out by changing training function, transfer function and learning rate for the given hypothesis by providing the training inputs are shown in the table [5].

| S.No | Training Function | Transfer Function | Learning Rate | Accuracy (%) |
|---|---|---|---|---|
| 1 | Traingdx | logsig | 0.01 | 96.3740 |
| 2 | Traingdx | tansig | 0.01 | 96.5649 |
| 3 | Trainbr | logsig | 0.01 | 96.7557 |
| 4 | Trainbr | tansig | 0.01 | 96.1832 |
| 5 | Traingdx | tansig | 0.1 | 96.3740 |
| 6 | Trainbr | logsig | 0.1 | 97.7099 |
| 7 | Trainbr | tansig | 0.1 | 96.3740 |

*Table 1*

The table above show that, the trainbr as training function, logsig as transfer function and learning rate as 0.1 give the best result.

### 4.5.4 Testing on testing input

For testing the network now I provided the testing inputs only and record the predicted output. After that, I calculated the accuracy of the network. Experiments that I have carried out for the given hypothesis by providing the testing inputs is shown in the table.

| S.No | Training Function | Transfer Function | Learning Rate | Accuracy (%) |
|---|---|---|---|---|
| 1 | Traingdx | logsig | 0.01 | 99.4286 |
| 2 | Traingdx | tansig | 0.01 | 99.4286 |
| 3 | Trainbr | logsig | 0.01 | 98.8571 |
| 4 | Trainbr | tansig | 0.01 | 99.4286 |
| 5 | Traingdx | tansig | 0.1 | 97.1429 |
| 6 | Trainbr | logsig | 0.1 | 98.8571 |
| 7 | Trainbr | tansig | 0.1 | 99.4286 |
| 8 | Traincgb | tansig | 0.1 | 99.4286 |

*Table 2*

The table above show that, all the combinations gives almost the same accuracy. Most of them (1, 2, 4, 7 and 8) are very accurate.

### 4.5.5 Analysis of experiment # 1

After different experimentations (shown in table 1 and 2), the results were quite good. But I cannot say that my hypothesis is correct unless I evaluate the same networks with different data distribution. For doing that, I changed the data distribution to; **Training: 50%** and **Testing: 50%**.

## 4.6 Experiment # 2

This experiment is to find whether the hypothesis I have made is actually correct or not. May be with equal distribution of data for training and testing will also give good results.

### 4.6.1 Hypothesis

This is a counter hypothesis of the first hypothesis. The hypothesis is that it will perform not very well as compared to the first hypothesis. For evaluating this, the distribution I choose was **Training: 50%** and **Testing: 50%.**

### 4.6.2 Training

I train the network by providing 50% of the total data as Training input and its respective actual output. After training the network, I tested the network providing only the training input and testing input separately.

### 4.6.3 Testing on training input

For testing the network I provided the training inputs only and record the predicted output. After that, I calculated the accuracy of the network. Experiments that I have carried out are shown in the table.

| S. No | Training Function | Transfer Function | Learning Rate | Accuracy (%) |
|---|---|---|---|---|
| **1** | Traingdx | Tansig | 0.01 | 96 |
| **2** | Traingdx | Logsig | 0.01 | 94.5714 |
| **3** | Trainbr | Logsig | 0.01 | 95.7143 |
| **4** | Trainbr | Tansig | 0.01 | 96 |
| **5** | Traingdx | Tansig | 0.1 | 96.2857 |
| **6** | Trainbr | Logsig | 0.1 | 95.7143 |
| **7** | Trainbr | Tansig | 0.1 | 94.8571 |
| **8** | Traincgb | Tansig | 0.1 | 96.5714 |

*Table 3*

The table above show that, the last combination which traincgd as training function, tansig as transfer function and 0.1 as the learning rate give the best result.

### 4.6.4 Testing on testing input

For testing the network now I provided the testing inputs only and record the predicted output. After that, I calculated the accuracy of each network. Experiments that I have carried out by providing the testing inputs is shown in the table.

| S.No | Training Function | Transfer Function | Learning Rate | Accuracy (%) |
|---|---|---|---|---|
| **1** | Traingdx | tansig | 0.01 | 98.5673 |
| **2** | Traingdx | logsig | 0.01 | 98.5673 |
| **3** | Trainbr | logsig | 0.01 | 97.1347 |
| **4** | Trainbr | tansig | 0.01 | 97.1347 |
| **5** | Traingdx | tansig | 0.1 | 97.9943 |
| **6** | Trainbr | logsig | 0.1 | 96.8481 |
| **7** | Trainbr | tansig | 0.1 | 97.4212 |
| **8** | Traincgb | tansig | 0.1 | 97.9943 |

*Table 4*

The table above show that, the first two combination; traingdx as training function, tansig (for first combination) and logsig (for second combination) as transfer functions and 0.1

as the learning rate give the best result. The accuracy of all other combinations were also closed enough to the best.

#### 4.6.5 Analysis of experiment # 2

After the above experimentations (shown in table 3 and 4) where data distribution was **Training: 50%** and **Testing: 50%,** the accuracies of these networks are not very different from the accuracies shown in table 1 and 2 where data distribution was **Training: 75%** and **Testing: 25%.** Which mean that the hypothesis I have made is true but not very well because difference is not too much. Now let's change the data distribution to **Training: 30%** and **Testing: 70%.**

## 4.7 Experiment # 3

The second experiment did not make much difference, that's why to challenge my hypothesis I have made at the start, I decease the training data too much to identify the difference.

#### 4.7.1 Hypothesis

This is another counter hypothesis of the first hypothesis. The hypothesis is that, with this distribution, the accuracy will be much lower than the accuracy of my first hypothesis. For evaluating this, the distribution I choose was **Training: 30%** and **Testing: 70%.**

#### 4.7.2 Training

I train the network by providing 30% of the total data as Training input and its respective actual output. After training the network, I tested the network providing only the training input and testing input separately.

#### 4.7.3 Testing on training input

For testing the network I provided the training inputs only and record the predicted output. After that, I calculated the accuracy of the network. Experiments that I have carried out are shown in the table.

| S.No | Training Function | Transfer Function | Learning Rate | Accuracy (%) |
|---|---|---|---|---|
| **1** | Traingdx | tansig | 0.01 | 97.6190 |
| **2** | Traingdx | logsig | 0.01 | 98.0952 |
| **3** | Trainbr | logsig | 0.01 | 96.1905 |
| **4** | Trainbr | tansig | 0.01 | 97.6190 |
| **5** | Traingdx | tansig | 0.1 | 98.0952 |
| **6** | Trainbr | logsig | 0.1 | 97.1429 |
| **7** | Trainbr | tansig | 0.1 | 97.6190 |
| **8** | Traincgb | tansig | 0.1 | 96.1905 |

*Table 5*

The above table show that, when the training function is traingdx, the accuracy will be much better as in combination 1, 2 and 5. In these combinations the accuracy is greater than the other combinations.

### 4.7.4 Testing on testing input

For testing the network now I provided the testing inputs only and record the predicted output. After that, I calculated the accuracy of each network. Experiments that I have carried out by providing the testing inputs is shown in the table.

| S. No | Training Function | Transfer Function | Learning Rate | Accuracy (%) |
|---|---|---|---|---|
| **1** | Traingdx | tansig | 0.01 | 96.1145 |
| **2** | Traingdx | logsig | 0.01 | 97.1370 |
| **3** | Trainbr | logsig | 0.01 | 96.9325 |
| **4** | Trainbr | tansig | 0.01 | 96.5235 |
| **5** | Traingdx | tansig | 0.1 | 96.3190 |
| **6** | Trainbr | logsig | 0.1 | 96.7280 |
| **7** | Trainbr | tansig | 0.1 | 97.3415 |
| **8** | Traincgb | tansig | 0.1 | 96.9325 |

*Table 6*

This table show that, traingdx (training function) with logsig (transfer function) and learning rate as 0.01 and trainbr (training function) with tansig (transfer function) and learning rate as 0.1 gives the best accuracy. Over all, the accuracy of almost all the combinations (1 to 8) was very close.

### 4.7.5 Analysis of experiment # 3

Experiment no 3, shows that the accuracy is not so much different from the results of previous two experiments. The accuracies of all three experiments are approximately differ by a factor of 1. Best accuracy on testing inputs of experiment # 2 is 99.4286 %, experiment # 3 is 98.5673 % and experiment # 3 is 97.3415 %.

## 5. Over All Analysis

At the start of the project my hypothesis was that, if we train the network on high percentage of the total data then the accuracy will be much better. But after all the experiments, the difference of the accuracies of different networks were not too much. The difference between the best accuracies of the all three experiments was 1. The difference was not too much because the distribution of data (benign and malignant) is equally distributed through the whole dataset.

## 6. References

[1] Introduction to Breast Cancer: Causes and Risk Factors, accessed date 2 December 2017, https://www.babymed.com/cancer/introduction-breast-cancer-causes-and-risk-factors

[2] The Number of Hidden Layers, accessed date 25 November 2017, http://www.heatonresearch.com/2017/06/01/hidden-layers.html

[3] Vehbi Olgac, A & Karlik, Bekir. (2011). Performance Analysis of Various Activation Functions in Generalized MLP Architectures of Neural Networks. International Journal of Artificial Intelligence And Expert Systems. 1. 111-122.

[4] basic+architecture+of+nn – Google Search https://www.google.com.pk/search?q=basic+architecture+of+nn&rlz=1C1GCEB_enPK768PK769&source=lnms&tbm=isch&sa=X&ved=0ahUKEwjUl6u3pe3XAhVLvY8KHVziC0AQ_AUICigB&biw=1600&bih=720#imgrc=B9YVqgpENJktoM:

[5] Peace, I. C, Uzoma, A. O, & Ita, S. A. (2015). Effect of Learning Rate on Artificial Neural Network in Machine Learning. *International Journal of Engineering Research & Technology, 4(02),* 2278-0181.