COS 314 Assignment 3

Tayla Orsmond u21467456

# Machine Learning: Comparing the Effectiveness of Different Machine Learning Models in Classifying Breast Cancer

## Introduction

The goal of this assignment is to design different machine learning models to perform classification of breast cancer. This assignment makes use of a dataset from the UCI Machine Learning Repository: <https://archive.ics.uci.edu/ml/datasets/breast+cancer>. The effectiveness of each model is then assessed according to different statistical significance tests.

## Environment (Specs)

This assignment was completed on a Dell Inspiron 7490 laptop with the following specs:

**Processor**: Intel(R) Core (TM) i7-10510U CPU @ 1.80GHz 2.30 GHz

**RAM**: 16.0 GB (15.8 GB usable)

**Development** **Environment**: Visual Studio Code, JDK version 17.0.6

## Artificial Neural Network (ANN)

An Artificial Neural Network (ANN) is a mathematical model of the brain that aims to predict patterns within data. ANNs are designed to model how humans learn. In this case, the ANN performs binary classification, i.e., it assigns one of two labels (classes) to a particular instance based on the pattern the ANN recognizes. The classes the ANN can assign in this case are **recurrence-events** or **no-recurrence-events**.

**Recurrence-events** refers to the patient having cancer and is represented by **a class of 1** in this instance. Similarly, **no-recurrence-events** refers to the patient not having cancer and is given **a class of 0**.

The rest of the dataset was pre-processed for use in the ANN as follows:

### Data Transformation and Pre-Processing

*This breast cancer domain was obtained from the University Medical Centre, Institute of Oncology, Ljubljana, Yugoslavia. Thanks go to M. Zwitter and M. Soklic for providing the data.*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Domain | # Attributes | # Nominal | # Linear | # Integer | # Instances |
| Breast cancer | 9 (+ class) | 5 (+ class) | 3 | 1 | 286 |

*Table 1. Summary of Dataset*

This dataset includes 286 instances in total, with 201 instances of no-recurrence-events and 85 instances of recurrence-events. The instances are described by 9 attributes, some of which are linear, and some are nominal. There are four instances with missing values, denoted by “?”.

|  |  |
| --- | --- |
| Attribute | Possible Values |
| **class** | no-recurrence-events, recurrence-events |
| **age** | 10-19, 20-29, 30-39, 40-49, 50-59, 60-69, 70-79, 80-89, 90-99 |
| **menopause** | lt40, ge40, premeno |
| **tumor-size** | 0-4, 5-9, 10-14, 15-19, 20-24, 25-29, 30-34, 35-39, 40-44, 45-49, 50-54, 55-59 |
| **inv-nodes** | 0-2, 3-5, 6-8, 9-11, 12-14, 15-17, 18-20, 21-23, 24-26, 27-29, 30-32, 33-35, 36-39 |
| **node-caps** | yes, no |
| **deg-malig** | 1, 2, 3 |
| **breast** | left, right |
| **breast-quad** | left-up, left-low, right-up, right-low, central |
| **irradiat** | yes, no |

#### Table 1.1.: Attribute Information

As was previously stated, **“recurrence-events” were represented by the class 1, and “no-recurrence-events” were represented by the class 0**. Instances that had **values** **missing**, had those values **replaced by 0**, in order that they have no effect on the outcome of the ANN’s decision making. This was done because these missing values belonged to instances of class “recurrence-events”, which already make up a smaller portion of the data, so it was important that these instances were preserved.

The rest of the data was encoded through **Hot-one encoding**. This type of encoding involves assigning each value of an attribute a unique binary string indicating its position in the sequence. For example, for the node-caps attribute, the value of “yes” is denoted by the string 10, while “no” is denoted by 01. This was done to avoid nominal variables of higher values accidentally having a higher impact on the outcome of the ANN’s decision making process (like what might be the case with integer encoding). For example, the age attribute has value ranges from 10-99. If these values were encoded using integer encoding (i.e., 10-19 was given the value 1 and 90-99 was given the value 9), then 90-99 would have a higher impact on results than 10-99 etc. This could otherwise lead to erroneous results or poor performance.

The downside of Hot-one encoding is that there are many inputs to the ANN, which could lead to overfitting as the dimensionality of the data is very large. This is addressed again later.

The data was then **split into a training and testing set by a 70-30% split** respectively. The instances were shuffled and randomly selected to be in either set in order that a good portion of the differences in the domain are represented.

### Design Components and Parameters

|  |  |  |
| --- | --- | --- |
| Parameter | Initial Values | Final Values |
| seed |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

*Table 2: ANN Design Components and Parameters Before and After Tuning*

Above is a table of parameters before and after fine-tuning. The fine-tuning process happened offline, whereby each parameter was incrementally adjusted until there was no more improvement in the ANN’s performance. An “improvement” was an increase in the average accuracy of the algorithm over 100 runs using randomly generated seeds. The best of these seeds was then used as the seed for the final ANN.

A statistical representation of these parameters during tuning is shown in the graphs below:

[graphs]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]

The **ReLu** Activation function was used as the activation function for the Hidden layer because [][][][][][][[][]

The **Binary** **Sigmoid** activation function was used as the activation function for the output layer to scope the output between 0 and 1 for easy binary classification of instances. The data was binary encoded, which also makes this function a good fit in this case. **Outputs over 0.5 were considered class 1, and outputs below 0.5 were considered class 0.**

To **avoid** **overfitting** (due to, as previously mentioned, the large number of inputs into the ANN), early stopping and thinning of the layers was implemented. The number of hidden nodes was thinned to [\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_] 9 (the number of classes) from 51 (equal to the number of inputs). A fixed cap on the maximum number of epochs of the training set was also implemented [\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_] 50, as well as a stopping condition that indicated if the ANN’s output did not change by a tolerance margin of [\_\_\_\_\_\_\_\_\_\_] 0.01 for [\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_] 10 iterations, then the algorithm was considered to have converged and the training was stopped. This was done to hopefully allow the algorithm to generalize better and thus perform better on the test set.

## Genetic Programming (GP)

The Genetic Programming metaheuristic uses Darwin’s principle of natural selection to produce programs which provide optimal solutions to problems. The algorithm begins by generating an initial population of individuals (programs) and, through several generations, modifies and updates the population to reflect the evolving individuals.

In this case, the GP is a classification algorithm that is used to evolve decision trees (classifiers) that classify breast cancer. The classes the trees can assign in this case are also **recurrence-events** or **no-recurrence-events**.

### Data Transformation and Pre-Processing

The Decision Trees were capable of learning directly from the nominal data and so **no transformation was necessary**.

The **exact same training set and test set instances** (just with the data in its raw form) were used in the training and testing of the decision trees as the ANN to ensure these methods were comparable.

### Individual Representation

Individuals in the algorithm are represented as decision trees in which each node can either be a functional or terminal node.

The functional and terminal sets for this case are shown below:

**Functional** **Set**: {age | menopause | tumor-size | inv-nodes | node-caps | deg-malig | breast | breast-quad | irradiat} with the output of each attribute (function) being one of the values of said attribute.

**Terminal** **Set**: {recurrence-events | no-recurrence-events}

### Design Components and Parameters





|  |  |  |
| --- | --- | --- |
| Parameter | Initial Values | Final Values |
| seed |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

*Table 3: GP Design Components and Parameters Before and After Tuning*

Above is a table of parameters before and after fine-tuning. Like the ANN parameters, these underwent fine-tuning in a similar manner. The best of the seeds collected was then used as the seed for the final GP algorithm.

A statistical representation of these parameters during tuning is shown in the graphs below:

[graphs]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]

The inclusion of mutation was found to drastically decrease the fitness of individuals generated. Thus, it was eventually decided that mutation would be removed from the algorithm, as the random generation of individuals provided enough diversity in the program space that mutation was unnecessary. This meant, as can be seen from the graphs [\_\_\_\_\_\_\_\_\_\_\_\_\_] that the fitnesses of the existing individuals were better exploited and thus better accuracy values of solutions were generated on average.

Weka C4.5 Decision Tree

This Machine Learning (ML) tool was used as a control to measure the standard for the other two algorithms used in this assignment. The parameter values were kept as default and so the algorithm was not tuned [[][]][][][][ at all for this dataset. The data was not preprocessed, but given in its raw form as an .arff file [][][][][][][][][ to the ML tool for classification. There was thus no way to ensure the training and test instances would be the same for this tool, but one can assume that there would be significant overlap in the instances due to the inclusion of only one small dataset.

## Results



Seed: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Dataset | # Instances | # Correct Instances | Accuracy | F-measure |  |
| Training set | 200 (70%) |  |  |  |  |
| Test set | 86 (30%) |  |  |  |  |
| Breast cancer (total) | 286 |  |  |  |  |

Table 4: Table Comparing Different Statistical Measures of the Performance of the ANN on the Breast Cancer Dataset

Seed: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Dataset | # Instances | # Correct Instances | Accuracy | F-measure |  |
| Training set | 200 (70%) |  |  |  |  |
| Test set | 86 (30%) |  |  |  |  |
| Breast cancer (total) | 286 |  |  |  |  |

Table 4: Table Comparing Different Statistical Measures of the Performance of GP on the Breast Cancer Dataset

Seed: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Dataset | # Instances | # Correct Instances | Accuracy | F-measure |  |
| Training set | 200 (70%) |  |  |  |  |
| Test set | 86 (30%) |  |  |  |  |
| Breast cancer (total) | 286 |  |  |  |  |

Table 4: Table Comparing Different Statistical Measures of the Performance of the Weka ML tool on the Breast Cancer Dataset

[graphs]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]

## Conclusion

As can be seen from the results above, the \_\_\_

Performing a [STAT TEST HERE] statistical analysis of the data, we find that the \_\_\_\_ has ….