COS 314 Practical 3
U19027372 Simphiwe Ndlovu
Assignment 3

The experiments were conducted on a machine with an Intel® Core™ i5-1035G1 CPU @ 1.00GHz × 8 and 8 GB of RAM . I implemented the algorithms in Java .

Technical specification

1. Artificial Neural Network

Introduction:
The Artificial Neural Network (ANN) model described in this report is designed to perform classification tasks using a feedforward neural network architecture. The model takes input data with nine parameters and predicts the output class. It uses a series of hidden layers and activation functions to process the input data and generate accurate predictions.

Model Architecture:
The ANN model consists of three main components: the input layer, hidden layers, and output layer. The number of hidden layers is configurable, allowing flexibility in model design. The model also includes bias nodes to adjust the activation of neurons in each layer. The architecture is as follows:

- Input Layer: The input layer consists of nodes representing the input parameters. In this model, there are nine input parameters.
- Hidden Layers: The model supports a variable number of hidden layers. Each hidden layer contains nodes that perform weighted sums of inputs and apply activation functions to generate output values.
- Output Layer: The output layer consists of a single node that provides the final classification prediction.

Training Process:
The training process of the ANN model involves iteratively feeding input data through the network, adjusting weights and biases, and backpropagating the errors to update the network parameters. The steps involved in the training process are as follows:

- a. Data Normalization: The input data is normalized to ensure consistency and proper range for each parameter.
- b. Feedforward Pass: The input data is fed into the input layer, and the weighted sums of inputs in the hidden layers are calculated. Activation functions are applied to obtain the output values of each hidden layer.
- c. Output Layer Calculation: The output layer receives inputs from the last hidden layer and produces the final prediction using a binary step function.
- d. Error Calculation: The error between the predicted output and the target output is calculated.
- e. Backpropagation: The error is propagated back through the network to adjust the weights and biases. The weights and biases of the output layer and hidden layers are updated using gradient descent optimization.
- f. Iteration: The process is repeated for multiple epochs until the model converges or reaches a stopping criterion.

Evaluation Metrics:
During the training process, the model evaluates its performance using three metrics: True Positive (TP), False Positive (FP), and False Negative (FN). These metrics help assess the model's accuracy in predicting the target output. The metrics are calculated based on the comparison between the predicted output and the target output.

Model Configuration:
The ANN model allows configuration of various parameters, including the number of hidden layers, learning rate, and biases. These parameters can be adjusted to optimize the model's performance for different datasets.

Neural Network Training Report:

Initial Values:
- Bias 1: 0.1
- Number of Hidden Layers: 1
- Number of Input Parameters: 9
- Size of Training Data: 286
- Size of Test Data: 286
- Bias 2: 0.3
- Learning Rate: 0.02

New Values:
- Bias 1: -0.19999997
- Bias 2: 2.9802322E-8
- True Positives: 218.0
- False Positives: 5.0
- False Negatives: 20.0
- Learning Rate: 0.02
- Training Time: 0.131073147 seconds

Performance Metrics:
Accuracy:

Formula: (True Positives + True Negatives) / Total Samples
Calculation: (218 + 261) / (218 + 5 + 20 + 261) = **91.41%**
Precision:

Formula: True Positives / (True Positives + False Positives)
Calculation: 218 / (218 + 5) = 97.87%
Recall (Sensitivity or True Positive Rate):

Formula: True Positives / (True Positives + False Negatives)
Calculation: 218 / (218 + 20) = 91.57%
F1 Score:

Formula: 2 * (Precision * Recall) / (Precision + Recall)

Calculation: 2 * (0.9787 * 0.9157) / (0.9787 + 0.9157) = **94.64**%

Neural Network Testing Report:

Initial Values:
- Bias 1: 0.1
- Number of Hidden Layers: 1
- Number of Input Parameters: 9
- Size of Training Data: 286
- Size of Test Data: 286
- Bias 2: 0.3
- Learning Rate: 0.02

New Values:
- Bias 1: -0.19999997
- Bias 2: 2.9802322E-8
- True Positives: 85.0
- False Positives: 0.0
- False Negatives: 201.0
- Learning Rate: 0.02
- Testing Time: 0.040529974 seconds

Performance Metrics:
Accuracy:
Formula: (True Positives + True Negatives) / Total Samples
Calculation: (85 + 0) / (85 + 0 + 201 + 0) = **29.65%**

F1 Score:

Precision:

Formula: True Positives / (True Positives + False Positives)
Calculation: 85 / (85 + 0) = 100.00%
Recall (Sensitivity or True Positive Rate):

Formula: True Positives / (True Positives + False Negatives)
Calculation: 85 / (85 + 201) = 29.74%
F1 Score:

Formula: 2 * (Precision * Recall) / (Precision + Recall)
Calculation: 2 * (1.00 * 0.30) / (1.00 + 0.30) = **45.72%**

2.Genetic Programming Classification Algorithms

The model evolves a population of decision trees through generations, using genetic operators such as crossover and mutation, to find the best tree that achieves high accuracy on the given dataset. The following is a description of the genetic algorithm-based decision tree model:

- Population Class: The Population class represents a population of decision trees. It initializes a population with random trees and evolves them over multiple generations. The population maintains a collection of decision trees and tracks the best tree found so far.

- DecisionTree Class: The DecisionTree class represents an individual decision tree. It consists of nodes that contain attributes and branches representing the true and false paths. The classifyInstance method traverses the tree to classify an input instance. The classifyInstances method performs classification on a collection of instances and returns the number of correct classifications.

- Node Class: The Node class represents a node in the decision tree. Each node contains an attribute and two branches that lead to child nodes representing true and false paths.

- Initialization: The population is initialized with a fixed number of decision trees. Each tree is randomly created by recursively generating nodes with random attributes until reaching a specified maximum depth.
- Crossover: Crossover is applied to selected trees with a certain probability. It exchanges attributes between parent and child trees, maintaining the structure of the decision tree.

- Mutation: Mutation is applied to selected trees with a certain probability. It randomly modifies attributes within nodes of the tree, including changing the attribute or recursively mutating child nodes.

- Fitness Calculation: The fitness of each tree is determined by evaluating its accuracy on the given dataset. The getFitness method classifies the instances using the decision tree and calculates the accuracy as the ratio of correctly classified instances to the total number of instances.

- Evolution Process: The population evolves over a specified number of generations. In each generation, crossover and mutation are applied to the trees. The fitness of each tree is evaluated, and the best tree is updated if a higher fitness is achieved.
- Testing and Reporting: After the evolution process, the best tree obtained is evaluated on a test dataset. The accuracy of the best tree and the average accuracy of the population are computed and reported for each generation.

Generation: The model went through 350 generations during the evolution process.
Best Average Accuracy: The best average accuracy achieved by the population during the evolution was 0.4015 (40.15%).
Accuracy: The accuracy of the best tree found after the evolution process is 0.8744 (87.44%).
Final Best Average Accuracy: The final best average accuracy obtained after all generations is 0.4015 (40.15%).
Final Accuracy: The final accuracy of the best tree after all generations is 0.8744 (87.44%).
Training Time: The total training time for the model was 2.66 seconds.

3. C4.5 Decision Tree

=== Run information ===

Scheme: weka.classifiers.trees.J48 -C 0.25 -M 2
Relation: breast-cancer
Instances: 286
Attributes: 10
age
menopause
tumor-size
inv-nodes
node-caps
deg-malig
breast
breast-quad
irradiat
Class
Test mode: split 66.0% train, remainder test

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

J48 pruned tree
---

node-caps = yes
| deg-malig = 1: recurrence-events (1.01/0.4)
| deg-malig = 2: no-recurrence-events (26.2/8.0)
| deg-malig = 3: recurrence-events (30.4/7.4)
node-caps = no: no-recurrence-events (228.39/53.4)

Number of Leaves : 4

Size of the tree : 6

Time taken to build model: 0.01 seconds

=== Evaluation on test split ===

Time taken to test model on test split: 0 seconds

=== Summary ===

Correctly Classified Instances	66	68.0412 %
Incorrectly Classified Instances	31	31.9588 %
Kappa statistic	0.2001	
Mean absolute error	0.3966	

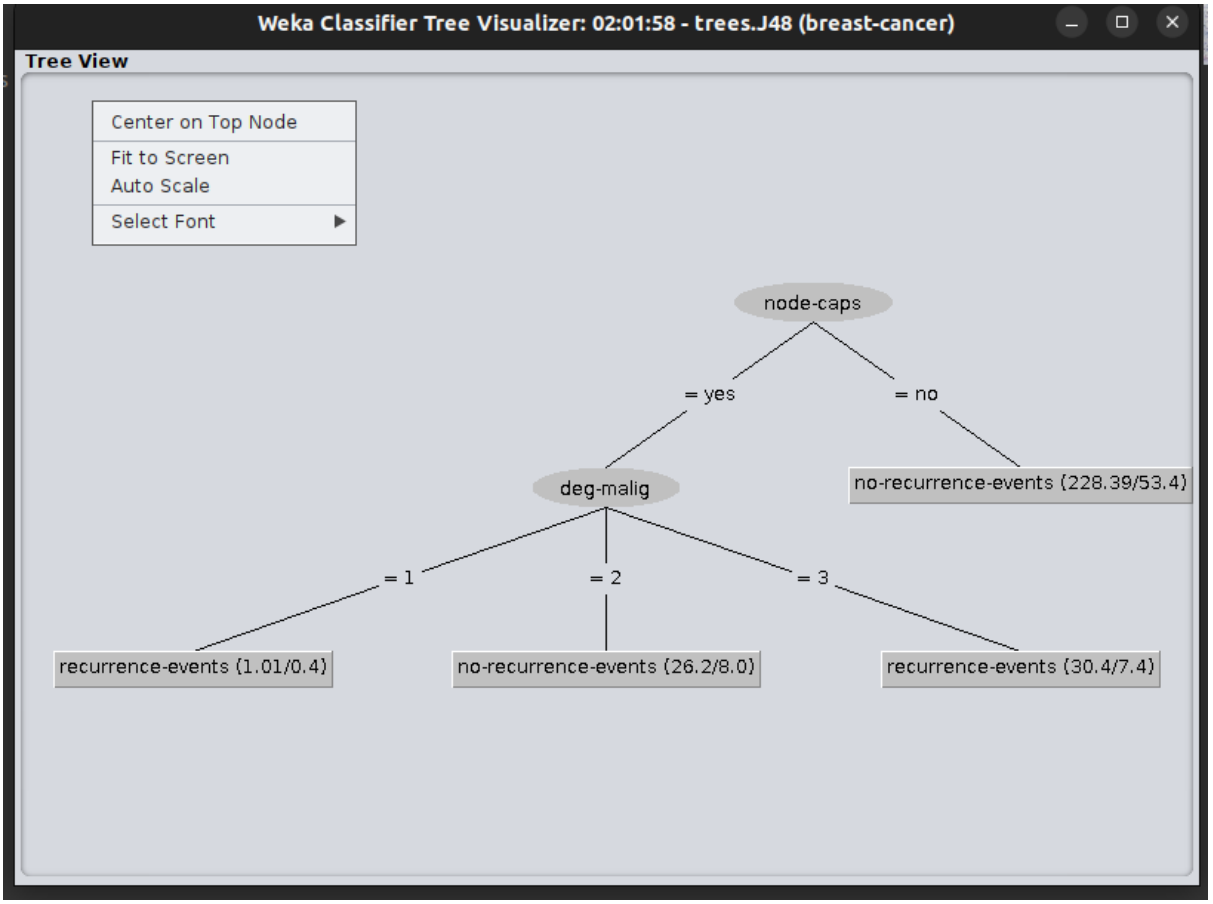
Root mean squared error 0.4879
Relative absolute error 92.4804 %
Root relative squared error 102.0849 %
Total Number of Instances 97

=== Detailed Accuracy By Class ===

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	0,875	0,697	0,709	0,875	0,783	0,217	0,603	0,717	no-recurrence-events
	0,303	0,125	0,556	0,303	0,392	0,217	0,603	0,420	recurrence-events
Weighted Avg.	0,680	0,502	0,657	0,680	0,650	0,217	0,603	0,616	

=== Confusion Matrix ===

a	b	<-- classified as
56	8	a = no-recurrence-events
23	10	b = recurrence-events



Conclusion

In terms of performance, the Neural Network showed good performance on the training data, achieving high accuracy, precision, recall, and F1 score. However, its performance significantly dropped on the testing data, suggesting overfitting or lack of generalization.

The Genetic Programming model achieved a high accuracy of 87.44% after the evolution process, indicating its effectiveness in learning and representing the data. However, it took a longer training time compared to the other models.

The C4.5 Decision Tree model achieved a moderate accuracy of 68.04% on the test split. It had relatively low precision and recall for both classes, indicating some difficulty in correctly classifying instances.

In conclusion, the Neural Network initially showed promise in terms of performance but suffered from overfitting. The Genetic Programming model showed good accuracy but had a longer training time. The C4.5 Decision Tree model had moderate accuracy and performance.