Johannes De Clercq
U19046121

Assignment 2 - Report

1) Introduction to the data used

The assignment required that we make use of a heart disease data set. Following the link provided I decided to use the data set called processed.cleaveland.data. It contained 303 data entries each consisting of 13 variables, named later in Genetic Program Breakdown, and 1 classifier, also expanded on in Genetic Program Breakdown. The data was then split into a training and testing set, with the training set containing 80% of the data and thus the testing the remaining 20%.

Before the data could be used for training the distribution of classifiers had to be fixed as the original data was weighted towards 0 with over half the options being it. The problem was fixed by removing 70 elements from the data set that classified to 0, changing the data as follows:

Original	Modified
0: 132	0: 61
1: 43	1: 43
2: 23	2: 23
3: 30	3: 30
4: 11	4: 11
Sample Size: 239	Sample Size: 168

2) Genetic Program Breakdown

The genetic program made use of a decision tree to produce the classifications. Each tree consisted of terminal and functional nodes. The terminal nodes represented each of the possible classifications, being 0, 1, 2, 3, or 4, and the functional nodes representing each of the variables that the data uses: age, sex, chest pain type, resting blood pressure, cholesterol, fasting blood sugar, resting electrocardiographic, maximum heart rate, exercise induced angina, ST depression, ST segment, major blood vessel colored by fluoroscopy, thal and angiographic disease stage. Therefore, our terminal and functional sets are:
Terminal: = (0, 1, 2, 3, 4)
Functional = (age, sex, cp, trestbps, chol, fbs, restecg, thalach, exang, oldpeak, slope, ca, thal)

The fitness function used for this genetic program went through many iterations from raw fitness to accuracy but ultimately the best fitness measurement turned out to be f1-score [1]. The aim of the f1-score is to calculate the precision and recall of a classifier, precision being what proportion of predicted positives were true positives and recall being what proportion of actual positives is correctly classified. The entire formula is:
Precision: TP/(TP+FP)
Recall: TP/(TP+FN)
F1: $2 \times (\text{precision} \times \text{recall}) / (\text{precision} + \text{recall})$

Johannes De Clercq
U19046121

Fitness = max(F1)

Tournament selection was chosen as the selection method. It works by creating a tournament in which a certain number of population members are pitted against each other, using their fitness scores, and the best one chosen. This tournament is done with replacement so that each time the tournament is held each of the population has a probability to be selected, with the best fitness scores having the highest probability of being selected. The selected members are then passed on to the genetic operators

3 different genetic operators were chosen: Crossover, mutation, and reproduction. The crossover operator takes two members of the population, making use of the selection method to select them, and swaps a subtree between each of the members. With the aim of exploiting the local search space in which the two members lie. A crossover chance is used to determine how many times the crossover operator will be called on the population.   The mutation operator makes use of the selection method to select a member of the population. Once a member has been selected a random subtree in the population is chosen, it could also be the root, and builds a new tree from that node. This way a large portion of the tree remains the same, but variation is also placed into it. Since we have a stagnant population size our mutation chance, chance of mutation happening, is applied to the len(population) * (1-crossoverChance). This way we maintain the population size. Since we make use of a mutation chance, we inherently end up reproducing a large portion of the population that mutation was not applied to.

Finally, the 3-termination criterion were used. The first and simplest was a hard cap of 100 generations as it was found that most runs produced a convergence between 30 – 50 but as there were other termination criterion a prudent approach was used. The second criterion is if the fitness score gets above 0.6 as the algorithm struggles to produce a result above 0.5, I consider it to be sufficient. The final criterion is a fitness stagnation. If the algorithm has reached 50 generations or higher, a check is made to determine if the last 5 highest fitness scores are the same and if they are we have reached an optimum and can terminate. The reason I wait 50 generations is to give the mutation operator a chance to explore the optimum we have encountered rather than immediately presuming we have stagnated.

3) Results

The results obtained are shown below, they are split into two tables. The first one represents the training results and the second represents the testing results

Training results (all values are percentages)

All training accuracies:	[48.214285714285715, 50.0, 47.61904761904761, 52.38095238095239, 50.595238095238095, 50.0, 50.0, 49.404761904761905,

	51.19047619047619, 45.23809523809524]
Average Accuracy:	49.46428571428572
Standard Dev:	2.012691214748534
Best Accuracy:	52.38095238095239

Testing Results (all values are percentages)

All training accuracies:	[57.377049180327866, 47.540983606557376, 45.90163934426229, 47.540983606557376, 44.26229508196721, 59.01639344262295, 45.90163934426229, 42.62295081967213, 55.73770491803278, 6.557377049180328]
Average Accuracy:	45.24590163934426
Standard Dev:	14.768258738018682
Best Accuracy:	59.01639344262295

4) Conclusion:

In conclusion the program proved to be semi-reliable at classification with its major complication coming from the vast fitness landscape that must be navigated to achieve a global optimum. A potential fix to the problem is a fitness function like what was discussed for this algorithm but making use of the I(DT) portion of Nikolaev and Slavov, 1998 [2] fitness function. I(DT) = nf + nz + (nf * log(f) + (nz * log(l)), where nf-functional nodes in DT; nz-leaves in DT; f-possible functions; l- leaf classes; The aim of this formula is to work out the complexity of a tree. We could therefore combine this complexity formula with a f1-score to produce a minimizing fitness function that will produce the best accuracy alongside becoming more compressed rather than expanding.

5) Bibliography

1) Shmuili, B., 2022. *Multi-Class Metrics Made Simple, Part I: Precision and Recall*. [online] Medium. Available at: <https://towardsdatascience.com/multi-class-metrics-made-simple-part-i-precision-and-recall-9250280bddc2> [Accessed 9 May 2022].
2) Nikolaev, N. and Slavov, V., 1998. Inductive Genetic Programming with Decision Trees. *Intelligent Data Analysis*, 2(1), pp.31-44.