JH2199 – Report

## Implementation

### Achieved

In this assignment I was able to create a modelling.java script which modified the Credit.arff file to match the criteria. This included analysing the data and dynamically allocating ranges to each attribute and then setting the number of categories based on this. Then finally updating the credit.arff file with the correct header information about attributes and correct data. I was also able to implement a functioning GA which would return classification rules. To do this I initialised the population with random Booleans based on the population size and number of Bits which was the amount of attributes. Additionally, I was able to implement a tournament selection function which selected the best individual from a group of TOURNAMENT\_SIZE which was a GA parameter. As well as this I implemented a fitness function based on sensitivity and specificity rules which checks whether instances are covered by rules and meet the target. I was also able to extend the rule by adding to the toBinary and Covers methods. The to binary method will encode the inputted values which make up an attribute, it will then check if this attribute is a numeric attribute. If it is a numeric attribute, then an extra bit is added on to the end based on where the value falls into order of the categories. Finally, I was able to extend the covers method to allow it to process attributes with an extra bit. The method does this by extracting the value of the rule and instance and then seeing if the operator bit is applicable to that instance. If it is then the count is incremented and match = true, nominal attributes were then processed as normal.

### Difficulties

During the implementation process of this assignment, I encountered multiple setbacks. Initially I found some issues when developing the modelling.java. I found it hard to ensure that the outputted file was correct as it was hard to check that the data had been put into the right categories. Despite this I overcame these quickly however also found issues with changing the Dataset.java file. One of these issues was the processInstance method accounting for the extra bits while being initialised with the length of original data. Unfortunately, I was unable to find a dynamic solution to this problem therefore if the amount of numeric attributes changed the length would have to be adjusted manually. This would be something I would look to improve upon if I did the assignment again. Mainly I found difficulties with changing the covers method. Initially I didn’t understand how the rule operator would be compared to the instance, whether it would be compared based on the operator or the operator applied to the actual numeric value of the category. This was something I got wrong which caused me to spend additional time on the section however I eventually found a solution where the instance value is compared to the ruleValue. Despite this I would like to optimise this method as I believe it could be improved.

## Experiments and Results

Once I had completed the GA tasks and extended the rule, I tinkered with multiple parameters to test how the algorithm would react. These included the Population Size, Generations, Mutation and Crossover probabilities, tournament selection size and the number of categories for each attribute. Mutation and Crossover seemed to have little effect on the results with specificity only changing by 0.01. Similarly increasing the population over 1000 seemed to have little effect.

Optimal results

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Tournament Size | Population Size | Mutation | Crossover | Generations | Best Specificity | Best  Sensitivity |
| 9 | 1000 | 0.1 | 0.9 | 100 | 0.94 | 0.79 |

Experimental

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Tournament Size | Population Size | Mutation | Crossover | Generations | Best Specificity | Best  Sensitivity |
| 21 | 1000 | 0.1 | 0.9 | 100 | 0.92 | 0.78 |
| 9 | 1000 | 0.15 | 0.85 | 100 | 0.93 | 0.79 |
| 9 | 2000 | 0.1 | 0.9 | 100 | 0.92 | 0.78 |

In conclusion, the optimal results where achieved with a tournament size of 9 and population size of 1000 mutation and crossover at 0.1 and 0.9 and generation at 100.