COMP6560 – Assignment Report
"Predicting Price Movements with Genetic Algorithms"

Edward-Christian Marin (em632)

1. The Task

The task of this assignment is to predict price movements (more specifically, the increase in 14 days) using a Genetic Algorithm with the input being a few technical indicators (such as Simple Moving Average), derived from the daily closing prices of my data. An example of a data point is: [0, 1, 1, 0, 1, 1, 0] – the first six bits are the technical indicator outputs, and the last bit is the increase in 14 days (actual value).

2. The Implementation

Using the data mentioned above, I managed to write a genetic algorithm that returns an individual with which you can predict the increase of a price in 14 days quite accurately (with standard GA parameters, always having 50%+ success rate).

I decided to represent my individuals as weights ranging from -1.0 to 1.0 (double numbers); each weight representing a technical indicator output. An example of an individual: [0.000120, 0.544450, -0.232220, 1.0, 0.4555230, 0.112210]. I used the standard one-point mutation (mutate by randomising the double value of the weight) and one point crossover as my genetic operators. The GA uses the tournament selection for its parent selection process and the termination criteria is a set number of generations.

To improve the evaluation of my GA, I divided my input data into training (70%) and testing (30%). The GA uses the training data to generate a good individual, which is then used to predict the increase in 14 days of the remaining 30% test data.

The fitness for this GA works as follows: it starts at 0.0 and increases in value as it predicts wrong (so the lower the fitness, the better the individual).

One of the design decisions that I have made early on, was to introduce a kind of confidence factor into my fitness score during the training phase. This would discourage biases to happen towards particular technical indicators, and always produce a well performing individual.

The formula for my fitness is:

$$F = F + (prediction - Y)^2$$

, where *prediction* ∈ [0.0, 1.0] (double data type), and Y is the increase in 14 days from the input. This way, two individuals making identical predictions will have different fitness values based on how confident they were in that prediction (coming from their weight values).

The formula for predicting an output based on the weights is:

$$prediction = S(\sum_{i=0}^{n}(w_i * x_i))$$

, where n = BITS – 1, w is the weight of that technical indicator and x is the value of the technical indicator from the input data (but false is represented as -1 instead of 0). Also, S is equivalent to the Sigmoid function:

$$S(x) = \frac{1}{1 + e^{-x}}$$

This gives the predictions more value, as there is a confidence factor included in their value (predictions range from 0.0 to 1.0, so the lower/higher the value, the more confident it is in its prediction). I have learnt these concepts from further reading on artificial neurons and weight averaging.

To predict on real data, instead of using the equation for the fitness function of the training data, I simply compare the *prediction* to 0.5 and generate a true or false value based on that:

$$realPrediction = \begin{cases} 1, & if\ prediction \geq 0.5 \\ 0, & if\ prediction < 0.5 \end{cases}$$

3. Experimental Results

This algorithm performs really well, almost always generating an individual that achieves over 50% success rate. Here is a table highlighting several runs (5 runs each row, averaged):

Population Size	Max Generations	Tournament Size	Training Fitness	Rate of Success (testing data)
100	50	3	217.6151465851801	56.25%
50	25	3	219.67130067957112	54.6875%
50	25	5	217.34288464596585	55.6770833%
500	100	5	216.81029398344407	56.5104166%
1000	50	10	216.8195355339737	57.8125%
20	15	2	228.36771965235286	49.7395833%

It is clear that the higher the population, max generations or tournament size are, the more consistent the predictions get. It seems that there is an early "fitness improvement" during the first 2-3 generations, where it decreases rapidly to a value <220.0, after which it starts decreasing at a decimal level. This is directly influenced by the tournament size, so a sort of "artificial elitism" is being introduced with a bigger tournament size.

4. Personal reflection ("Would I do anything differently?")

I would have started with the weight solution from the start if I were to re-do. When I started working on this assignment, my individual had a quite simple 0 and 1s representation, which was not giving a satisfactory rate of success. I spent a bit too much time on that, which could have been used on reading more about how artificial neurons work, so that I could apply concepts from that topic to this assignment.