## 1. Integrity Statement

For the 24-hour duration of this test, I, Hasnain Cheena, confirm that I will not discuss the content of the test with anyone else. I will not give any assistance to another student taking this test. I will not receive any assistance from any person or tutoring service. I will not post any information about this test on the Internet.

## 2. Overfitting

Tree 1
Tree 1 has been underfit. This is because it contains a single node that only outputs the majority class. The tree has not captured the underlying patterns in the dataset.

Tree 2
Tree 2 is a good fit. It is a relatively short tree (max depth of 2) with reasonable accuracy. Moreover, each leaf node ends up with a decent proportion of the data to assess when compared to an overfit tree like Tree 3. Therefore, the model seems to be capturing the underlying pattern in the data and not modelling the noise.

Tree 3
Tree 3 seems to be overfit. The amount of data fit on trees 2 and 3 is the same but tree 3 is using more attributes thus increasing the tendency to overfit. Further, the number of training examples that tree 3 is attempting to correctly classify using the additional variables (variable 4 and variable 5) is relatively small indicating it is possibly modelling the noise.

Tree 4
Tree 4 is a good model but there is still a chance it has been overfit. Tree 4 is the same as Tree 3 except has been fit on a greater volume of data. Therefore, the chance of overfitting the tree has decreased.

## 3. Bias

Machine learning algorithms use bias to learn. The bias these algorithms have is called inductive bias. Inductive biases are the assumptions a learning algorithm makes in-order to generalize and predict unseen examples. In comparison, underfit models typically have a large bias on the training data. This is because underfit models are not able to capture the underlying pattern in the training data. Therefore, underfit models have not made the correct/enough assumptions to model the pattern in the data. As a result of this they have a large bias and their predictions are on average relatively far from the true value.

## 4. ANN – Activation Functions

| **Activation Function** | **Pro's** | **Con's** |
|---|---|---|
| Sigmoid | 1. It is a non-linear function and so it can be used to represent non-linear target functions.<br>2. It is differentiable and continuous therefore can be used in backpropagation.<br>3. The gradient is smooth preventing large jumps in the output.<br>4. The output of each neuron is bound between 0 and 1. This means the output of each neuron is normalized. | 1. Computationally expensive to calculate<br>2. Issue with vanishing gradients. For very high or low input values, derivative becomes close to 0. This means the network it is harder for the network to learn and train.<br>3. As a result of the shape of the sigmoid function the |

| | | |
|---|---|---|
| | | output struggles to reach values 0 and 1. |
| ReLu | 1. Computationally cheap as no complex math is involved.<br>2. Converges faster when using SGD than the other activation functions.<br>3. Does not have the vanishing gradients issue.<br>4. Can be used with backpropagation<br>5. It is a non-linear function and so can be used to represent non-linear target functions. | 1. Has the dying ReLu issue where the neurons that use ReLu become inactive and output 0 for any input. Therefore, the network cannot correctly perform backpropagation.<br>2. When the input is 0 the gradient is undefined. |
| Linear function | 1. Allows multiple outputs and therefore can be used for multi-category classification problems | 1. Cannot be used in conjunction with backpropagation as the derivative is constant and thus has no relation to the input.<br>2. A neural network using linear activation functions is simply a linear regression model. Therefore, it has limited prediction ability as it cannot be used to model non-linear target functions. |
| Step function | 1. Computationally inexpensive<br>2. Useful in binary classification problems | 1. Not differentiable and so cannot be used in the backpropagation algorithm.<br>2. Cannot be used in multi-class classification. |

The models each of these activation functions can learn:

- Step function can be used to learn binary classification models
- Linear function can be used in traditional linear regression models and can be used to learn binary/multiple category classification models
- The Sigmoid function can be used to learn non-linear functions
- ReLu can be used to learn non-linear functions

## 5. ANN and Local Minima

Getting stuck in a local minima could be either underfitting or overfitting but it is more likely to be underfitting. Typically, ANN's with lower dimensional error surfaces get stuck in local minima. ANN's with lower dimensional error surfaces have a lower number of weights (smaller network) and lower degrees of freedom. Because of the reduced degrees of freedom, the ANN is most likely to not have modelled the underlying patterns in the data, thus underfitting.

## 6. GA

Overcrowding is when too many individuals with the same/similar genetic makeup exist in the population, lowering the overall genetic diversity. Overcrowding occurs because individuals with higher fitness are more likely to pass their genetic makeup on into the next generation. This creates the situation where multiple high fitness individuals that are similar can pass on into the next generation and also have the ability to cross-over with one another. As a result, offspring are created that are the same/similar to the parents. Overcrowding is a case of early convergence and more likely

to cause underfitting. This is because the algorithm is stuck in a local minimum and not able to capture the underlying pattern in the data.

### 7. Particle Swarms

Particle swarm optimisation uses one-way information sharing. In this method only the best information is shared between particles and all particles have the ability to communicate with one another. In comparison within genetic algorithms information is shared in a multi-way approach. Parents share genetic information with each other only and there is a chance that poor genetic information is passed to the offspring.

Particle swarm optimisation and genetic algorithms have in common:

- Both algorithms start with a randomly generated population
- Both algorithms use fitness values to evaluate the population
- Both update the population iteratively and search for the optimal solution using random techniques
- Both algorithms do not guarantee the solution reached will be the global optimal
- Both are based on principles within nature

Particle swarm optimisation and genetic algorithms differ:

- Particle swarm does not use genetic operators like cross-over and mutation
- Particle swarm uses one-way information sharing versus genetic algorithms use multi-way information sharing
- Particle swarm is relatively easy to implement, and tune compared to genetic algorithms
- Particle swarm typically converges quicker than genetic algorithms

### 8. Practical Machine Learning

The first step is to determine task, training experience and performance metric. In our case the task is to play connect 4 and the performance metric is percentage of games won. The experience is generated by playing games against itself.

The second step is to determine the target function. In our case the selected target function (shown below) maps the legal connect 4 board moves to a real number. The idea here is better moves will have higher scores associated with them.

$$V: B \rightarrow R$$

The third step is to select a model to represent the target function. I selected a linear regression model (shown below) that contains the following attributes:

1. Number of red chips - $x_1$
2. Number of yellow chips - $x_2$
3. Number of potentially winning red positions (3 red chips in a row) – $x_3$
4. Number of potentially winning yellow positions (3 yellow chips in a row) – $x_4$
5. Number of paired red positions (2 red chips in a row) – $x_5$
6. Number of paired yellow positions (2 yellow chips in a row) – $x_6$

$$V(b) = w_0 + w_1x_1 + w_2x_2 + w_3x_3 + w_4x_4 + w_5x_5 + w_6x_6$$

The fourth step is to choose a learning algorithm. The learning algorithm in this case will be the least mean squares algorithm that uses gradient descent to learn.