This report details the ideas and approaches that support the computational method that I used to improve upon the dice game. One of the most highly regarded methods in reinforcement learning is the value iteration algorithm (Russell and Norvig, 2016). This is why I decided to choose this method to centre my approach around. The following report will explore in more detail why the aforementioned method was chosen. Touching on the implementation of the algorithm, how it was optimised, what the results were, and lastly suggestions on what could be implemented in future iterations.

The value iteration method is the basis for my algorithm because of it's strength in solving Markov Decision Processes which the dice game could be considered as. The method strikes has a medium complexity and strikes a fine balance between finding an optimal policy for the game and efficiency in terms of computation.

Iterative operation is at the core the value iteration algorithm and it updates each state value in the dice game until a level of error tolerance is met (Russell and Norvig, 2016). The criteria for the tolerance threshold can be considered as follows:

1. Initalise all state values to 0.
    a. This is implemented in the perform_value_iteration method within for state in self.game.states: self.state_values[state] = 0.
2. Calculate the expected return of all the possible actions for each state.
3. Updated the state value to the highest possible return and adjust the the policy in accordance.
    a. Each state's best action is in self.best_actions[state] = action, which is later used in the play method.
4. Iterate through steps 2 and 3 continuously until the greatest alteration in state values is lowered below a specific predefined tolerance threshold.
    a. This criteria is in the while loop of the perform_value_iteration method, where max_change < self.convergence_threshold is used to decide when the loop stops.

The ancipated return from each state, when all actions are contemplated, is calculated to be the sum of the immediate reward and the discounted value of the succeeding state, averaged across all possible subsequent states. This process is repeated iteratively until it gradually coverges on the optimum values and policy for the dice game. The calculation is inside the nested for loops of the perform_value_iteration method, where the reward for each action is based on the current state values and transition probabilities for the game.

The algorithm's overall performance is reliant on the specific tuning and optimisation of hyperparameters. Most notably the discount coefficient and tolerance threshold. The discount coefficient is set at self.discount_factor = 0.9 and was chosen as a factor to balance between rewards of a future and immediate nature. The tolerance threshold is set at self.convergence_threshold = 1e-4 and was picked to ensure a precise value convergence at the same time as conserving a desirable computation duration. Ultimately, both the discount coefficient and tolerance threshold were discovered using trial and error techniques and continually calculating the optimal balance between efficiency in computation and performance.

An optimal policy was found via the value iteration algorithm as the method easily negotiated the dice game's state space. In more realistic, complex and human-like environments it may not fair so well as the model it operated under assumed a perfect production of the game. A way to make the model more complex would be to test and deploy model-free methods for reinforcement learning within my algorithm such as SARSA or Q-Learning, as they do not need the model of the game to be perfect (Sutton and Barto, 2018). Moreover, an extension of the algorithm to run increasingly difficult and complex scenarios of games, while implementing optimisation methods of elegant design may boost performance and broaden understandings of the most optimal dice game tactics.

References:

- Russell, S. and Norvig, P., 2016. Artificial intelligence: a modern approach. Malaysia; Pearson Education Limited.
- Sutton, R.S. and Barto, A.G., 2018. Reinforcement Learning: An Introduction. MIT press.