**Introduction**

*Materials*

This project consists of two components:

1. This Observable notebook, focused on the theory, meaning, and interpretation of a reinforcement learning model designed to learn Blackjack.
2. A MATLAB live script showing how the model works and showing sample data, housed in a Github repository with supporting material.

Importantly, the MATLAB live script depends on MATLAB (version 2020b or later) and its built-in “Statistics and Machine Learning” and “Curve Fitting” toolboxes, as well as their dependencies. If you have not yet downloaded MATLAB, it is available to University of Victoria students and faculty ([through this link](https://www.uvic.ca/systems/support/computerssoftware/softwaredistribution/matlab.php)), where you can download the software and select the required toolboxes during the installation process. If you forget, they can be added through the add-on explorer, or you will be prompted to download them when you try to run the script without them installed.

Once that is complete, simply download the Github repository linked here, open the “Learning Blackjack.mlx” script in MATLAB, and click “Run”. MATLAB live scripts show code and accompanying text explanations on the left of the screen and selected outputs from that code on the right of the screen (although, this view can be changed in the “View” tab).

*Blackjack Rules*

In Blackjack, players are dealt two cards and play opposite a ‘dealer’ opponent. In each ‘hand’, the player views their two cards and determines whether they want to ‘Hit’ for additional cards or ‘Stay’ with their current cards to outscore the dealer. However, if their score exceeds 21 they lose - this is called a 'Bust'.

Cards are worth as many points as their numbered value, face cards are worth 10 points, and aces are worth 1 or 10 points depending on which is better for the player’s score. Consider the example below - the player is dealt a 5 and 3, which means their score is 8. They should probably 'Hit' for another card - they cannot 'Bust' because there isn't a card that would put them over 21, and the dealer is likely to have a larger score than 8.

A close-up of a card game

AI-generated content may be incorrect.

Based on these rules, an "optimal", but not guaranteed, strategy emerges. By considering the score of the player's hand and how many cards in a given deck would cause the current hand to 'Bust', the player can weigh how valuable it is to 'Hit' or 'Stay'. Importantly, Blackjack rules vary from casino to casino (ex. how many decks are shuffled into play, the dealer's behaviour), but the general pattern of **optimal performance follows a sigmoidal pattern with increasing scores**.

The figure below, replicated from Hewig et al. (2007), demonstrates this well. Players should 'Hit' on every trial with a score of 11 or less since there is no possibility of a 'Bust'. Then, the probability of a 'Hit' should decrease sharply around 15-17 - this is the inflection point of the sigmoid curve and will be important later. Scores less than 15 are less likely to 'Bust' because they need to be dealt a high card, so players should 'Hit'. Scores larger than 17 should indicate a 'Stay' since players will likely outscore the dealer and have a high risk of going over 21 if they 'Hit'.

A graph of a card game

AI-generated content may be incorrect.

However, while this strategy is optimal, it is not perfect. Players can lose with 19, or even 'Bust' and lose on a 12. As such, to "learn" Blackjack, players really must learn this response curve - it is their best route to winning more hands (and of course, more money). Indeed, this figure represents participant behaviour in Blackjack, indicating we can learn this strategy (Hewig et al., 2007; Yu et al., 2009). However, given the trade-off between strategy and random chance in this task, how are participants learning to behave optimally?

The prominent theory on reward processing indicates humans rely on a midbrain reinforcement learning (RL) system to guide behaviour. Specifically, this theory suggests that we compute prediction errors – differences between expected and actual outcomes – and use them to guide action selection (Holroyd & Coles, 2002; Holroyd & Yeung, 2012; Nieuwenhuis et al., 2004). Interestingly, this RL system is still highly active in scenarios where learning is unnecessary, or impossible. Indeed, RL models can explain human behaviour in unlearnable albeit rewarding environments (Cohen et al., 2007; Gershman, 2018; Wilson et al., 2014).

This prompts the question - are humans using an RL system to guide behaviour in scenarios designed for them to lose over time (i.e. the odds of success are random or at best, heavily favoured against them)? Consider Blackjack - are humans using an RL system to guide behaviour in scenarios where learning does not guarantee success? Here, I will attempt to determine if a simple RL model can learn optimal strategy in Blackjack. To test the model, I will compare its score-action values, which are analogous to the probability the model will 'Hit' or 'Stay' on a given hand, to a sigmoidal curve representing optimal Blackjack strategy.

**Methods**

*Computational Model Construction*

I applied a reinforcement learning computational model to the modified Blackjack task in the MATLAB programming environment (Version 9.6, Mathworks, Natick, USA). The model was tasked with determining whether to “Hit” for another card or “Stay” with its current hand based on the summed score of the current trials’ cards, using the using a Softmax equation involving choice values, *Equation 1*:

(1)

Where *aj*represents the two possible actions (Hit or Stay) for a given score, based on the value *V(s, ai)t* where *s* is the initial score of the hand, *ai*is one of two possible actions *i*, on trial *t,* and τ represents temperature, the explorative parameter of the model. The model was then given binary feedback indicating the success of the trial, and changed the value of its decision based on *Equation 2*:

(2)

Where *r* is +1 for success and -1 for failure (a loss or a bust). As such, the value of the action taken for a given score on the next trial where the score is presented was updated by *Equation 3:*

*N*(*s,*1) (3)

Where the update function is the product of the prediction error, δ, the learning rate, α, and a normal distribution function, *N,* centered on the score with a standard deviation of 1. This update function differs from the “traditional” reinforcement learning approach, where only the value of the current choice is changed by the outcome. In most reinforcement learning problems choices are relatively independent, in that the probability of success for one choice is not influenced by the other choice(s). However, choices in the Blackjack paradigm are highly related by score – hitting on a five has a similar performance benefit to hitting on a six. As such, the gaussian update function in this model changes the values of actions for the current score and its neighbouring scores.

Finally, the value of an action for a given score was limited within range of [-1,1], to ensure they did explode past bounds that would constrain the effect of temperature on decision-making.

*Learning and Testing*

Once the model was built, I sought to determine whether the model could learn optimal behaviour in the Blackjack task. To accomplish this, I had the model complete the task with 100, 500, 5,000, and 10,000 trials, 10 times each, and recorded its’ performance. For starting parameters, I set initial values for each choice to 0.5, temperature to 1.5, and learning rate to 0.20.

Recall, optimal strategy in Blackjack indicates that the player’s decision to hit should follow a sigmoidal function with an inflection point at a score of 16. As such, I compared fit the model’s score-action values to the following sigmoidal function (*Equation 4*):

(4)

Where *b* is the slope of the sigmoid curve, *c* is the inflection point, and *U* and *L* represent upper and lower limits, respectively. To fit the model data to this curve, I constrained the state-action values to the difference in value for hitting and staying on scores of 10 to 20. I focused on these higher scores because scores of less than 10 are relatively uncommon in Blackjack. The search space for these fitting parameters is described below in *Table 1*.

***Table 1***

Parameter space for model fitting

|  |  |  |  |
| --- | --- | --- | --- |
| Parameter | Lower Bound | Upper Bound | Initial Value |
| b | -20 | 10 | -8 |
| c | 1 | 20 | 16 |
| U | 0 | 10 | 1 |
| L | -5 | 0 | -1 |

*Note:* Initial values represent the “optimal” Blackjack strategy.

**Results**

*Can a simple Reinforcement Learning model learn Blackjack?*

**Discussion**

*What’s missing from the model?*

The RL model presented here is missing a key component of Blackjack – considering the dealer score in its behaviour. Indeed, the model is agnostic of dealer performance – it just knows the impact of its’ actions on the outcome of each hand. To change this, the RL model’s choice and update functions could be adjusted to expand its score-action values to include the dealer’s score, effectively adding a dimension to the ‘weights’ variable in the model. However, this would greatly increase the trial count needed to train the model – each player score would now have a corresponding range of dealer scores that would need to be experienced to learn score-action values. As the results presented here show, the model was able to learn Blackjack *without* considering dealer scores, albeit only with extensive experience.

Nonetheless, these results cannot determine whether humans use an RL strategy to learn Blackjack, they just demonstrate an RL strategy *can* learn it. Addressing the latter question will prove to be difficult – if the RL model itself cannot learn the strategy with 500 trials of practice, how can we compare it to human data showing learning within 440 trials (Hewig et al., 2007)? Fitting RL models to human data typically involves training the model on a task identical to the version given to humans, and tuning the RL parameters (temperature, learning rate, and initial state-action values) to the human data.

**Bibliography**

**Disclaimer**

Previously, I proposed that this project would focus on fitting a Blackjack RL model to human data. I elected to pursue the project here shortly after piloting the model and seeing just how much experience the model needed to learn the task. In the previously described experiment, participants only completed 200 trials, which is clearly not sufficient for the model to learn. As such, I elected to further explore the construction of the model.