MLPR SEMESTER 4 PROJECT

Reimagining Playing Blackjack Using RL

Dipit Golechha,
Harsh Siroya,
Krishnav Mahansaria

What’s the Problem ?

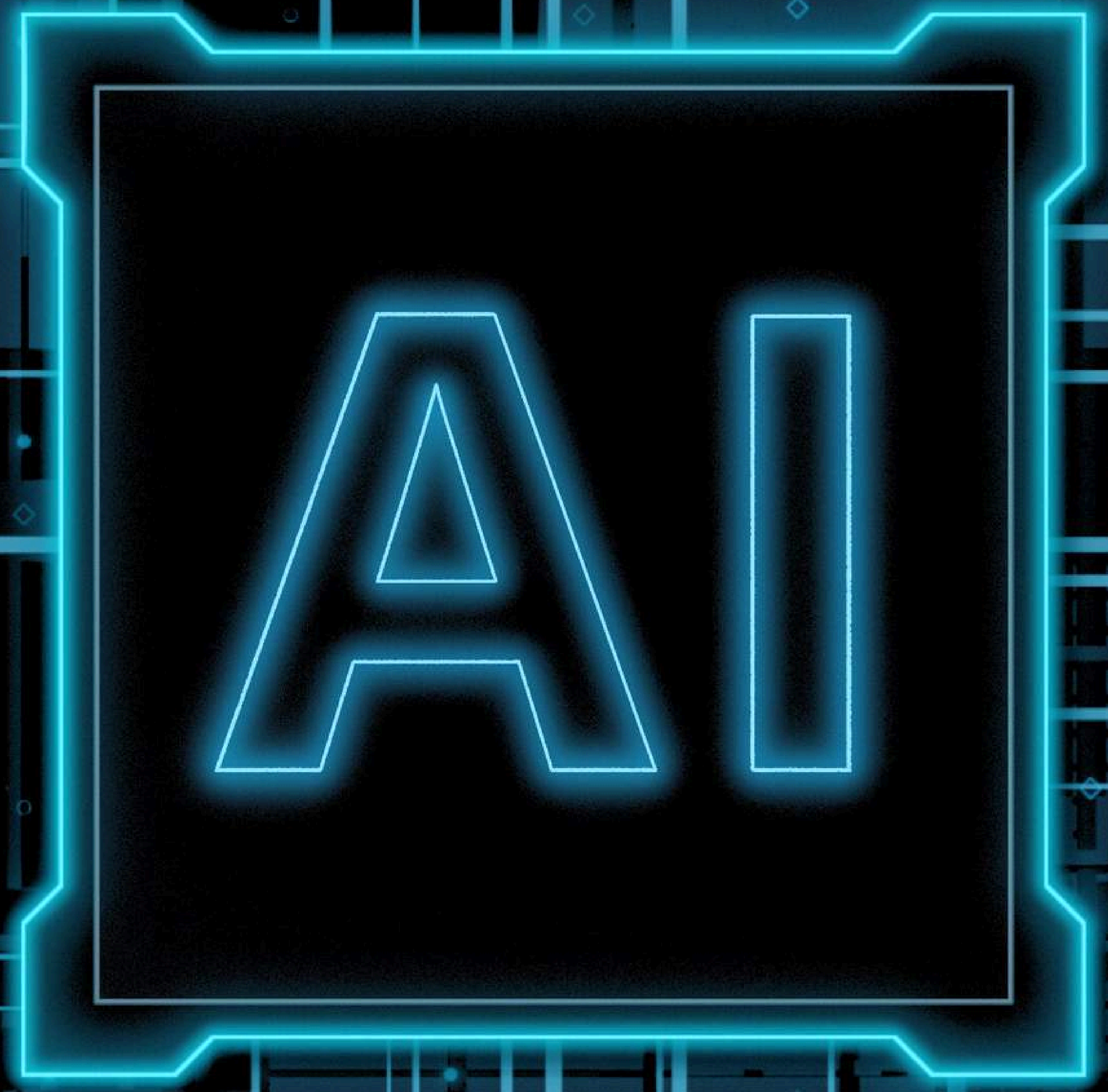
Blackjack combines skill, strategy, and luck, challenging players with the critical choice of 'hit', 'stand', ’double down’ and ’surrender’. Our project aims to automate these decisions using a reinforcement learning model to navigate the game's complexity and potential outcomes, aiming to outperform traditional strategies.

Why This Problem?

This will not only allow us to make playing blackjack easier and increase the odds of winning but will also allow us to explore different games and problems which require similar strategy making and consists of similar probabilistic outcome allowing us to reimagine the use of RL.

Goal

We have developed a stratergy using a sophisticated reinforcement learning model designed to minimise risk in Blackjack.



Literature Review

Previously, we had researched on the methods we can use and came across some papers telling us about the methods' effectiveness in playing blackjack. One study explored Deep Q-learning, comparing Deep Q-Network (DQN) models against a traditional Q-Network (QN) model. The DQN models outperformed the QN model, indicating promise in learning effective strategies for blackjack. However, none of the models discovered the exact optimal strategy, suggesting room for improvement.

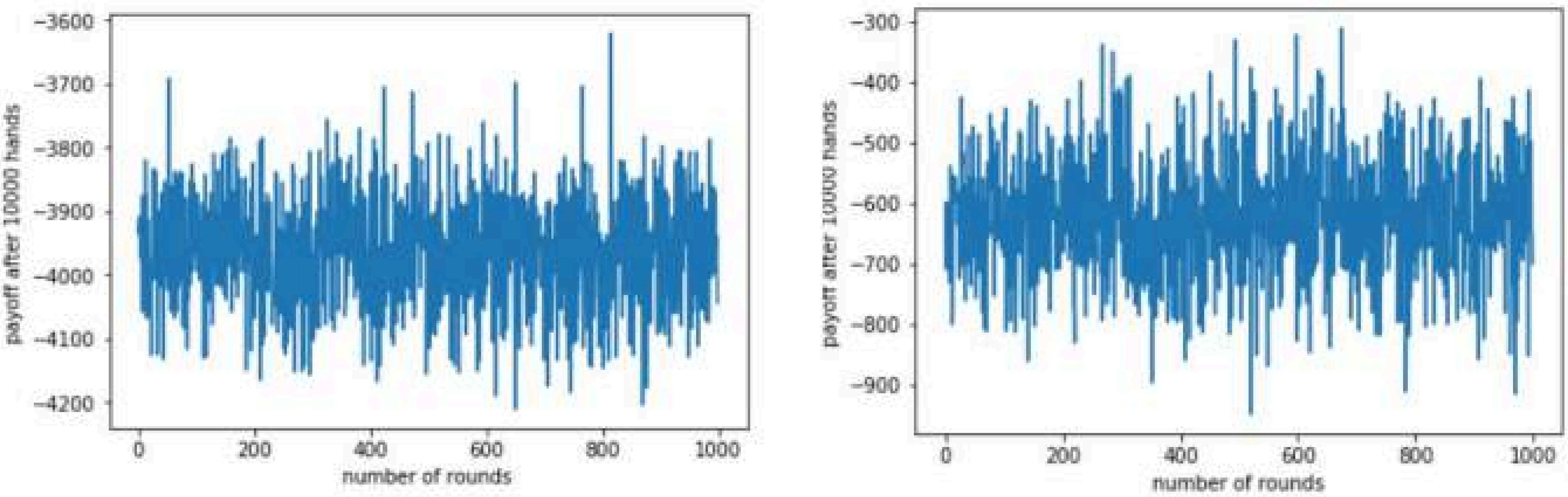
Another paper investigated the application of the Q-learning algorithm, showcasing its potential in approximating an optimal blackjack strategy. Despite not achieving perfect convergence, the reinforcement learning (RL) agent showed significant improvement over random actions, approaching the performance of a basic strategy player.

"Learning to Play Blackjack with Deep Learning and Reinforcement Learning" by Ish Handa

"Applying Reinforcement Learning to Blackjack Using Q-Learning" by Charles de Granville

"Playing Blackjack with Deep Q-Learning" by Allen Wu from Stanford University

"Learning to Play Blackjack with Deep Learning and Reinforcement Learning" by Ish Handa

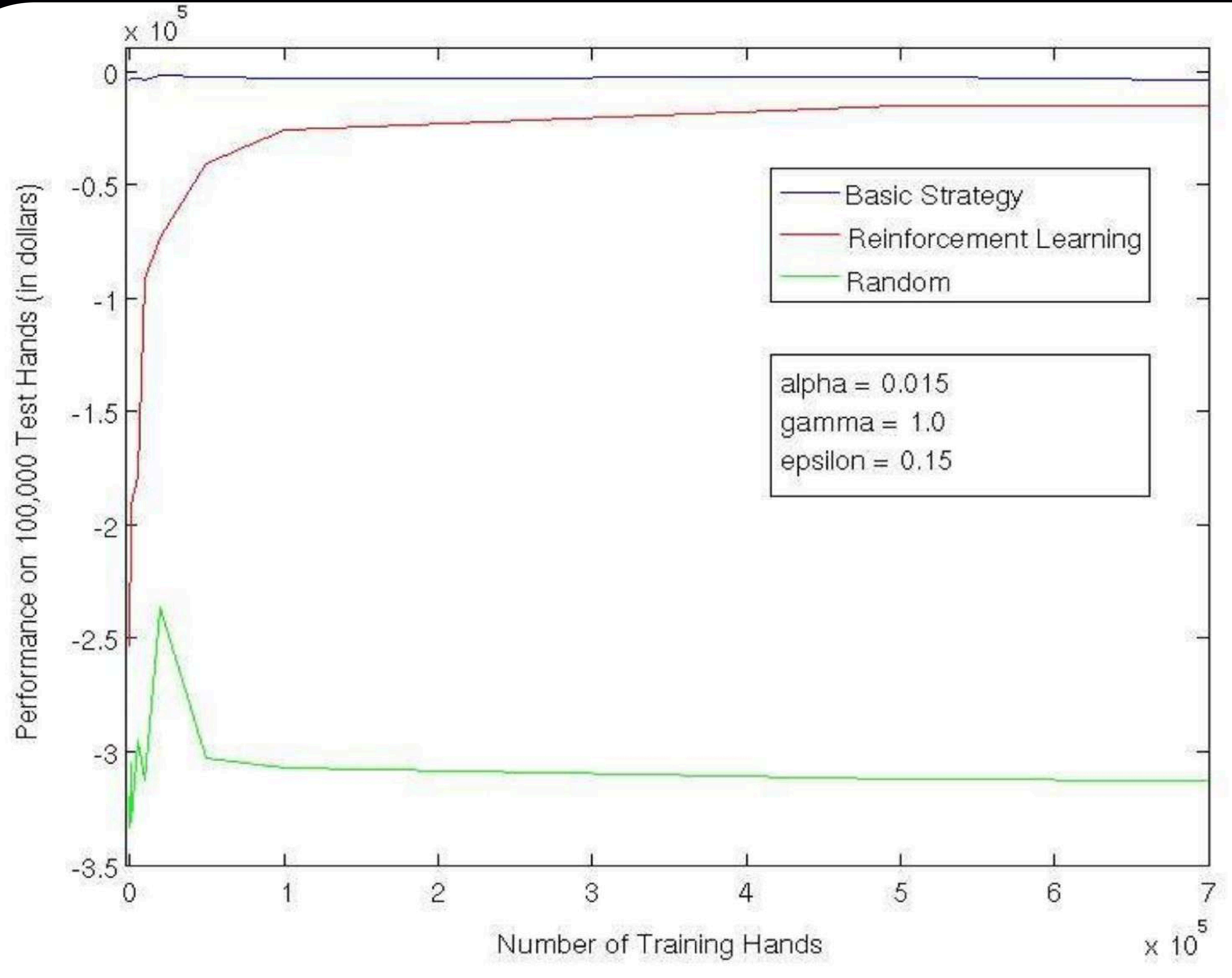


(a) using random strategy

(b) using Edward Thorp's strategy

Figure 6: Average Payoff after 10000 hands per round to estimate range

"Applying Reinforcement Learning to Blackjack Using Q-Learning" by Charles de Granville



"Playing Blackjack with Deep Q-Learning" by Allen Wu from Stanford University

player/dealer	2	3	4	5	6	7	8	9	10	A
5	hit	hit	hit	hit	hit	hit	hit	hit	hit	hit
6	hit	hit	hit	hit	hit	hit	hit	hit	hit	hit
7	hit	hit	hit	hit	hit	hit	hit	hit	hit	hit
8	hit	hit	hit	hit	hit	hit	hit	hit	hit	hit
9	hit	double	double	double	double	hit	hit	hit	hit	hit
10	double	double	double	double	double	double	double	double	hit	hit
11	double	double	double	double	double	double	double	double	double	double
12	hit	hit	stand	stand	stand	hit	hit	hit	hit	hit
13	stand	stand	stand	stand	stand	hit	hit	hit	hit	hit
14	stand	stand	stand	stand	stand	hit	hit	hit	hit	hit
15	stand	stand	stand	stand	stand	hit	hit	hit	surrender	hit
16	stand	stand	stand	stand	stand	hit	hit	surrender	surrender	surrender
17	stand	stand	stand	stand	stand	stand	stand	stand	stand	stand
18	stand	stand	stand	stand	stand	stand	stand	stand	stand	stand
19	stand	stand	stand	stand	stand	stand	stand	stand	stand	stand
20	stand	stand	stand	stand	stand	stand	stand	stand	stand	stand
soft 13	hit	hit	hit	double	double	hit	hit	hit	hit	hit
soft 14	hit	hit	hit	double	double	hit	hit	hit	hit	hit
soft 15	hit	hit	double	double	double	hit	hit	hit	hit	hit
soft 16	hit	hit	double	double	double	hit	hit	hit	hit	hit
soft 17	hit	double	double	double	double	hit	hit	hit	hit	hit
soft 18	double	double	double	double	double	stand	stand	hit	hit	hit
soft 19	stand	stand	stand	stand	double	stand	stand	stand	stand	stand
soft 20	stand	stand	stand	stand	stand	stand	stand	stand	stand	stand

(a) Optimal policy [4]

player/dealer	2	3	4	5	6	7	8	9	10	A
5	hit	hit	hit	hit	hit	hit	hit	hit	hit	hit
6	hit	hit	hit	hit	hit	hit	hit	hit	hit	hit
7	hit	hit	hit	hit	hit	hit	hit	hit	hit	hit
8	hit	hit	hit	hit	hit	hit	hit	hit	hit	hit
9	double	double	double	double	double	hit	hit	hit	hit	hit
10	double	double	double	double	double	double	double	hit	hit	hit
11	double	double	double	double	double	double	double	double	hit	hit
12	double	hit	hit	hit	hit	hit	hit	hit	hit	hit
13	hit	hit	hit	hit	hit	hit	hit	hit	hit	hit
14	double	hit	hit	hit	hit	hit	hit	hit	hit	surrender
15	hit	hit	stand	stand	hit	hit	hit	hit	surrender	surrender
16	hit	hit	hit	stand	stand	stand	surrender	surrender	surrender	surrender
17	stand	stand	stand	stand	stand	stand	stand	stand	surrender	surrender
18	stand	stand	stand	stand	stand	stand	stand	stand	stand	surrender
19	stand	stand	stand	stand	stand	stand	stand	stand	stand	stand
20	stand	stand	stand	stand	stand	stand	stand	stand	stand	stand
soft 13	hit	hit	hit	hit	hit	hit	hit	hit	double	hit
soft 14	hit	hit	hit	hit	hit	hit	hit	hit	hit	hit
soft 15	hit	hit	hit	hit	hit	hit	hit	hit	hit	hit
soft 16	hit	hit	hit	hit	hit	hit	hit	hit	hit	hit
soft 17	double	double	double	hit	hit	hit	hit	hit	hit	hit
soft 18	stand	stand	stand	double	double	stand	stand	stand	hit	hit
soft 19	stand	stand	stand	stand	stand	stand	stand	stand	stand	stand
soft 20	stand	stand	stand	stand	stand	stand	stand	stand	stand	stand

(b) Derived policy

"Playing Blackjack with Deep Q-Learning" by Allen Wu from Stanford University

Network	Policy Score	Similarity	Expected Return
QN	228.5	0.672058824	-0.26189
DQN(3,1)	248.5	0.730882353	-0.1140973
DQN(13,7)	249.5	0.733823529	-0.1218813
DQN(11, 13)	259.5	0.763235294	-0.107426
optimal policy	340	1	-0.095752

DQN(depth (no of layers), width (number of neurons))

Why we chose RL over supervised and unsupervised?

Reinforcement Learning (RL) stands out as the preferred approach for training a blackjack-playing model due to its ability to optimize rewards, adapt to dynamic gameplay, balance risk and reward, learn from game outcomes, and handle uncertainty. Unlike supervised learning, which relies on predetermined optimal moves, and unsupervised learning, which lacks explicit reward signals.

Dataset

- We're implementing a reinforcement learning (RL) model using the OpenAI Gym Blackjack environment accessed via API calls. This environment simulates the classic card game, providing a framework for our RL agent to learn and improve its strategies through interactions with the game.

- Since we are using RL and not ML, the model has to learn through its actions and hence has to be given the freedom of action that is either exploitation or exploration. Whereas if the model was trained using a datset it would have been limited to the decisions and action mentioned in the dataset.

Dataset

- Through the OPen AI Gym Model we were able to get the following state space:

  Players Sum
  Dealers Up Card
  Usable Ace
  Burst or Not
  Reward

- Following action space: Hit, Stand, Double Down
- Number of rounds of blackjack played - 20 Million

Double Q Learning

Double Q-learning is an extension of Q-learning that addresses the issue of overestimation of action values. In traditional Q-learning, a single Q-value is updated based on the maximum Q-value for the next state which can lead to overestimation.

In Double Q-learning, two sets of Q-values are maintained, and during the update step, one set is used to select the best action, while the other set is used to evaluate the value of that action. This helps mitigate the overestimation problem by decoupling action selection from action evaluation.

Hyperparameters

- Alpha (α): Alpha is the learning rate, determining the extent to which new information overrides old information during Q-value updates. It typically ranges between 0 and 1, with higher values indicating more weight given to new experiences.

- Gamma (γ): Gamma represents the discount factor, indicating the importance of future rewards relative to immediate rewards. It ranges between 0 and 1, with higher values prioritizing long-term rewards.

- Epsilon (ε): Epsilon is the exploration rate, governing the balance between exploration and exploitation in the agent's behavior. It determines the probability of selecting a random action instead of the optimal action according to the current policy.

- Number of Episodes: The number of episodes refers to the total number of training iterations or games played by the model. Each episode consists of interactions with the environment, updating Q-values based on observed rewards and transitions. T

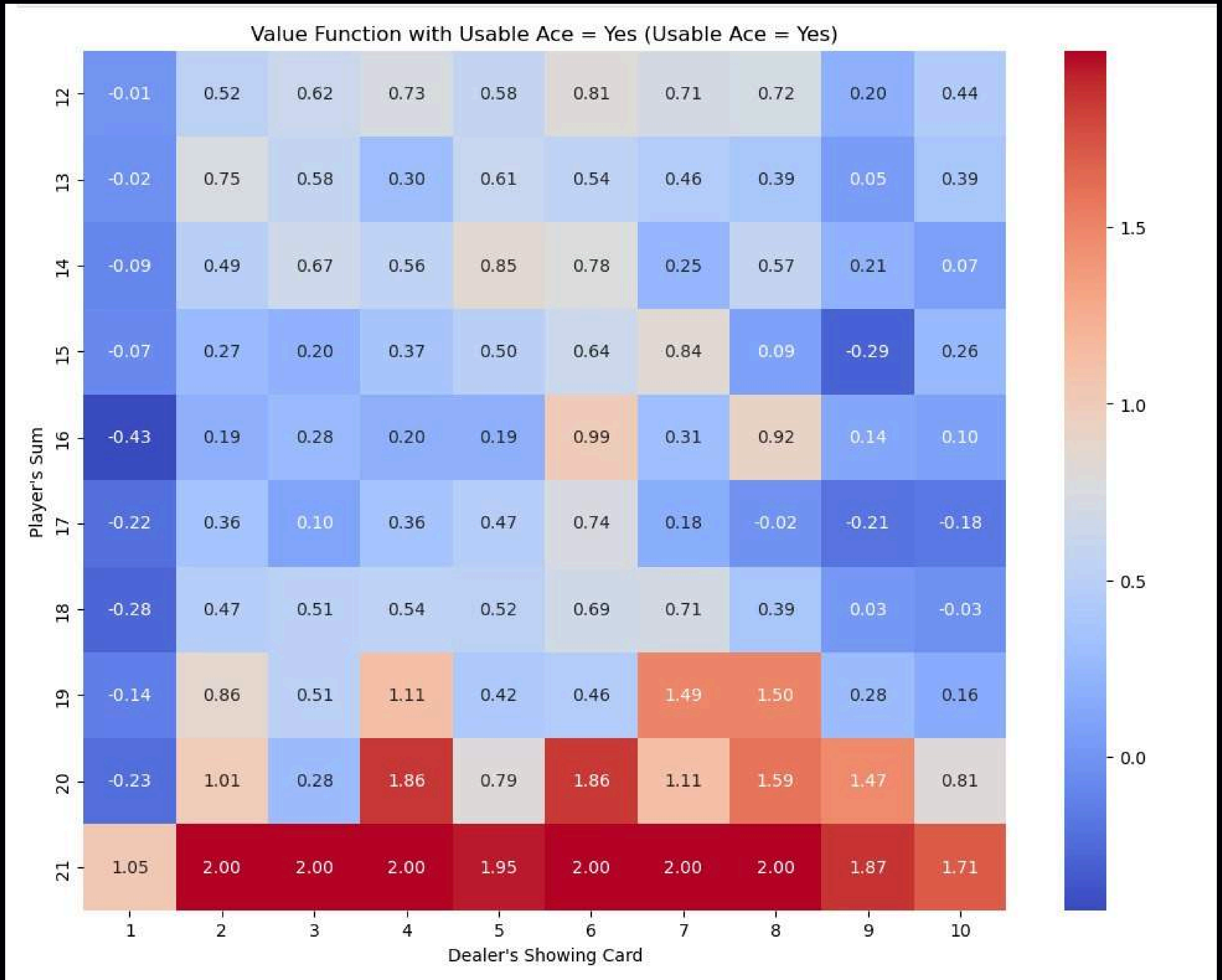
Hyperparameters Optimisation

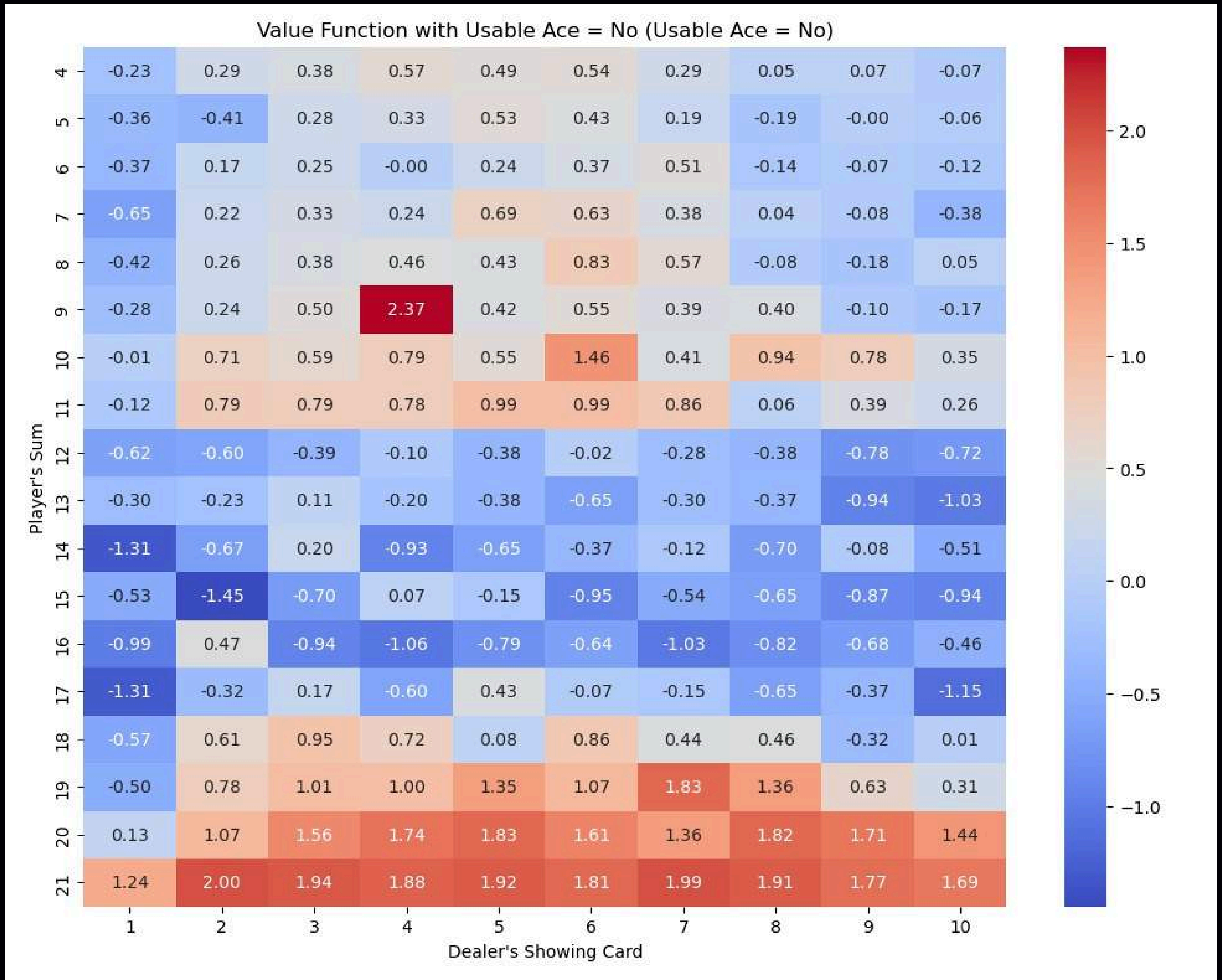
- We experimented with different values for gamma, alpha, and epsilon simultaneously in our Double Q-learning code.

- On running the code with all combinations, we evaluated the results to determine which combination of gamma, alpha, and epsilon produces the most optimal outcomes. The best-performing parameter values were then selected and incorporated into our model permanently.

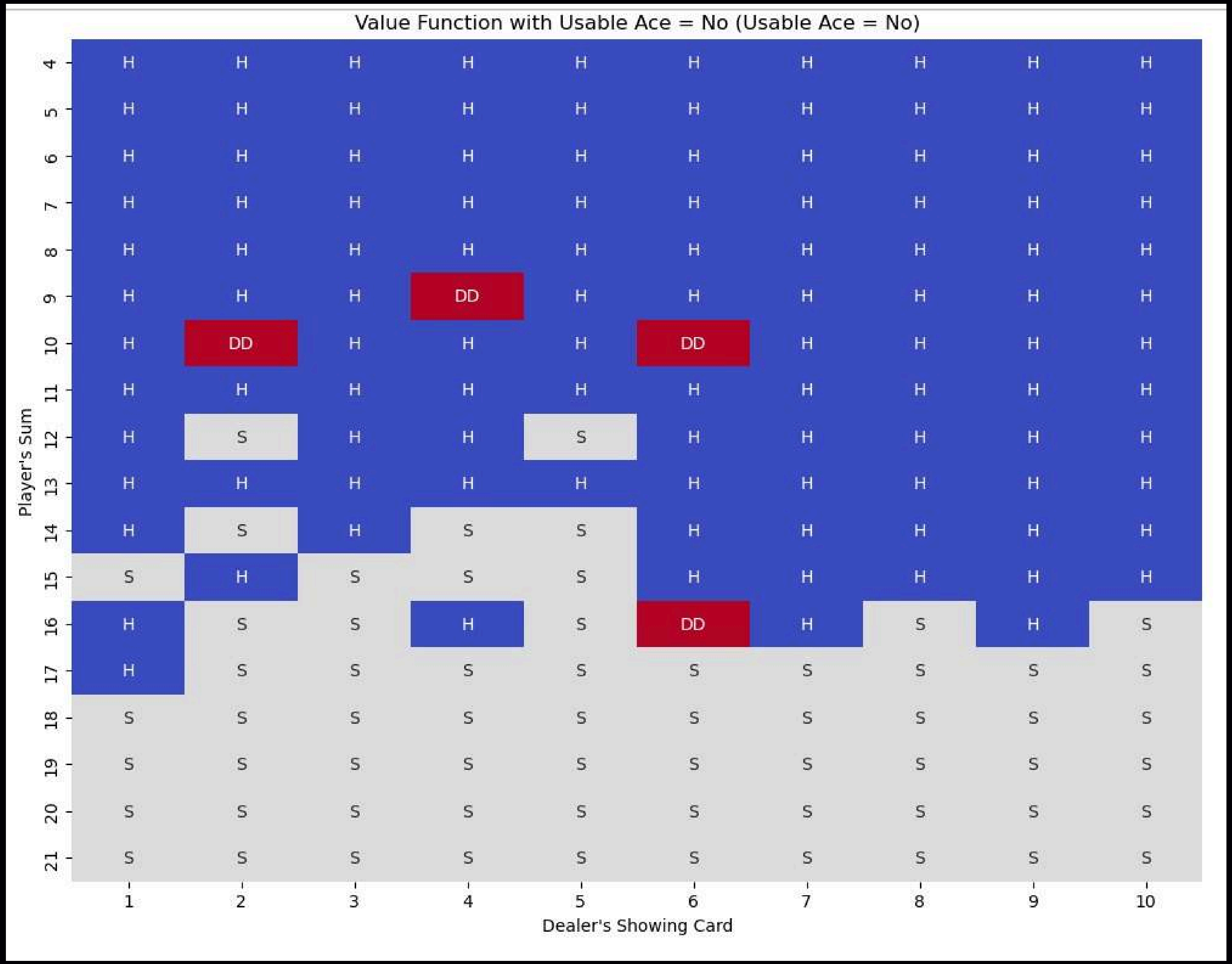
- The best values were Alpha - 0.1, Epsilon - 0.2 and Gamma - 0.8

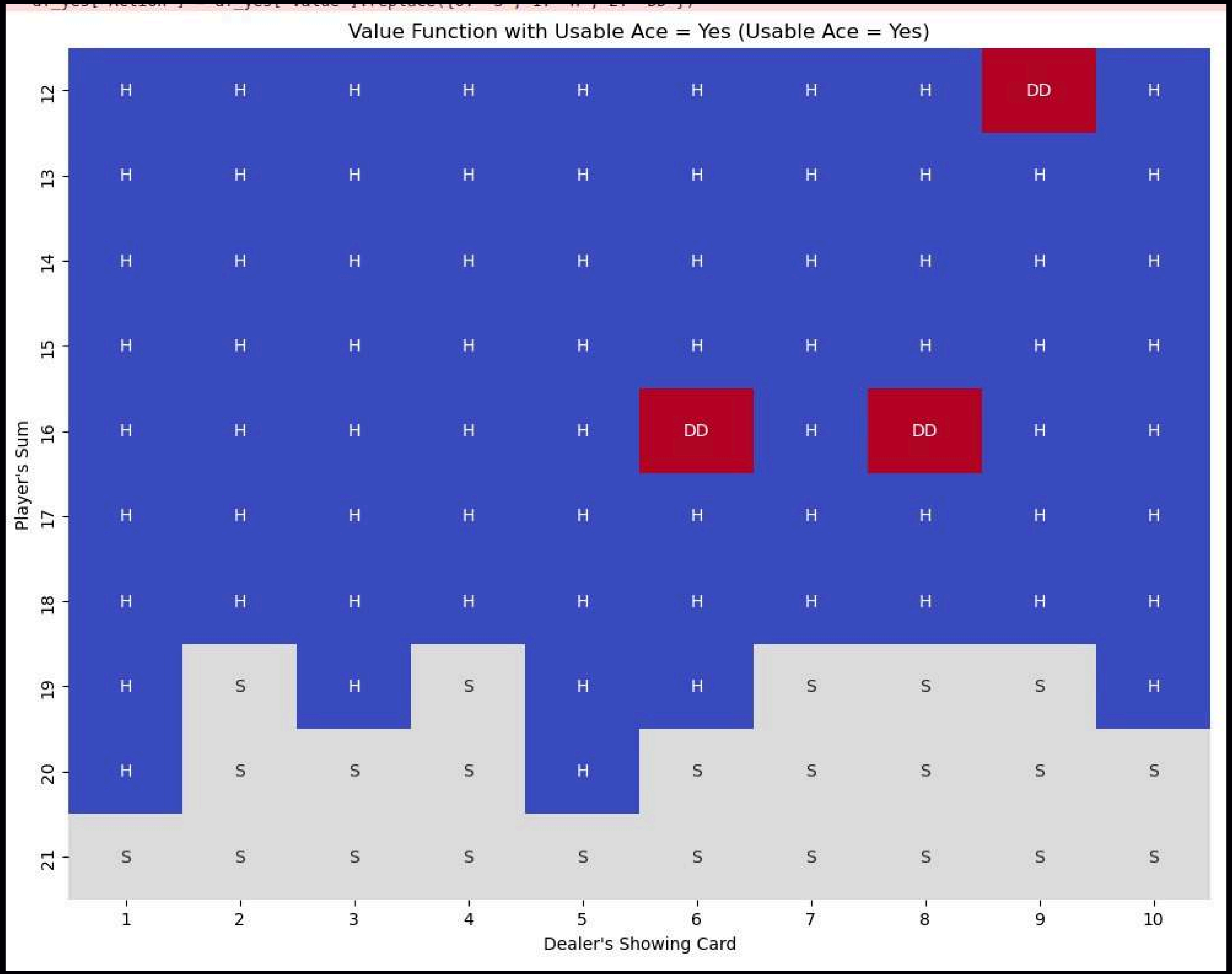
Our Models Value Functions





Our Models Suggested Strategy

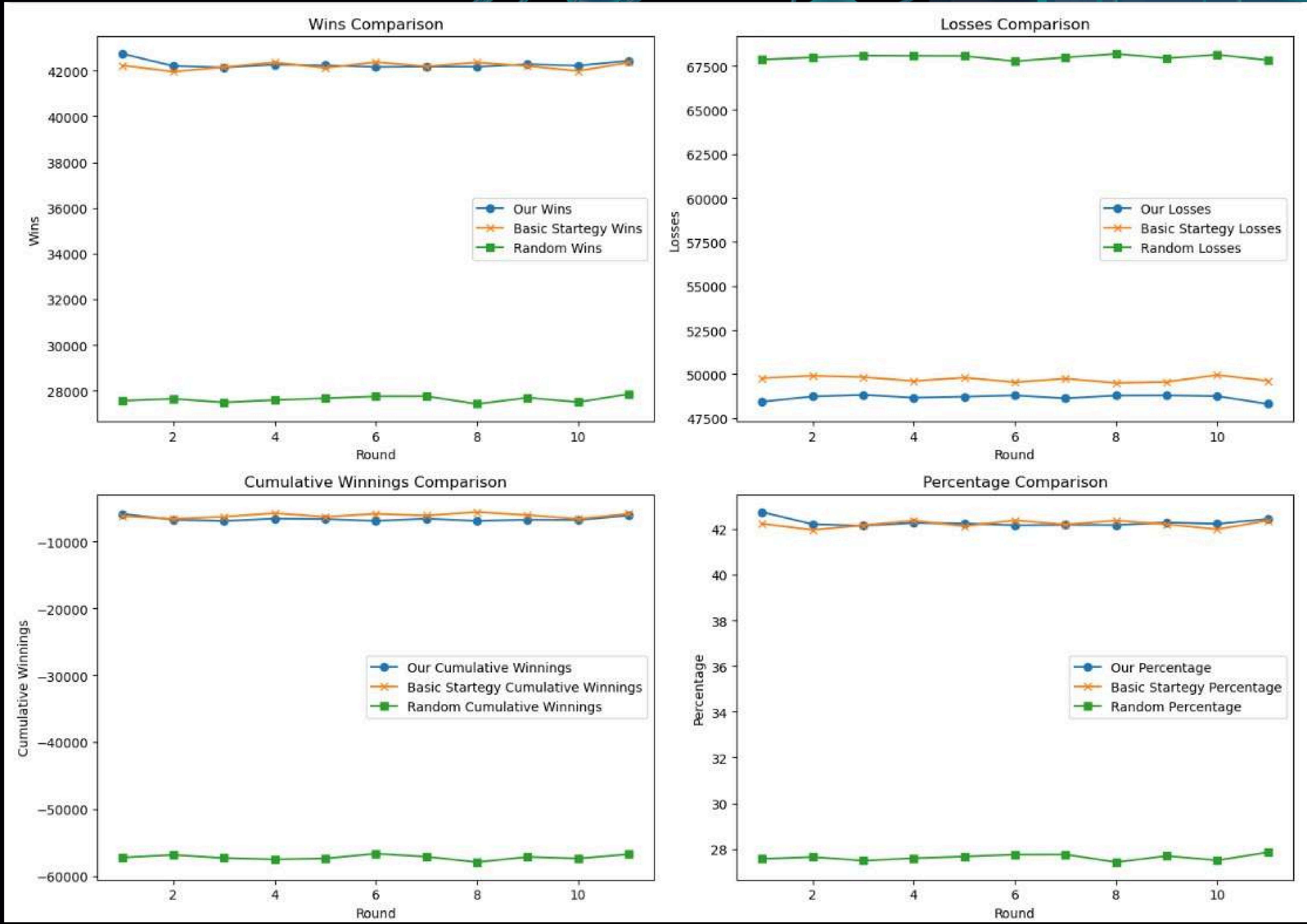




How did we evaluate the model.

- Finding the winnings and losses of basic strategy over 100K games
- Finding the winnings and losses of our implemented strategy over 100K games
- Comparing the accuracy and working of the model.
- Comparing the model policy score with basic startegy (optimal)

Model Accuracy

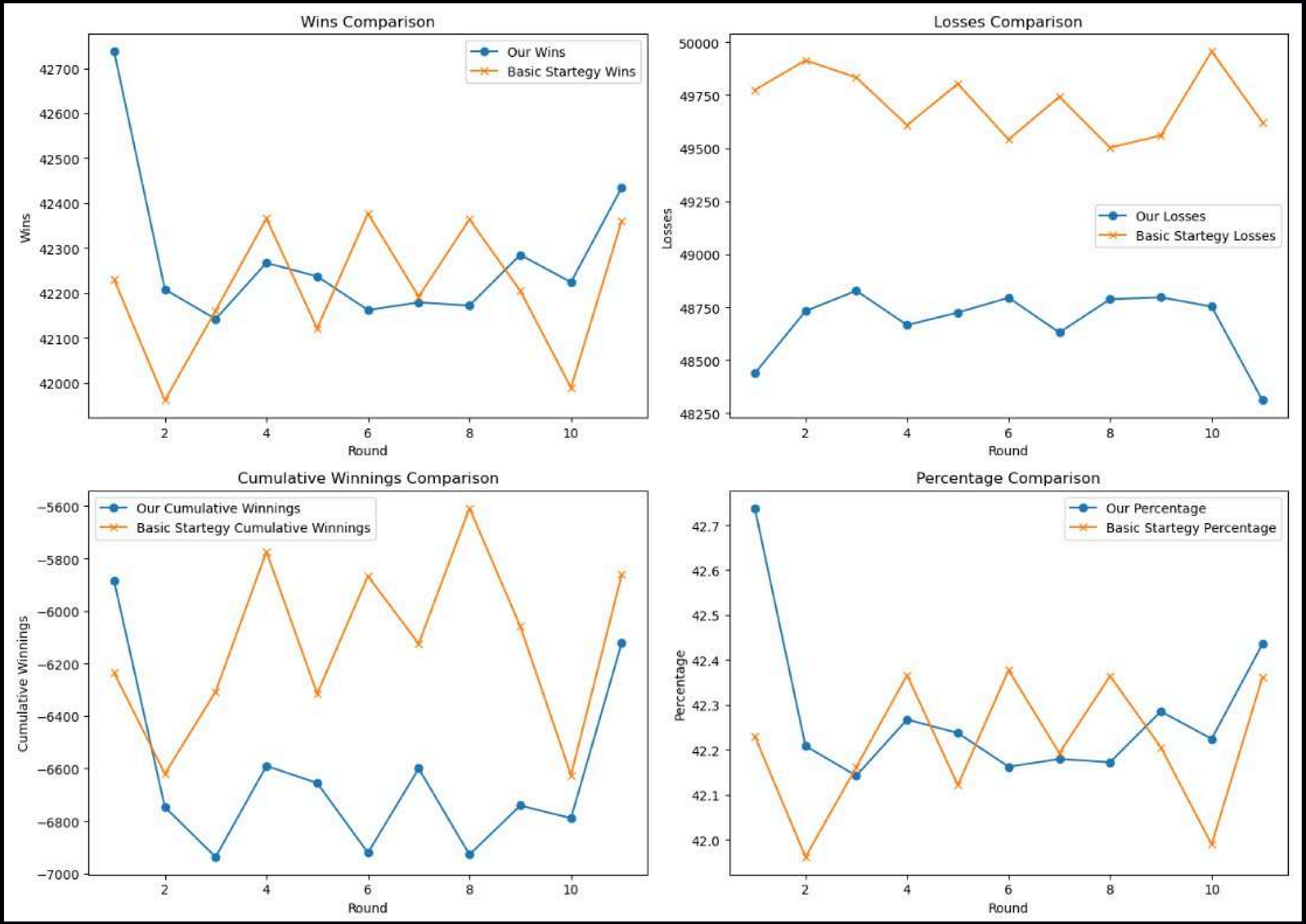


Policy Score

- For Ace Matrix - 48.5/100
- For Non - Ace Matrix - 154/180
- Total = 202.5/280
- 72.3%

Thank you

Hoping that you would never gamble


Wins Comparison
Our Wins
Basic Startegy Wins
Wins
Round
Losses Comparison
Our Losses
Basic Startegy Losses
Losses
Round
Cumulative Winnings Comparison
Our Cumulative Winnings
Basic Startegy Cumulative Winnings
Cumulative Winnings
Round
Percentage Comparison
Our Percentage
Basic Startegy Percentage
Percentage
Round

```
Top performing hyperparameters:
     Gamma  Alpha  Epsilon     Wins   Losses  Cumulative Winnings  \
3     0.80   0.10     0.20  42859.0  47866.0              -4864.0
0     0.80   0.10     0.10  42808.0  48492.0              -5647.0
26    0.85   0.10     0.15  42719.0  47991.0              -5107.0
109   1.00   0.15     0.30  42654.0  48578.0              -5790.0
5     0.80   0.15     0.10  42647.0  49039.0              -6334.0

     winperecentgae
3            42.859
0            42.808
26           42.719
109          42.654
5            42.647

Best hyperparameters:
Gamma                      0.800
Alpha                      0.100
Epsilon                    0.200
Wins                   42859.000
Losses                 47866.000
Cumulative Winnings    -4864.000
winperecentgae            42.859
Name: 3, dtype: float64
```