MASTERING BLACKJACK WITH REINFORCEMENT LEARNING

Agenda

1.

ENVIRONMENT

This section presents the construction of the environment used to model the reinforcement learning agent. It outlines the **observation space**, the set of **available actions**, the **reward structure**, and **specific rules** governing the environment's dynamics..

2.

BASELINE STRATEGIES

This part presents the implementation of **two baseline strategies**, which serve as **benchmarks** for evaluating the **policy performance** of the reinforcement learning agent.

3.

FEATURE TRANSFORMATION

This segment covers the development of **two feature transformers** designed to simplify the observation space by **reducing its size** or **capturing nonlinearities** and **potential interactions.**

4.

AGENT IMPLEMENTATION

This final phase presents the implementation of a reinforcement learning agent using **linear function approximation**. The agent is trained with **model-free algorithms** to learn an effective strategy for playing the game.

ENVIRONMENT

GYM PACKAGE

Observation Space

32

×

PLAYER SUM

Total value of the player's hand, ranging from 0 to 31, with face cards counting as 10 and aces as either 1 or 11 depending on usability

11

×

DEALER UPCARD

Value of the **dealer's visible face-up card**, ranging from 1 to 11.

(0,1)

×

USABLE ACE

Boolean indicating whether the player holds an **ace** that **can be counted as 11** without busting

Size

32 x 11 x 2
= **704**

CUSTOMIZED

Observation Space

32 × 11 × (0,1) × 11

PLAYER SUM

Total value of the player's hand, ranging from 0 to 31, with face cards counting as 10 and aces as either 1 or 11 depending on usability

DEALER UPCARD

Value of the **dealer's visible face-up card**, ranging from 1 to 11.

USABLE ACE

Boolean indicating whether the player holds an **ace** that **can be counted as 11** without busting

PEEKED CARD

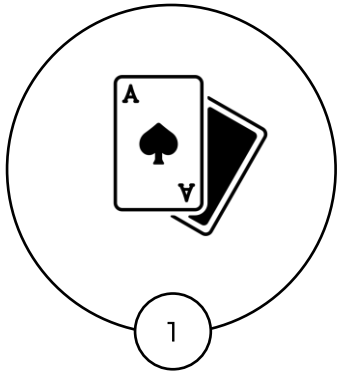
Value of the **peeked card**, ranging from 1 (ace) to 10.

Size

32 x 11 x 2 x 11
= **7744 !**

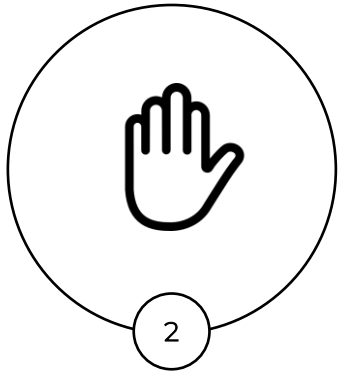
GYM PACKAGE

Action Space



HIT

The player chooses to **draw another card** from the deck.

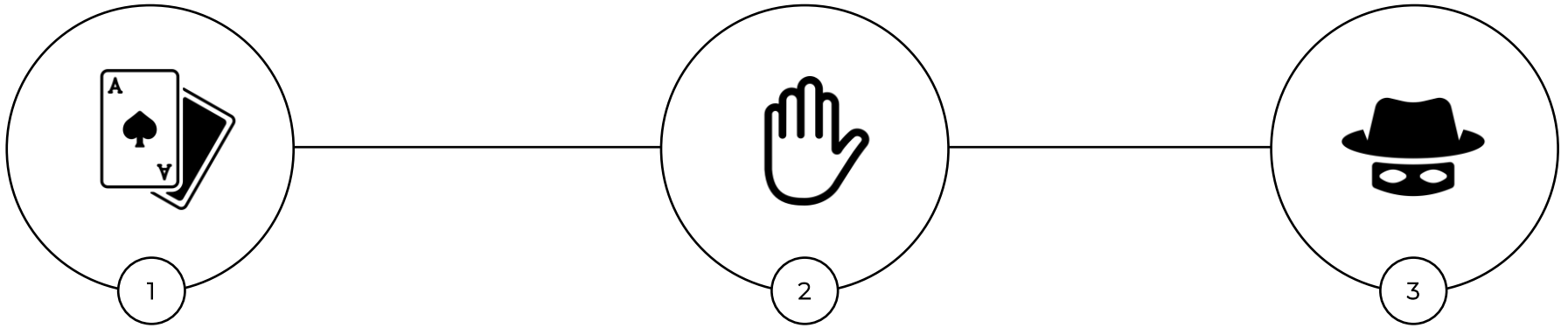


STAND

The player chooses to **stop drawing cards** and **end their turn**.

CUSTOMIZED

Action Space



HIT

The player chooses to **draw another card** from the deck.

STAND

The player chooses to **stop drawing cards** and **end their turn**.

PEEK

The player **attempts** to **look at the next card** in the deck; if **successful**, it provides **valuable information** for deciding whether to hit or stand, but if **caught**, it results in a **penalty**.

Rules

SHUFFLING RULES

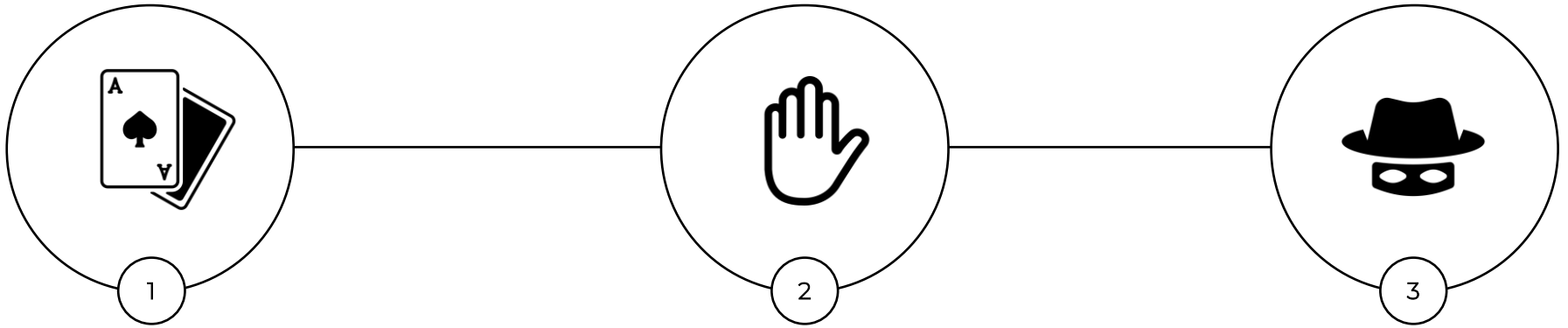
- The deck is reshuffled at the **start of the game**.
- As the game progresses, drawn cards are **not returned** to the deck and **cannot appear again**.
- Once **fewer than 30%** of the original cards remain, the **deck is reshuffled**.

PEEKING RISK

- The player is assumed to be an **expert gambler**, **successfully** peeking at cards **85% of the time**.
- However, each time the player is **caught** peeking, the dealer becomes more watchful, **reducing** the player's future **success rate by 40%** per **failed attempt.**

SCENARIO 1

Reward Structure



HIT

The player receives a reward of **0** if the action **does not result in a bust**, and **-1** if it does, indicating the **game is lost**.

STAND

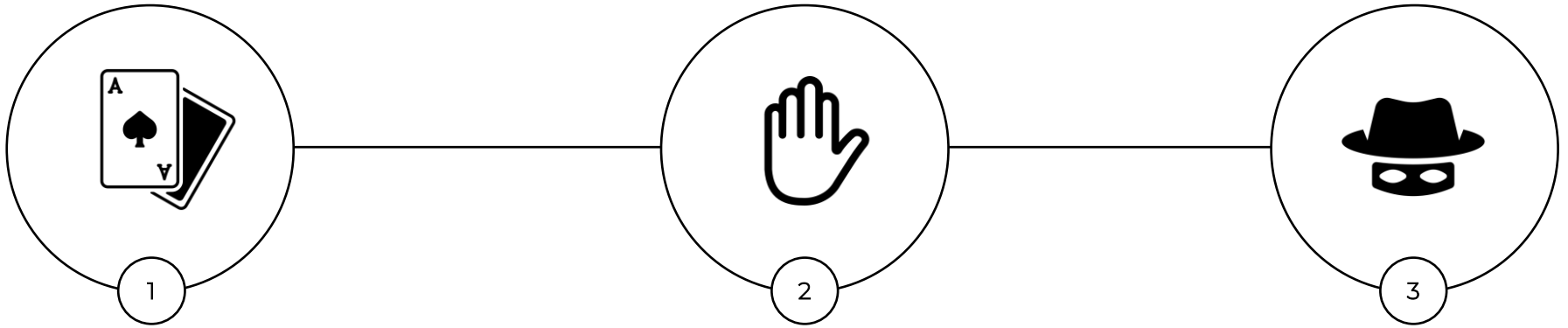
The player receives **+1** for **winning** the game, **-1** for **losing**, **0** for **draw**

PEEK

The player receives **0** for **successfully peeking**, and incurs a **penalty** of **-0.5** if the peek is **unsuccessful**.

SCENARIO 2

Reward Structure



HIT

The player receives a reward of **0** if the action **does not result in a bust**, and **-1** if it does, indicating the **game is lost**.

STAND

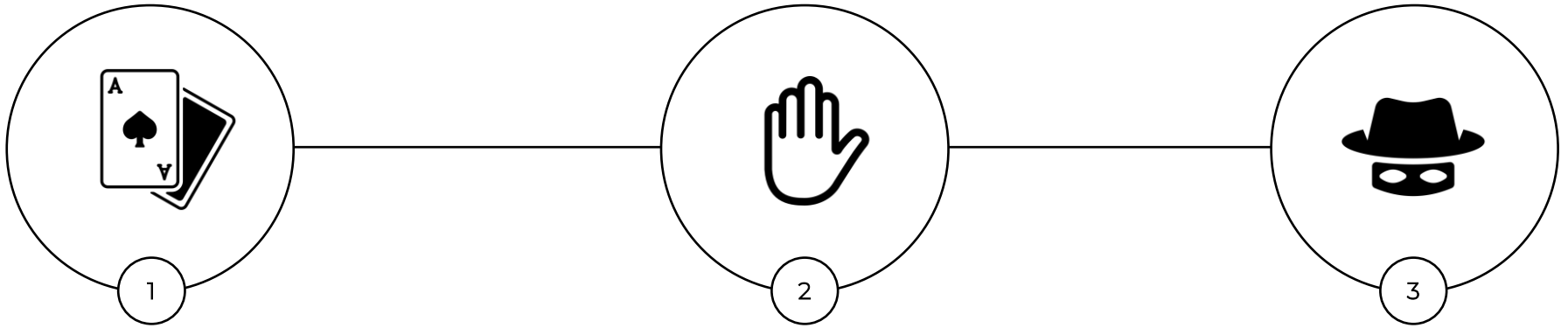
The player receives **+1** for **winning** the game, **-1** for **losing**, **0** for **draw**

PEEK

The player receives **0** for **successfully peeking**, and incurs a **penalty** of **-2** if the peek is **unsuccessful**.

SCENARIO 3

Reward Structure



HIT

The player receives a reward of **0** if the action **does not result in a bust**, and **-1** if it does, indicating the **game is lost**.

STAND

The player receives **+1** for **winning** the game, **-1** for **losing**, **0** for **draw**

PEEK

The player receives **0** for **successfully peeking**, and incurs a **penalty** of **-1** if the peek is **unsuccessful**.

BASELINE STRATEGIES

Baseline Strategies

RANDOM

- The player chooses **randomly** between the actions: **hit**, **stand**, and **peek**.

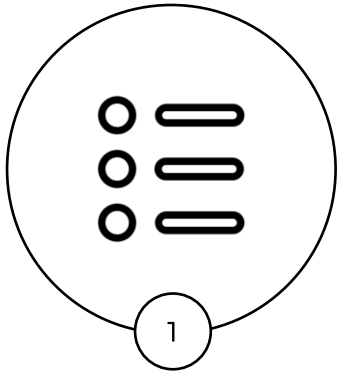
BASIC

- The player follows a basic strategy: **80% of the time**, they use a naïve approach, **hitting** when their **card total** is **17 or less**, and **standing** if it is **18 or higher**.

- In the **remaining 20%**, the player uses **peeking**. If the peek is **successful**, they **hit** only if the **sum of their cards and the peeked card** is **21 or less**; otherwise, they **stand**.

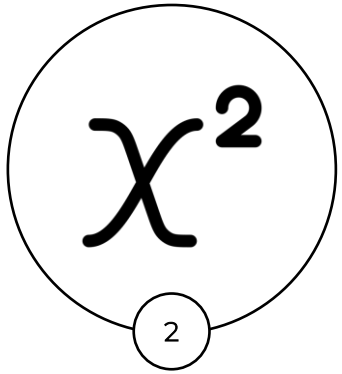
FEATURE TRANSFORMATION

Feature Transformation



THRESHOLD

- Converts game state into **categorical levels** (**Low**, **Medium**, **High**) mapped to **float values**.
- Focuses on **strategic abstractions** like usable ace and dealer danger level.
- Embeds the state only for the selected action using **block-wise encoding**.



POLYNOMIAL

- **Normalizes** raw state values and applies a **degree-2 polynomial expansion**.
- Encodes action using **one-hot vectors** and **captures all pairwise feature interactions**.
- Focuses on **non-linear** modeling of complex game patterns.

Threshold Transformation

EXAMPLE

PLAYER SUM:  14

DEALER UPCARD: 

USABLE ACE: ∅

PEEKED CARD: ∅

ACTION: 

Threshold Transformation

EXAMPLE

PLAYER SUM:  14

DEALER UPCARD: 

USABLE ACE: ∅

PEEKED CARD: ∅

ACTION: 

[0. 0. 0. 0. Hit
0. 0. 0. 0. Stand
0.5 1. 0. -0.5] Peek

Medium High No Usable Ace No Card

Polynomial Transformation

$$\tilde{s}_i = \frac{s_i - \min_i}{\max_i - \min_i}$$

NORMALIZATION

Each raw state value s_i is **linearly scaled** to the interval *[0,1]*, where min_i and max_i are the predefined bounds for that feature.

$$\mathbf{e}(a)_k = \begin{cases} 1, & k = a, \\ 0, & k \neq a. \end{cases}$$

ACTION ENCODING

The integer **action** $a \in \{0,1,2\}$ is converted to a **one-hot vector** $e(a) \in \{0,1\}^3$

$$\phi(\mathbf{x}) = \left\{ x_i\, x_j \mid 0 \le i \le j < d \right\}$$

FEATURE EXPANSION

A **bias term 1**, the **normalized state** $\tilde{s}$,and the **one-hot action** $e,a«$are concatenated into a **vector** x. Then every **pairwise** (including squared) **product is computed**, where d is the length of x.

AGENT IMPLEMENTATION

Agent Implementation

$$Q(s,a) = w \cdot \phi(s,a)$$

Q-VALUES FOR (s,a)

The agent begins by estimating how good it is to take a particular action in a given state by projecting the state-action features onto its weight vector. In practice, the feature transformer produces a **vector $\phi(s,a)$** and the **inner product with w** yields the **current value estimate**

1

$$w \leftarrow w + \alpha \left[r + \gamma \max_{a'} Q(s',a') - Q(s,a) \right] \phi(s,a)$$

Q-LEARNING UPDATE

After executing an action and **observing reward r** and **successor state $\tilde{s}$**, the agent computes the target as the **immediate reward plus the discounted maximal estimated future value**. The difference between this target and the prior estimate serves as an **error signal**, which when scaled by the **learning rate α** and the **feature vector $\phi(s,a)$** produces the **weight adjustment**

2

$$w \leftarrow w + \alpha \left[r + \gamma Q(s',a') - Q(s,a) \right] \phi(s,a)$$

SARSA UPDATE

The update similarly hinges on the temporal-difference error, but uses the value of the **action $\tilde{a}$** actually selected in s' rather than the best possible. By doing so, the **agent's exploration choices directly shape the learning target**

Training Evaluation

Peek Reward = - 0.5

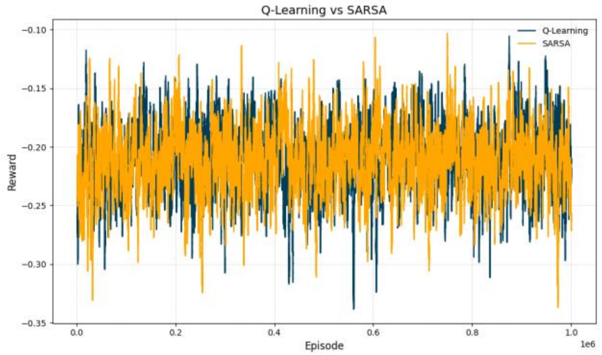
	Q-Learning		SARSA	
Feature Transformation	**Average Return**	**Win Rate**	**Average Return**	**Win Rate**
Threshold	-0.1263	49.21%	-0.1257	49.20%
Polynomial	-0.1329	49.49%	-0.1401	49.08%

Peek Reward = - 1

	Q-Learning		SARSA	
Feature Transformation	**Average Return**	**Win Rate**	**Average Return**	**Win Rate**
Threshold	-0.1417	48.37%	-0.14	48.45%
Polynomial	-0.1469	48.47%	-0.1542	48.12%

Peek Reward = - 2

	Q-Learning		SARSA	
Feature Transformation	**Average Return**	**Win Rate**	**Average Return**	**Win Rate**
Threshold	-0.1907	40.23%	-0.1891	40.33%
Polynomial	-0.2108	39.39%	-0.2119	39.33%



Training Evaluation

Peek Reward = - 0.5

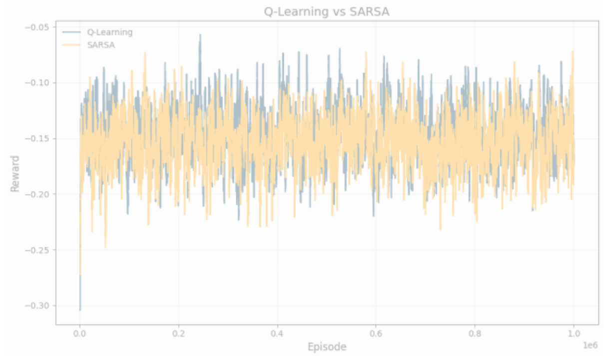
	Q-Learning		SARSA	
Feature Transformation	Average Return	Win Rate	Average Return	Win Rate
Threshold	-0.1263	49.21%	**-0.1257**	**49.20%**
Polynomial	-0.1329	49.49%	-0.1401	49.08%

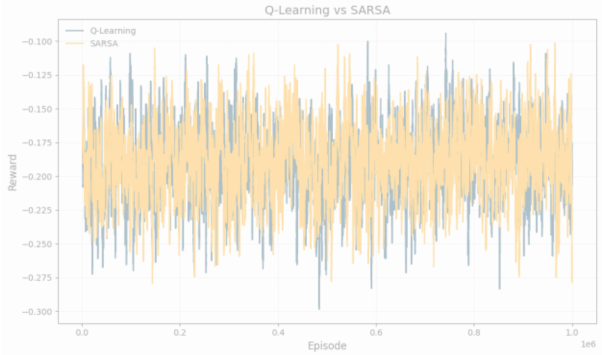
Peek Reward = - 1

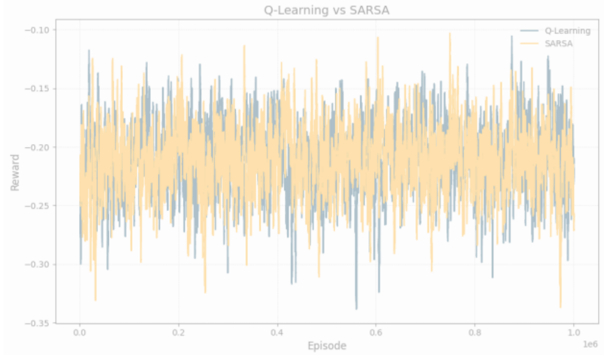
	Q-Learning		SARSA	
Feature Transformation	Average Return	Win Rate	Average Return	Win Rate
Threshold	-0.1417	48.37%	-0.14	48.45%
Polynomial	-0.1469	48.47%	-0.1542	48.12%

Peek Reward = - 2

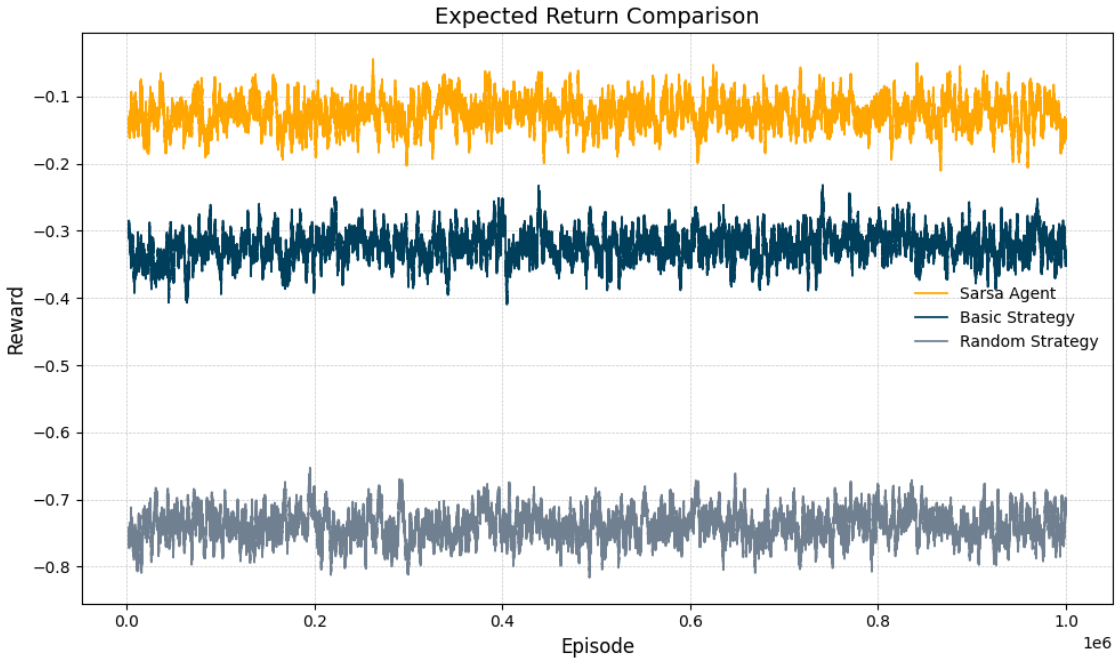
	Q-Learning		SARSA	
Feature Transformation	Average Return	Win Rate	Average Return	Win Rate
Threshold	-0.1907	40.23%	-0.1891	40.33%
Polynomial	-0.2108	39.39%	-0.2119	39.33%

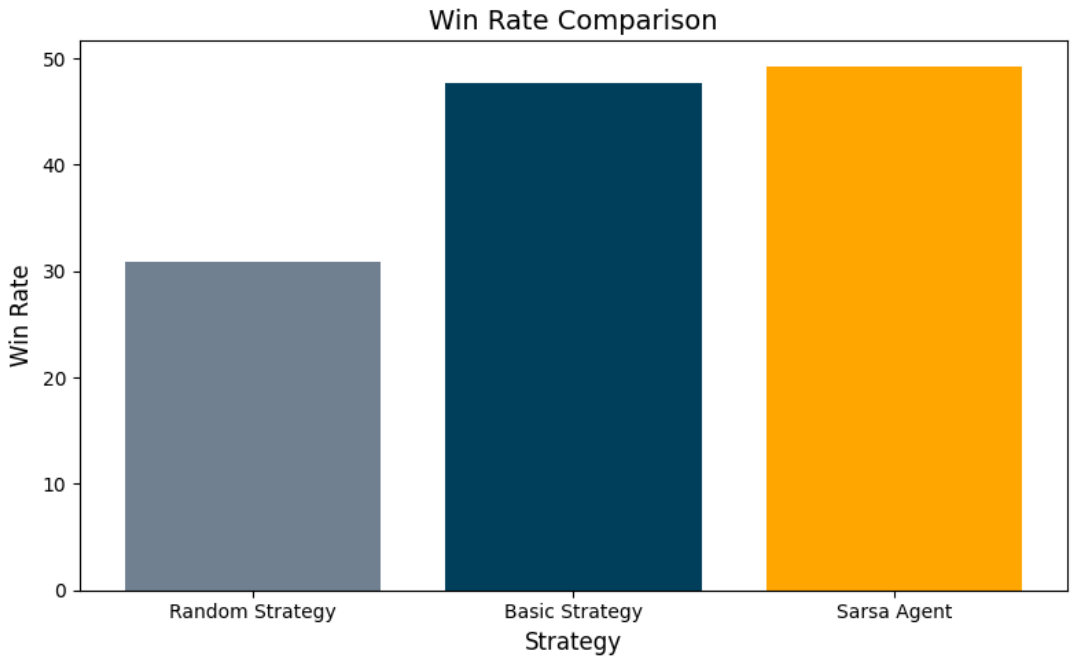




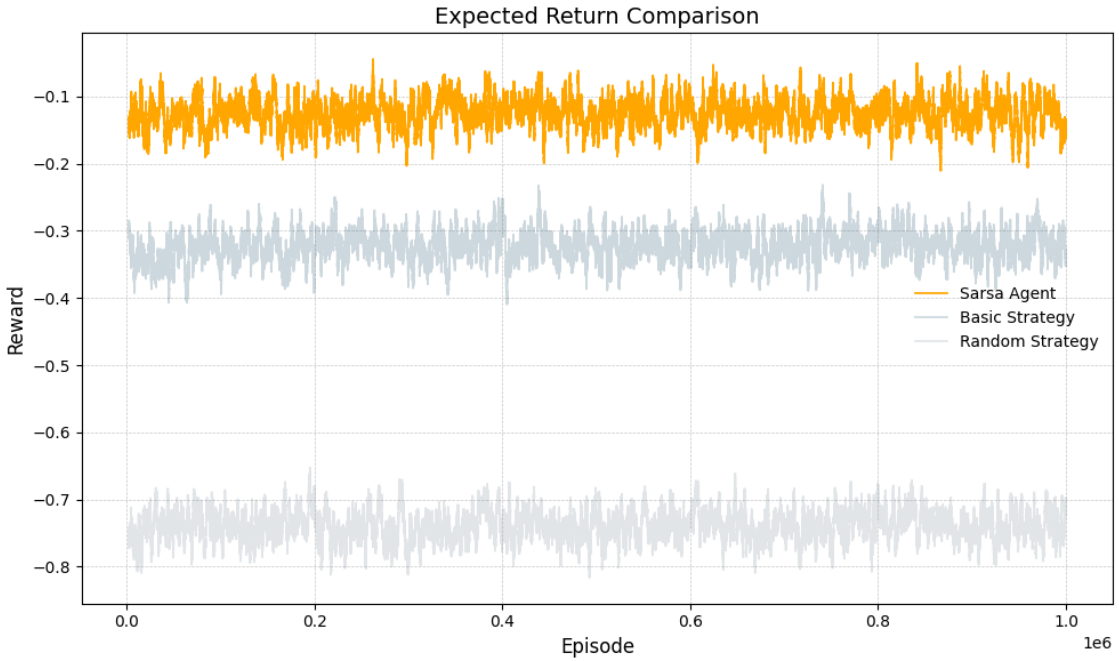


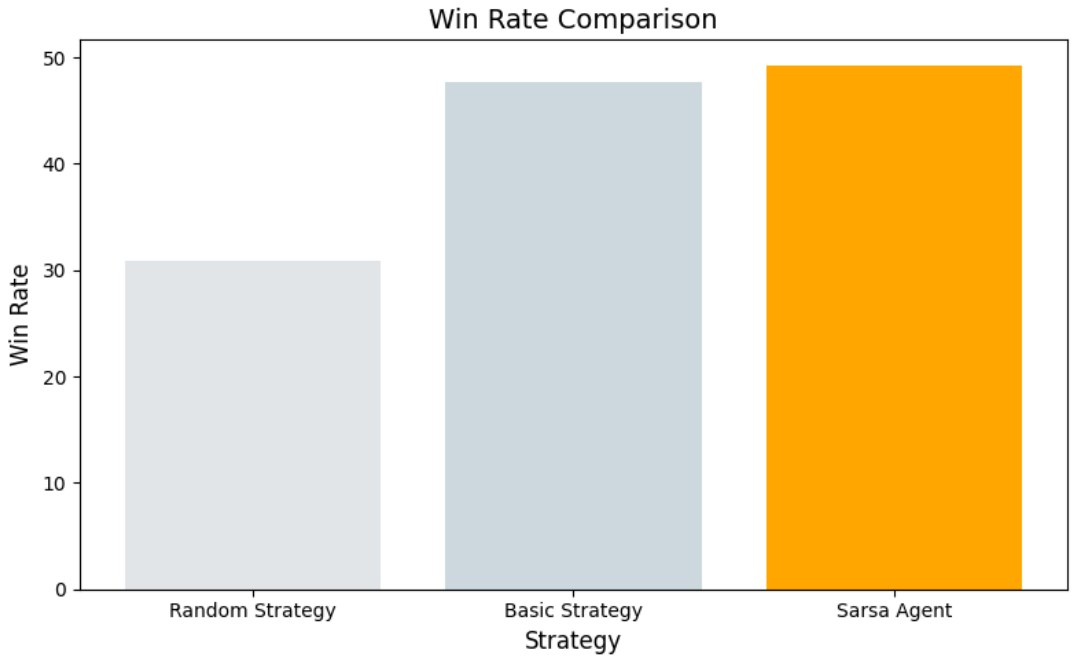
Final Comparison





Final Comparison





THANK YOU