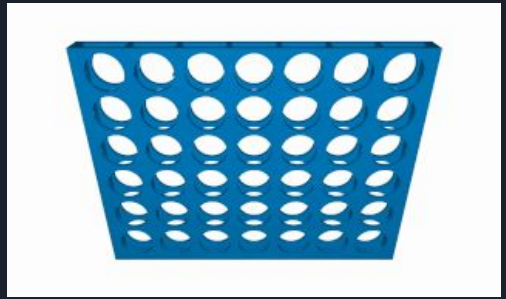
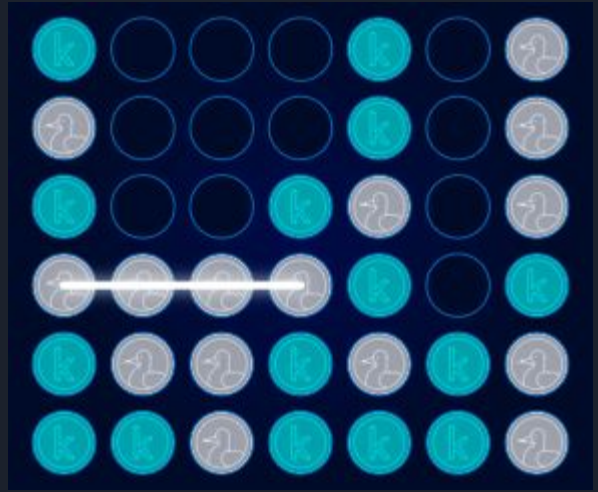
# ConnectX on Kaggle

Joe Johnson

# Goal

- Machines are trained to compete with each other to get a certain number of checkers (X) in-a-row before their opponent
- Players alternate placing one piece at a time at the lowest available free cell in a column
- The current default settings for this game are a gameboard of 6 rows and 7 columns with the players attempting to get 4 in-a-row

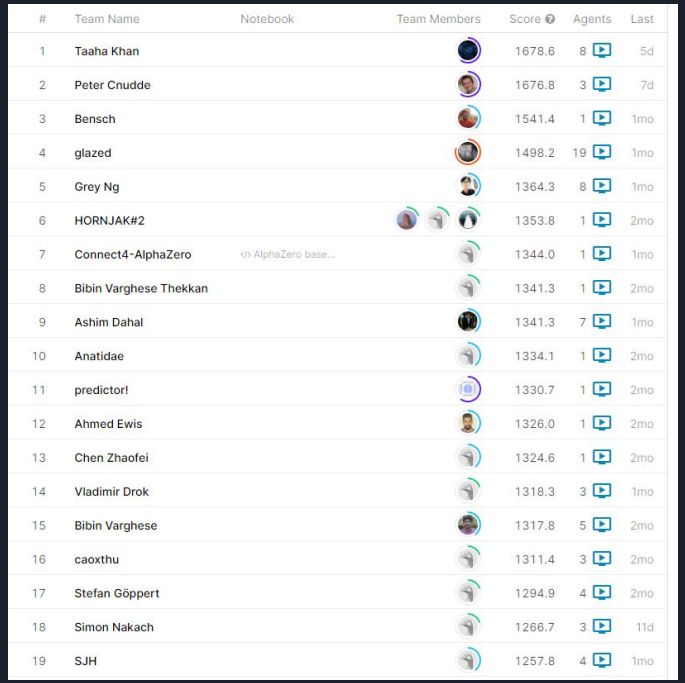




# What's different than normal?

- Rather than trying to get a high score in a challenge independently, players trained machines face-off and acquire a rating that fluctuates based off of wins and losses

- Similar to a competitive ranked system in Chess and Video Games



| # | Team Name | Notebook | Team Members | Score | Agents | Last |
|---|---|---|---|---|---|---|
| 1 | Taaha Khan | | | 1678.6 | 8 | 5d |
| 2 | Peter Cnudde | | | 1676.8 | 3 | 7d |
| 3 | Bensch | | | 1541.4 | 1 | 1mo |
| 4 | glazed | | | 1498.2 | 19 | 1mo |
| 5 | Grey Ng | | | 1364.3 | 8 | 1mo |
| 6 | HORNJAK#2 | | | 1353.8 | 1 | 2mo |
| 7 | Connect4-AlphaZero | AlphaZero base... | | 1344.0 | 1 | 1mo |
| 8 | Bibin Varghese Thekkan | | | 1341.3 | 1 | 2mo |
| 9 | Ashim Dahal | | | 1341.3 | 7 | 1mo |
| 10 | Anatidae | | | 1334.1 | 1 | 2mo |
| 11 | predictor! | | | 1330.7 | 1 | 2mo |
| 12 | Ahmed Ewis | | | 1326.0 | 1 | 2mo |
| 13 | Chen Zhaofei | | | 1324.6 | 1 | 2mo |
| 14 | Vladimir Drok | | | 1318.3 | 3 | 1mo |
| 15 | Bibin Varghese | | | 1317.8 | 5 | 2mo |
| 16 | caoxthu | | | 1311.4 | 3 | 2mo |
| 17 | Stefan Göppert | | | 1294.9 | 4 | 2mo |
| 18 | Simon Nakach | | | 1266.7 | 3 | 11d |
| 19 | SJH | | | 1257.8 | 4 | 1mo |

# My Method

- Model-Free methods
- Q-Learning Method: Model-Free reinforcement learning algorithm
  - Calculates which move from a given state has best expected rewards
  - ‘Q” is used to name the function that represents this path
  - Tries to check all states (through somewhat random moves) and records score of outcome to help calculate potential score of each possible move in a given state
- Epsilon-Greedy Algorithm (Epsilon chance of exploring new state)

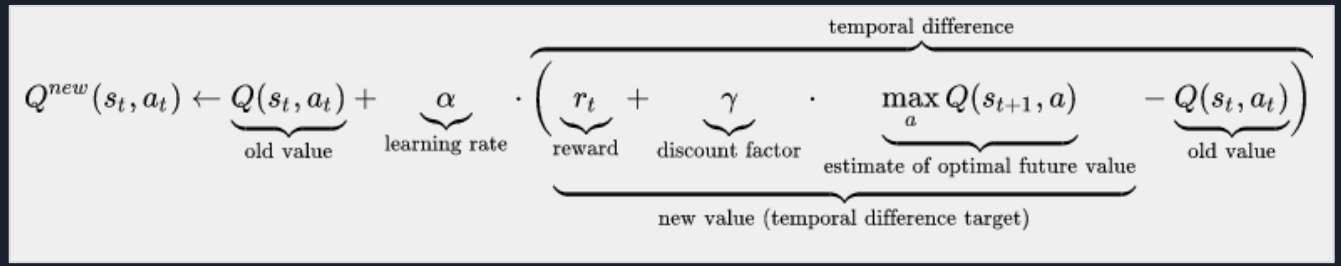
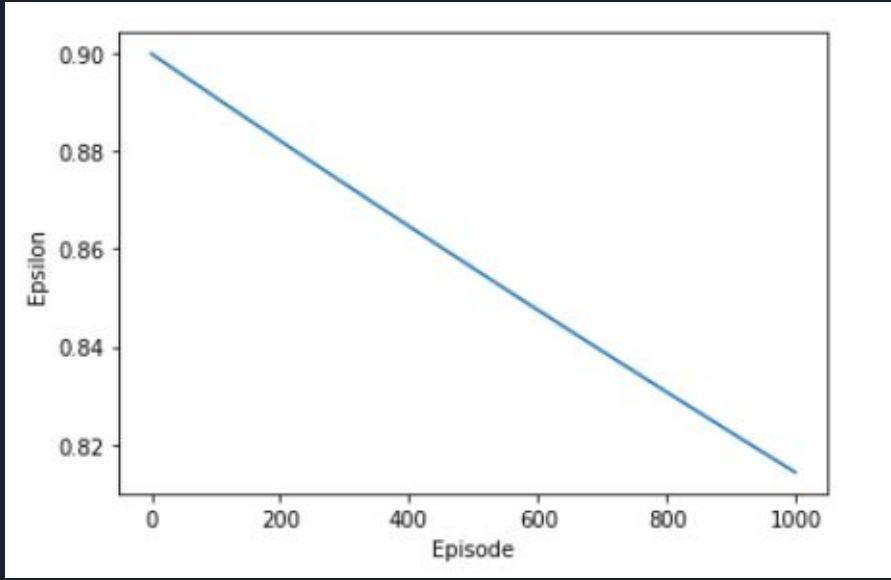
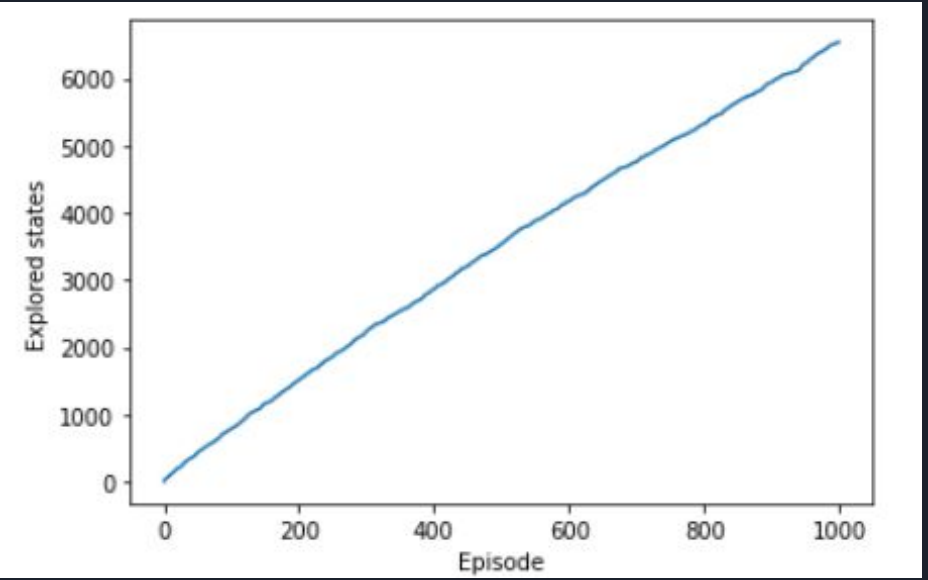



Image Source: https://en.wikipedia.org/wiki/Q-learning

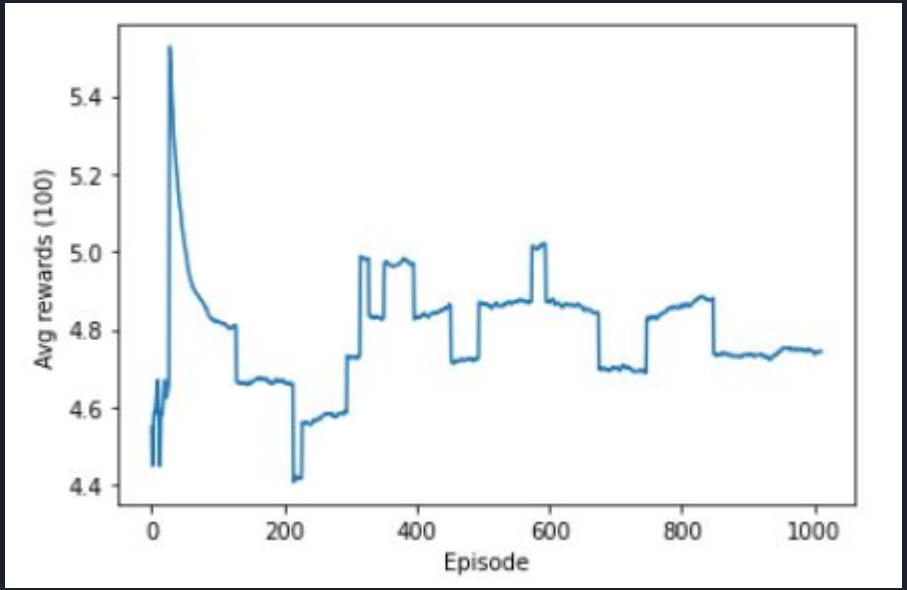
# Current Results (Small Run)

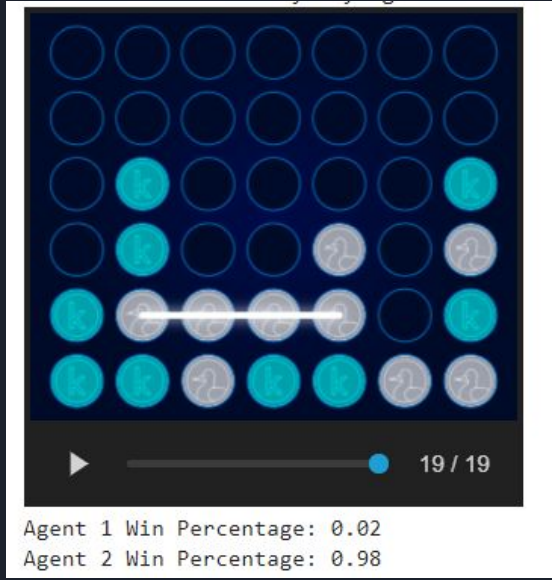



Epsilon Degradation (Probability of exploring random state)

# Current Results (Cont.)



# Agent vs. Agent

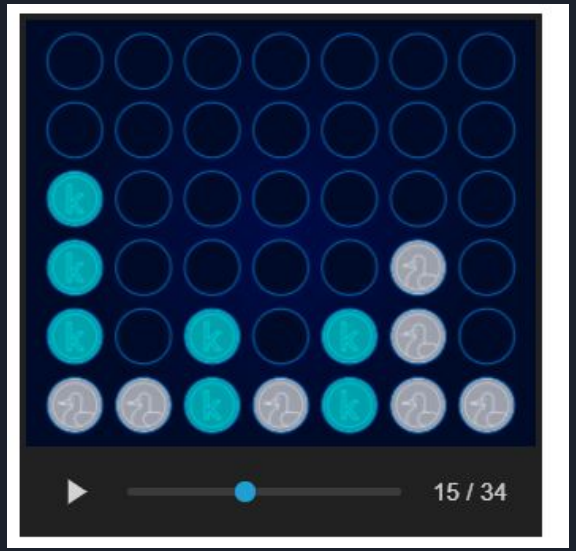
```
my agent vs random
Agent 1 Win Percentage: 0.49
Agent 2 Win Percentage: 0.51
Number of Invalid Plays by Agent 1: 0
Number of Invalid Plays by Agent 2: 0

depth 1 vs negamax
Agent 1 Win Percentage: 0.61
Agent 2 Win Percentage: 0.39
Number of Invalid Plays by Agent 1: 0
Number of Invalid Plays by Agent 2: 0

depth 3 vs negamax
Agent 1 Win Percentage: 0.94
Agent 2 Win Percentage: 0.02
Number of Invalid Plays by Agent 1: 0
Number of Invalid Plays by Agent 2: 0

depth 5 vs negamax
Agent 1 Win Percentage: 0.98
Agent 2 Win Percentage: 0.01
Number of Invalid Plays by Agent 1: 0
Number of Invalid Plays by Agent 2: 0

depth 3 vs depth 5
Agent 1 Win Percentage: 0.0
Agent 2 Win Percentage: 1.0
Number of Invalid Plays by Agent 1: 0
Number of Invalid Plays by Agent 2: 0

my agent vs depth 1
Agent 1 Win Percentage: 0.0
Agent 2 Win Percentage: 1.0
Number of Invalid Plays by Agent 1: 0
Number of Invalid Plays by Agent 2: 0
```

```
my agent vs random
Agent 1 Win Percentage: 0.5
Agent 2 Win Percentage: 0.5
Number of Invalid Plays by Agent 1: 0
Number of Invalid Plays by Agent 2: 0

depth 1 vs negamax
Agent 1 Win Percentage: 0.56
Agent 2 Win Percentage: 0.42
Number of Invalid Plays by Agent 1: 0
Number of Invalid Plays by Agent 2: 0

depth 3 vs negamax
Agent 1 Win Percentage: 0.94
Agent 2 Win Percentage: 0.01
Number of Invalid Plays by Agent 1: 0
Number of Invalid Plays by Agent 2: 0

depth 5 vs negamax
Agent 1 Win Percentage: 0.99
Agent 2 Win Percentage: 0.01
Number of Invalid Plays by Agent 1: 0
Number of Invalid Plays by Agent 2: 0

depth 3 vs depth 5
Agent 1 Win Percentage: 0.0
Agent 2 Win Percentage: 1.0
Number of Invalid Plays by Agent 1: 0
Number of Invalid Plays by Agent 2: 0

my agent vs depth 1
Agent 1 Win Percentage: 0.0
Agent 2 Win Percentage: 1.0
Number of Invalid Plays by Agent 1: 0
Number of Invalid Plays by Agent 2: 0
```

```
my agent vs random
Agent 1 Win Percentage: 0.51
Agent 2 Win Percentage: 0.49
Number of Invalid Plays by Agent 1: 0
Number of Invalid Plays by Agent 2: 0

depth 1 vs negamax
Agent 1 Win Percentage: 0.52
Agent 2 Win Percentage: 0.46
Number of Invalid Plays by Agent 1: 0
Number of Invalid Plays by Agent 2: 0

depth 3 vs negamax
Agent 1 Win Percentage: 0.98
Agent 2 Win Percentage: 0.01
Number of Invalid Plays by Agent 1: 0
Number of Invalid Plays by Agent 2: 0

depth 5 vs negamax
Agent 1 Win Percentage: 0.99
Agent 2 Win Percentage: 0.0
Number of Invalid Plays by Agent 1: 0
Number of Invalid Plays by Agent 2: 0

depth 3 vs depth 5
Agent 1 Win Percentage: 0.0
Agent 2 Win Percentage: 1.0
Number of Invalid Plays by Agent 1: 0
Number of Invalid Plays by Agent 2: 0

my agent vs depth 1
Agent 1 Win Percentage: 0.01
Agent 2 Win Percentage: 0.99
Number of Invalid Plays by Agent 1: 0
Number of Invalid Plays by Agent 2: 0
```

# Before Sunday

- Run longer training sessions and check performance
- Add check for winning moves (1-step MinMax possible win integration)
- Manipulate parameters (such as Epsilon, Learning Rate, etc)




Possible win on move 16 against random-agent

Thank you for listening!
Any questions?