Performance of Q-Learning and Value Iteration  
(CS605.449 – Project 6)

By Max Robinson

# Abstract

In this experiment the performance of

# Problem

The problem being investigated in this experiment is how well the Q-Learning Reinforcement algorithm performs on two different tracks in the Race Track problem and when compared to the Value Iteration solutions for the race tracks. The two race tracks used for this experiment are the L-Track and the R-Track. The Race Track problem is a control problem in which an agent controls a car on a race track and attempts to reach the finish line in the fewest number of steps possible. The R-Track also has two crash settings. The simple version places the agent on the nearest valid track space when the agent crashes. The harsh version places the agent back at the start of the track should a crash occur.

For this experiment I hypothesize that the Q-Learning algorithm will perform the best on the L-Track, while needing the most number of steps to reach the goal on the R-Track with the harsh crash setting. The L-Track is a much simpler track and it is also smaller. As a result, I expect the Q-Learning algorithm to be able to create a better strategy when compared to the larger more complex track.

I also hypothesize that the Value Iteration algorithms will create far more optimal policies than the Q-learning, though I believe it will take longer to train. Policy iteration looks at every possible state and iterates over the values until the values for the states stop changing very much. As a result, it is able to look at many more states and provide many more traceable rewards to every state that is involved in the game. As a result, I expect this algorithm to provide the shortest average number of steps to the goal.

# Algorithm Implementation

## Value Iteration

## Q-Learning

## Scoring

### Steps to Goal

### Iterations to Reach Minimum Steps

# Experimental Approach

The approach of this experiment was to train each of the algorithms, Q-Learning and Value Iteration, on each of the tracks, L-Track, R-Track, and R-Track with harsh crashing (R-Track-Harsh). For Q-Learning, the number of steps per epoch were captured to show how many steps were needed as the algorithm progressed in learning. The process of learning the models and capturing the number of steps needed per epoch was done ten times to average out each run.

A policy was then chosen to give a sample average number of steps it would take, applying that policy, to reach the goal. This provides the metrics by which the algorithms are compared. The average number of steps required are compared across tracks, and across algorithms.

The other information considered when comparing algorithms is amongst the Q-Learning algorithms. The number of epochs needed to converge towards the average number of steps for the model is considered when comparing the tracks to one another. This is shown through the average number of steps taken when learning for each of the ten runs of the experiments.

## Finding Parameters

Q-Learning

# Results

After the running the experiment as described in the above sections, the average error rates for each dataset and each positive class can be viewed.

|  |  |  |  |
| --- | --- | --- | --- |
| Algorithm | L-Track | R-Track | R-Trach-Harsh |
| Q-Learning | 19.0 steps | 75.98 steps | 66.95 steps |
| Value Iteration | 11.31 steps | 18.23 steps | 34.98 steps |

# Behavior

# Summary

# References

## Data Sources