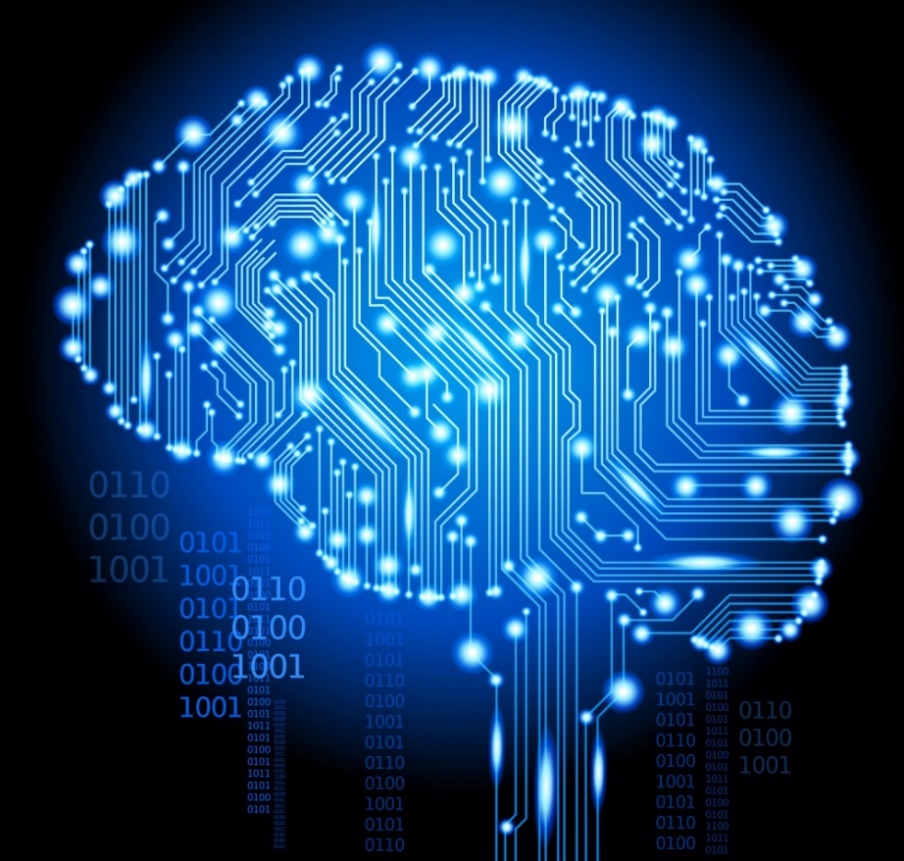


COS30018 – Intelligent Systems

**REPORT**

TASK B7 – EXTENSION – STOCK PREDICTION WITH REINFORCEMENT LEARNING



**TABLE OF CONTENTS**

*Table of contents…………………………………………………………………………………………………………………2*

*I. Agent……………………………………………………………………………………………………………………………..*

*II. Training the agent………………………………………………………………………………………………………*

*III. Evaluating the agent…………………………………………………………………………………………………*

*IV. Output visualizing……………………………………………………………………………………………………….*

*III. Conclusion……………………………………………………………………………………………………………………16*

**AGENT**

Intially, I defined a class named “RLAgent”, which is a Reinforcement Learning agent using Q-learning.

* Initializing method (\_\_init\_\_): This constructor takes several parameters:
  + “stateSize” – the dimension of the state space,
* “getModel()” method:
  + Defines and complies a Neural Network model using Keras
  + Has an input layer with “stateSize” input dimensions, followed by two layers with the units numbers of 32, 8 respectively
  + A output layer having “actionSize” units and a linear activation function is added
  + The loss function is calculated using MSE (mean squared error), then optimized with Adam optimizer (learning rate is 0.001)
* “act()” method:
  + Takes “state” as the input, while returns the action
  + If a randomly generated number is less than “epsilon”, a random action would be taken
  + Else, it use the existing neural network model to predict the Q-values for each action in the given state, then the action with highest Q-value would be returned
* “experienceReplay()” method:
  + Is used for experience replay – a technique in Reinforcement Learning helping to scale and improve the training process
  + Takes “batchSize” as the input
  + Samples a minibatch of experiences from the memory of the given agent
  + For each of those experiences, it calculates the target Q-value based on the Bellman equation



* + After that, it updates the Q-value in the neural network for the chosen action using the calculated target Q-value
  + If the “epsilon” value is greater than the min threshold (“epsilonMin”), it decays the “epsilon” (by multiplying it by the value of “epsilonDecay”)

**TRAINING THE AGENT**