Newsvendor Problem with Reinforcement Learning Approach

1. What is Newsvendor problem?

The newsvendor problem or newsvendor inventory problem is a classic problem in operations management, supply chain and applied economics. There are many variants of newsvendor problem with the situation faced by a business owner who must decide how many daily orders to stock in the face of uncertain demand and knowing that unsold products will be worthless or at least require inventory cost at the end of the day. The business owner needs to decide a sequence of daily orders to maximize the profits based on the observable data such as daily sales and daily inventory, etc.

Even though researchers have developed many traditional mathematical methods to solve the newsvendor problem, there is a hard problem turns out to be intractable for analytical solution which is the newsvendor problem with lead time. As we all know, the lead time is the delivery time from vendors to the warehouse. The lead time is difficult to anticipate because of many uncertain issues such as delayed shipment. Moreover, it is also very difficult to figure out the approximate solution by the traditional approaches. As an alternative way, it turns out that RL can find the asymptotic solution quickly and efficiently for the newsvendor problem. Therefore, I decided to develop a RL model to figure out the best sequential daily orders.

A well trained RL can find out the optimal sequential daily orders for the business owner very fast, which can save a lot of money for the business owner. However, how to train the RL and with what kind of data is the first thing we need to resolve. As one of the most powerful simulation models, the discrete event simulation (DES) is a method used to model real-world systems that can be decomposed into a set of logically separate processes that autonomously progress through time. Therefore, we decided to develop a discrete event simulator to represent the real-world newsvendor business and feed the outputs of the simulator as the training data for the RL.

2. How to shape the simulator to fit the reinforcement learning (RL) frame?

To address the intractable newsvendor problem with RL, we need to consider and shape the newsvendor simulator with the lens of RL frame. Recently, more research began to focus on the RL approach for newsvendor problem, but few of them articulated their relationship well. In this report, we would like to start from a simple newsvendor problem and explain how Q-learning algorithm and Deep Q-learning network (DQN) solve this issue.

A well-trained RL model can find an optimal way for a focal agent to interact with environment based on a redefined value function. The focal agent can receive state data from environment and take actions based on the state data. The agent can also receive a reward for taking each

action or a sequence of actions. The purpose of the agent is to maximize its redefined value by taking a sequence of actions. RL is a good algorithm to find this sequence of actions.

In this newsvendor problem, we have many observable features, to simplify the problem and control the size of the subsequent Q-table, we select the daily sales and inventory volumes as the state variables, since these two variables are the most important variables that can be observed by the business owner. The selected features are as follows:

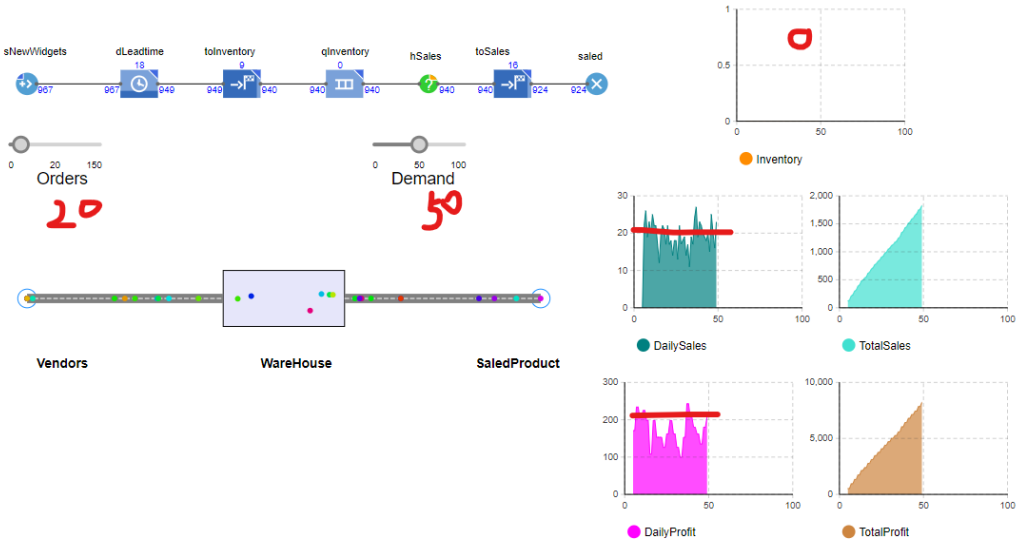
State: Daily Sales, Inventory

Action: Daily Orders (start from a random default value and business owner can only choose to either increase one-unit order or decrease one-unit order or keep the same amount of daily order)

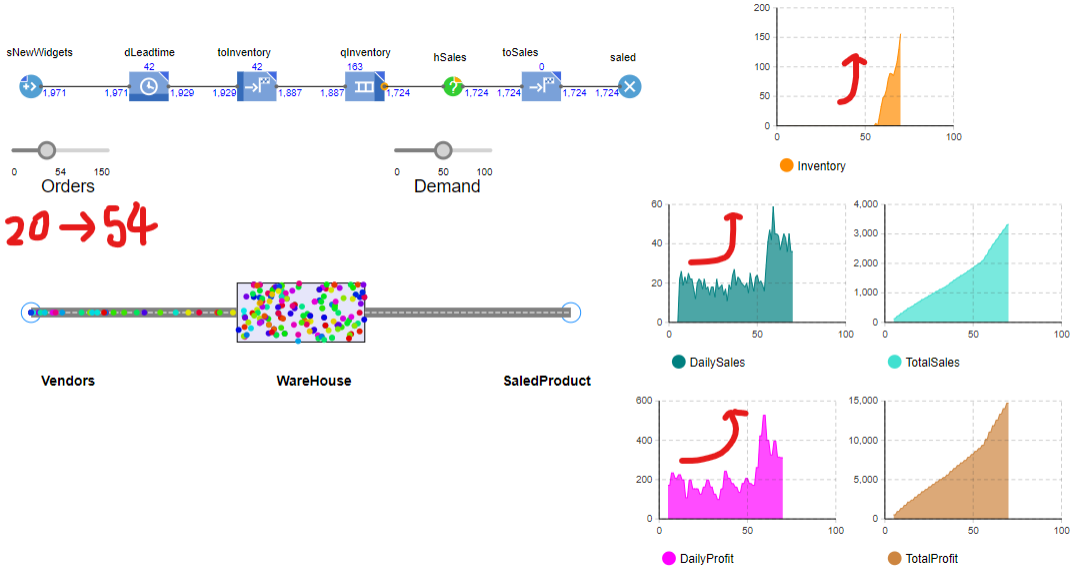
Rewards: Daily Profits

We can consider the newsvendor problem as a game and the business owner as the player, the amount of daily orders as the action of this game, the amount of daily sales and inventory as the observable state variables, the daily profit as the reward of this game. The target of this game is to maximize the daily profits. We can even set up a target daily profit value as the terminal of this game.

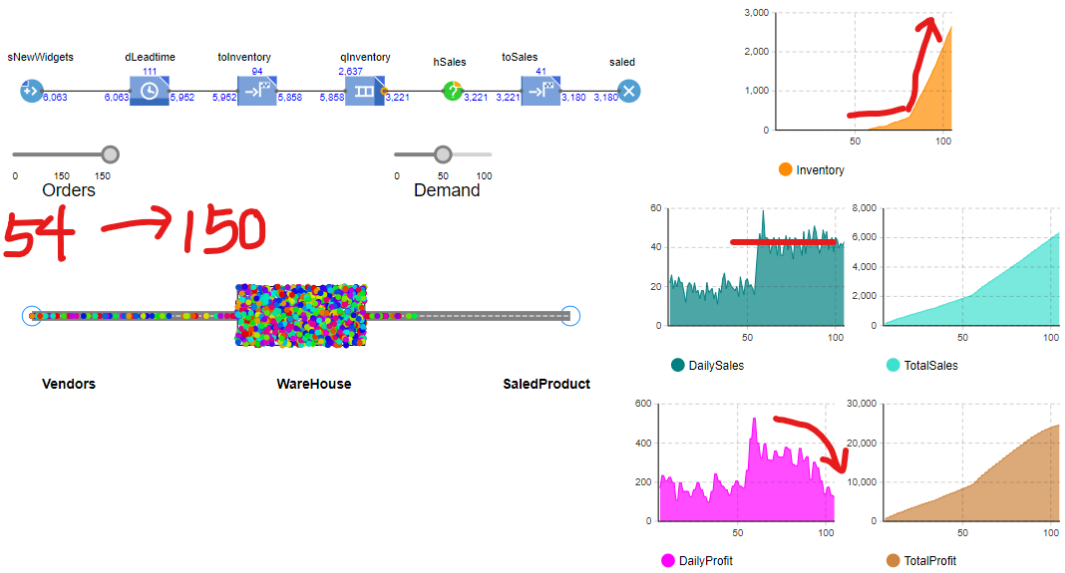
There is an unobservable variable demand in this game, to win the game, business owner should choose the daily order close to the demand. In our simulator, the daily demand follows a Poisson distribution with mean 50. The default daily order is 20. At the beginning of this game, the daily sales is around 20, inventory is 0, and the daily profit is around 200 as chart1 shows. Consider the empty inventory, business owner would increase the orders, since the order amount hasn't approached the potential demand value.



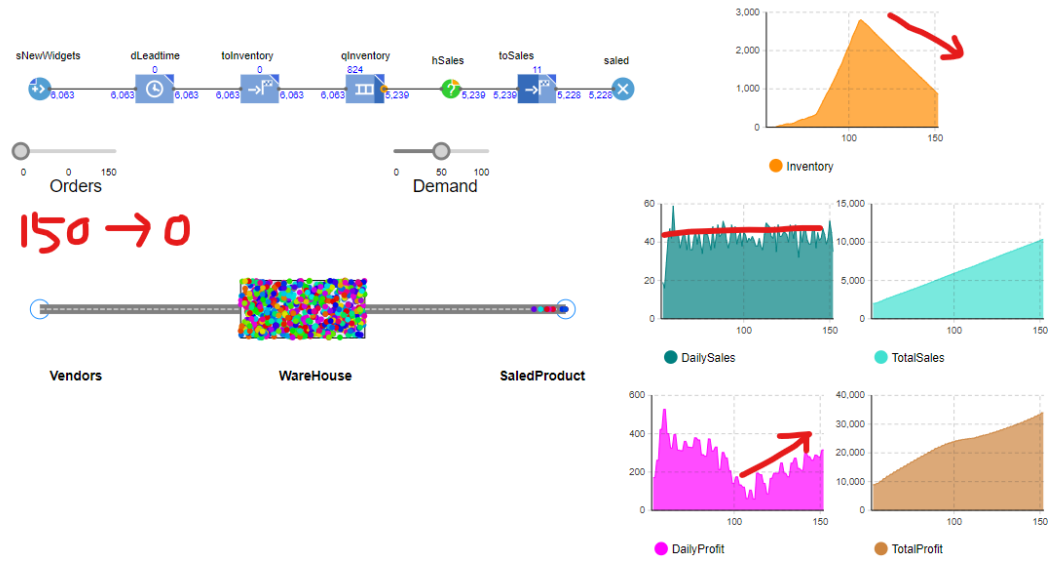
So, the business owner would increase the daily order amount to around 50. During this time, daily sales and daily profits both have a sharp increase and inventory appears as the Chart2 shows.



Suppose the business owner continued to increase the daily order, for example, the business owner increased the daily order from 54 to 150. You will find the inventory has a sharp increase, even though the daily sales kept nearly the same, the daily profits decreased dramatically because of the inventory cost. To avoid the decrease of daily profits as Chart3 shows.



To avoid the decrease of daily profits, the best strategy for the business owner would be stop placing orders from the vendor. So as Chart4 shows, once the business decreased the daily orders to 0, the inventory will decrease and the daily profits will increase again.



The business owner as the game player will adjust the daily orders back and forth based on the observable daily sales and inventory until he finds a maximum profit value or approach the target daily profit value. However, what is the fastest way to approach this target? To approach this target as soon as possible will save a lot of money for the business owner. So how to quickly find a sequence of actions to approach the game target becomes a crucial mathematical problem. The traditional mathematical approaches such as dynamic programming does not have a perfect performance especially facing the problem with lead time. So, let's talk about how to solve this problem with RL. For your reference, you can find the simulator file here:

NewsvendorLostSale.alp

3. How to solve newsvendor problem with RL?

The game we set up is very simple. A simple Q-learning is effective enough for such simple policy function. Also, the newsvendor simulator is a good fit for the Q-learning algorithm, because as a greedy algorithm, Q-learning will store all possible values in each state dimension. Our problem only contains integer states, so we can easily control the size of our Q-table. The RL attempts to figure out the policy function f as shown in the below chart5 and based on the policy function, we can easily generate the Q-table as well. This policy function f gives us the best action strategy we should take under each observable state.

Inventory / Sales	0	...	500
0	(0, 0)	...	(0, 500)
		...	
100	(100, 0)	...	(100, 500)

Action1: Add one-unit daily order (+)

Inventory / Sales	0	...	500
0	Q (0, 0, +)	...	Q (0, 500, +)
		...	
100	Q (100, 0, +)	...	Q (100, 500, +)

Action2: Remain daily order (*)

Inventory / Sales	0	...	500
0	Q (0, 0, *)	...	Q (0, 500, *)
		...	
100	Q (100, 0, *)	...	Q (100, 500, *)

Action3: Reduce one-unit daily order (-)

Inventory / Sales	0	...	500
0	Q (0, 0, -)	...	Q (0, 500, -)
		...	
100	Q (100, 0, -)	...	Q (100, 500, -)

The Bonsai brain is based on a DNN, which is too complex for this problem, thus it will be prone to overfitting. Therefore, I decided to develop a Q-learning algorithm from stretch. Generally speaking, Q-learning algorithm is very concise and intuitive. It calculates the Q-values as the entries of the Q-table as shown on the small tables above. Q-learning will randomly initialize the Q-table and update the Q-table by the following iterative formula derived from Bellman equation. Note that S, I, P, O stands for daily sales, daily inventory, daily profits and daily order action. Alpha is the learning rate, gamma is the discount rate.

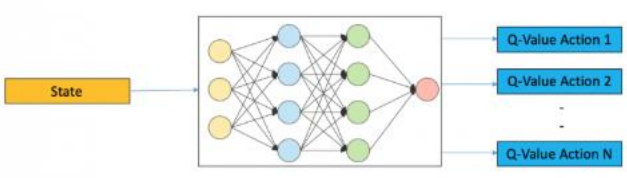
$$Q^{new}(S_t, I_t, O_t) = (1-\alpha)Q(S_t, I_t, O_t) + \alpha\{P_t + \gamma Max_O Q(S_{t+1}, I_{t+1}, O)\}$$

When we complete updating the Q-value or Q-table, our RL training step is done. What we will do to leverage the RL model is that randomly set a daily order value and mean value of Poisson distribution of demand. Once we start simulation, our simulator will take daily actions according to the Q-table and approach the daily profits target very fast. Therefore, by using simulation and Q-learning algorithm, we successfully teach the newsvendor simulator how to find an optimal strategy to maximize the profits! This is really a fast and easy approach compared with many

traditional methods! For your reference, you can find the Q-learning python script developed for our newsvendor problem here, more detailed parameter descriptions are also included in the following file, such as learning rate, discount rate, episodes, etc:

Newsvendor_Q-learning.ipynb

What if we take more observable variables as input states into our consideration to enhance our RL model? For example, the average lead time for the product delivery from vendors is a good choice. However, if we involve more state variables, the increasing dimension of state space will make the Q-table very huge until the it exceeds the memory's capacity. Therefore, the Q-learning algorithm is not effective for large state-dimension problem, in such case we should consider using a neural network to find the Q-value function. The structure of the deep Q-learning networks is as follows. We use a neural network to approximate the Q-value function. The state is given as the input and the Q-value of all possible actions is generated as the output.



Based on the current simple newsvendor problem, we developed a python script for training the deep neural network with the same structure as above by Keras. I will take the Deep Q-learning method as my future work and apply to different complicated newsvendor problems and check the performance.

Newsvendor_DQN.ipynb

4. Future Work

The purpose of this project is to explore the cons and pros of RL approach of Newsvendor problem compared with the traditional mathematical approaches. Our next step is conduct some experiments where we would like to increase the uncertainty and randomness of the system and test whether the solution of Newsvendor problem (best sequence of action strategy for the business owner) delivered by RL is better than delivered by traditional methods with the increase of system uncertainty.