STRATEGY LEARNER

FALL 2018

ALON AMAR
aamar32
aamar32@gatech.edu

Learning

Q-Learner

In order to utilize the q-learner I built in the previous assignment, I needed to define 4 things:

1. **States:** The only parameters my learner is using are the indicators. Since the values of the indicators are continuous, I needed to discretize them to have a finite number of states. The number of steps chosen to discretize them define the number of total states we have by: $\#states = steps^{\#indicators}$. The number of steps were picked while having 2 thoughts in mind:
    a. If the number of steps is too small, then our learner couldn't differentiate between the different values of the indicators and will perform poorly.
    b. If the number of steps is too big, then our learner will take a very long time to converge in our time frame, as the q-table will be big.

    Eventually, I picked steps=80 and used the 2 indicators from the previous assignment, thus creating 6400 states.

    The state itself is calculated by the formula of calculating a value of a number with a base of 80: s = ind1 * steps + ind2. For example: To get the last state, the formula will be –

    s = 79 * 80 + 79 = 6399 (The location in our q table, as it starts from 0).

    * Note: I considered our position (short/long) as one of the states, as it describes the position we are currently in, but as explained by Prof. Balch[1], it does necessarily add additional information about the next optimal action and increase our number of states.

2. **Reward:** The reward was defined as the immediate reward we get - meaning the daily return by percentage multiplied by the number of stocks as such:

$$reward = \left(\frac{price[i]}{price[i-1] * (1 \pm impact)} - 1\right) * \#shares$$

    The sign before our impact factor is determine by the action we have taken – buy/sell, while doing nothing does not take impact into consideration.

    The number of shares can be: 0, ± 1000. Given us a positive reward when we chose to go long/short correctly.

3. **Actions:** The actions that can be made by our learner are Long/Short/Cash. Each action describes the amount of shares we ought to hold. That way we can even describe the action of buying/selling 2000 shares, since we look at the difference between the previous trade and the current one.
4. **Convergence:** I defined convergence when the data frame of trades stayed the same – meaning our policy converged. In addition, I set a minimum value of iteration of 10.

Once I had those key building stones, I implemented the q-learner as described in the Q Trader Hints[2] page.

[1] https://youtu.be/K8xRATOpsqw?t=47m55s
[2] http://quantsoftware.gatech.edu/Q_Trader_Hints

The indicators:

I chose the 2 indicators I used in the previous assignment:

1. BBP - Bollinger Band® percentage (10 days backwards)
2. RSI of OBV (14 days RSI)

As discussed above, I discretized them using 80 steps to get an optimal balance between minimum information loss and short converging time.

I did not adjust the data, as I wanted to compare the same parameters for indicators, and leave the manual strategy as is to perform well.

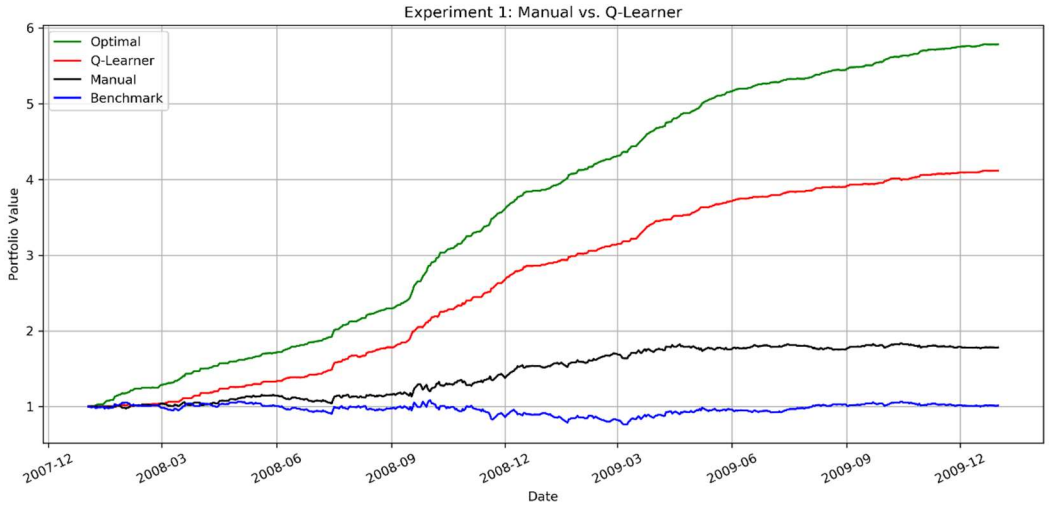
Experiment 1

In this experiment I took the exact same manual strategy from previous assignment and compared it to the current learner. I used the same indicator's windows for both. The impact used is 0.005 and commission is 0.

For the q-learner I used alpha=0.2, gamma=0.9, rar=0.98, radr=0.999, without dyna.

I canceled the random action selection for the test policy stage, as it should be consistent over multiple runs.

In addition to the two strategies, I calculated the benchmark and optimal strategy to serve as a measure of reasonable values.



	Optimal	Q-Learner	Manual	Benchmark
Sharpe Ratio	11.3453	8.0689	1.642	0.156
Cumulative Return	4.7837	3.1148	0.7809	0.0123
Average Daily Return	0.0035	0.0028	0.0012	0.0001
Standard Deviation	0.0048	0.0055	0.0117	0.017

As we can see, the q-learner outperformed the manual strategy in a significant way.

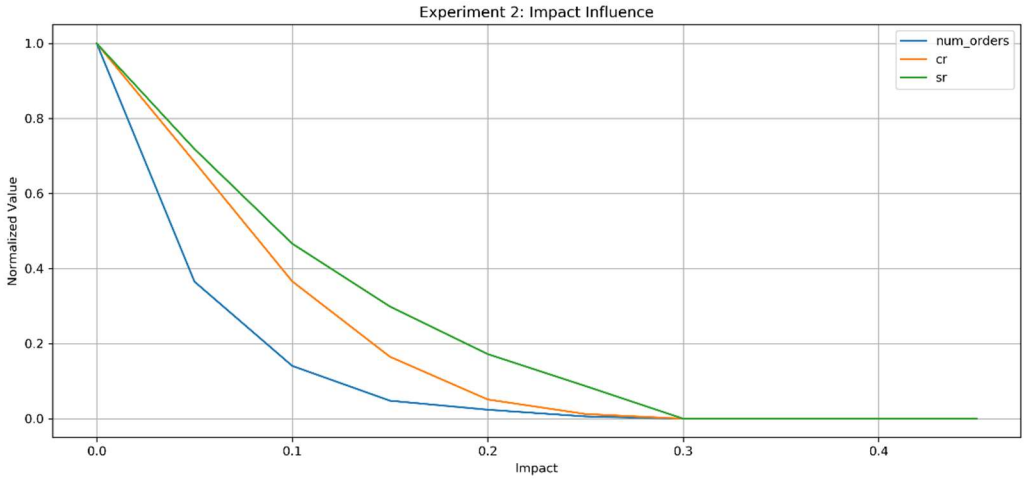
It is closer to the optimal strategy than the manual. That kind of behavior is to be expected to in-sample data, since our q-learner tries to get the maximum reward and optimize our trades. The q-learner tries various tactics regarding our indicators, while the manual was checked manually by me, and was set to certain values. The reason the q learner didn't achieve the optimal value is that he's basing his decisions on the indicators and not the price itself, as the indicators might cause an uninformed decision.

Experiment 2

The impact is the amount the price moves against the trader compared to the historical data at each transaction. Meaning that a higher impact will lower the reward for each trade, causing each trade to be costlier. Thus, **reducing the number of trades executed in our strategy.**

Lower reward also means lower return to our portfolio, causing **the values of Cumulative Return and Sharpe Ratio decrease as the impact increase**. In addition, fewer trades also means we largely based on the price of stock itself (Like the benchmark) or will just have lower values since our gain is based on few short-term trades.

I ran my learner on several different values of impact, range of 0-0.45 with intervals of 0.05, while showing the values of the SR and CR, as well as the number of trades that were made. I used the 'JPM' symbol and used the in-sample data.



* Note: All values in the graph are normalized.

As we can see, all values are decreasing as we increase our impact value. In the extreme case of zero trades, there is no return for our portfolio.

We can see in the table that it's not worth trading when impact ≥ 0.3. An impact of 0.3 means that unless our return is more than 30% for each trade, it will not be worthwhile to trade.

Impact	#Trades	SR	CR
0.0	334	7.9603	3.2896
0.05	122	5.7210	2.2516
0.1	47	3.7131	1.2042
0.15	16	2.3789	0.5429
0.2	8	1.3705	0.1683
0.25	2	0.6924	0.041
0.3	0	0	0
0.35	0	0	0
0.4	0	0	0
0.45	0	0	0