Strategy Learner Report

Noam Lerner

I created my Strategy Learner as a wrapper around the Qlearner I created in a previous assignment. The Strategy Learner I created takes stock data and maps it into a format the Qlearner can learn from and act on. In order to do this, I first had to understand that the Qlearner essentially works by memorizing states it has seen before, and then acting in the way that was most optimal in the past. So in order to map the trading problem so that the Qlearner could understand it, I had to map my trading data into a state.

I chose to represent my state as an integer where every digit represented some important piece of information. I used the same indicators as I did for my Manual Strategy, and these were: Bollinger Bands, RSI and momentum. The state I gave to the Qlearner is a 5 digit number. The first digit can take on a value from 0-2, the second digit can take on a value from 0-8 and the last 3 digits can take on values from 0-9. This allows for a 10 * 10 *10 *9 *3 = 27,000 different possible states.

The last digit represents the current state of portfolio. It can take on the value 0 1 or 2 where 0 represents a position of -1000 shares, 1 represents a position of 1000 shares and 2 represents a position of 0 shares. This is meant to help the Qlearner understand impact. By providing this data, I intended for the Qlearner to see that selling it is holding stock can be more detrimental than holding.

The second to last digit represents momentum. The momentum for the current day is calculated with a 2 day window. In order to discretized momentum, I first calculate the mean and standard deviation for the entire in-sample. Using these two values, the z_score is calculated for the momentum on any given day using the formula

zscore = (momentum – average_momentum)/(std_momentum).

To get the state I then multiply the zscore by 3, I add 5 and round to the nearest integer. Adding 5 causes the values to center around 5 instead of 0, which allows me to represent negative values with the range 0-4 and positive values with 5-9. Multiplying by 3 spreads the values out so that instead of 95% of the values lying between 3-7, they lye between -1 and 11. If a number is greater than 9 or less than 0, I round to 9 or 0. While some data is lost at the edges, these are extreme cases and will rarely be seen. Using this method has the benefit of giving the Qlearner meaningful data to work with while still discretizating it. This data can be detrimental if the Qlearner is not retrained often enough due to the possibility of a changing mean and standard deviation. This would work best if the Qlearner is retrained with every piece of data it receives, or for stocks that remain consistent in their movements.

The 2nd and 3rd digit are the current RSI, and the RSI for the previous date over a 5 day window. Both are calculated by taking the RSI and dividing it by 10 and rounding down. Since the RSI is a value between 0-100, I am always guaranteed an integer between 0 and 9. In my manual strategy I found that using the last RSI and the current RSI was useful, since it is best utilized when one is above a threshold and the other is below, by providing both values on a scale, the qlearner can differentiate this for itself.

The 1st digit is taken up by Bollinger bands. Bollinger bands tend to signal a buy when the price goes through one (from bellow the bottom band to above it, or from above the top band to below it). In order to provide the necessary information to help the Qlearner figure this out, I discretized this information into two values; current_price_state and last_price_state. For either of the given prices, the digit can take on a value of 0 1 or 2 where 0 represents that the price is below the lower band, 1 represents that the price is between the two bands and 2 represents that the price is above the upper band. The integer which is used in the state is then calculated using the expression

current_price_state x 3 + last_price_state

which provides a unique state for each of the combinations of the two prices.

Throughout the process of training I keep track of the portfolio value. Every time the Qlearner takes an action, I calculate the resulting gains using the expression

gains=(stocks bought) x (change in price from the day the action was taken to the next day) x (-1 if shorting, 1 if longing).

This value is used twice, once when it is added to the current portfolio value and the second time as the reward for the Qlearner on the next iteration. If a non-zero impact is passed in to the Qlearner, impact_loss is calculated using the expression

impact_loss=(price of stock) x (impact) x (number of stocks bought).

Anytime the Qlearner buys or sells stocks, the gains is updated using the expression

gains = gains – impact_loss

before it is used for the reward or portfolio value-updating.

The total portfolio value is used when training. The Qlearner will be retrained on the evidence provided for either 20 iterations or until the cumulative return is unchanged from one iteration to the next, whichever comes first.

Experiment 1

For this experiment, I trained my Strategy Learner on the in-sample which was the JPM stock from January 1st 2008 to December 31st 2009. I then allowed the Strategy Learner and my Manual Trader from a previous project to make trades on the same date range. I plotted the portfolio values of the two along with a benchmark of buying JPM and holding for the time range in figure 1.

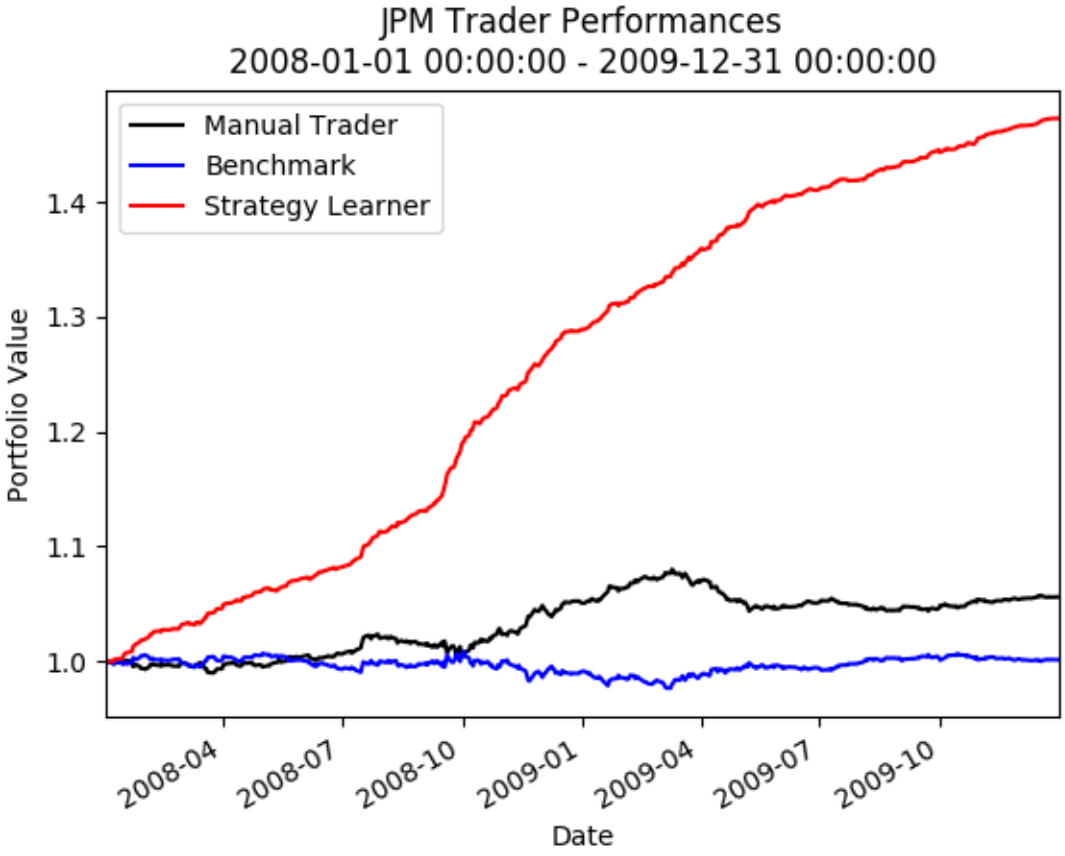

Figure 1, A plot of how the manual trader, strategy learner and bechmark perform on JPM for 01/01/08 – 12/31/09

As can be expected, the Strategy Learner massively out performed the benchmark and the manual trader. Table 1 shows their performance statistics

	Strategy Learner	Manual Trader	Benchmark
Cumulative Reward	0.473	0.0517	-0.00379
Mean of Daily Returns	0.00077	0.00010	-6.2215e-6
Standard Deviation of Daily Returns	0.0011	0.0016	0.00162

I would expect the Strategy Learner to out perform the Manual trader in every experiment where they can both be run on the in sample. The reason for this can be explained by understanding how they both work internally. Manual Trader works based on a few hard coded heuristics which are generalized to work for the data. The Strategy Learner on the other hand, works by essentially memorizing the best thing it could have done whenever it sees one of 27,000 possible states. That means every time it sees a state, it has already seen the state in training and has figured out what the best action to do in that given state is.

Experiment 2

I designed an experiment to observe how impact will affect the Strategy Learner. My experiment consists of two Strategy Learners. One trader, named Impact_Trader is informed of an impact value of 0.05, and one, named Free_Trader is not. Both are trained on the same data set (the JPM stock from January 1st 2008 to December 31st 2009). They are then both told to trade over the same time period, once with an impact of 0 and once with an impact of 0.05.

When the impact is increased, it causes trades to be more costly. I hypothesize that the Impact_Trader will trade significantly less than the Free_Trader. I also hypothesize that the Impact_Trader will maintain a positive cumulative return on both runs while the Free_Trader will suffer losses when an impact is present because it will have made too many trades. When impact is not present, Free_Trader should provide a significantly better cumulative return since it is taking advantage of the unlimited amount of trades available to it, while the Impact_Trader is not. Both Free_trader and Impact_Trader should perform better on the simulation when impact is 0, since trading costs less.

Results

	Trader Aware of Impact	Trader unaware of Impact
Cumulative Return When impact is 0.05	0.0518216789996	-0.494330845381
Cumulative Return When Impact is 0	0.37343	0.47201
Number of Trades	97	269

It is obvious from the table above that the presence of an impact had a negative effect on both traders. Neither was able to perform as well as it could have without an impact. On the other hand, Free_Trader made almost 3 times the amount of trades as Impact_Trader, and ended up losing roughly half it's starting value on the run with impact. The Trader that was aware of the impact made only 97 trades and managed to earn money – although significantly less than when there was no impact at all.

Both were able to earn a significant amount when the impact was not present, but the Free_Trader still earned more. This is because the Free_Trader made trades at every opportunity to earn money, and the Impact_Trader only made trades when the amount of money earned would beat what it perceived the impact to be.