ML4T – Strategy Learner

This project is to design a trading robot that develops trading strategy by learning from the empirical experiences. My strategy learner uses random forest, which consists of a multitude of random decision tress, to train a classification learner by exploiting the indicators constructed from the stock prices.

1. Random Forest Classification Trading Learner

Random forest operates by constructing a bunch of random decision trees at training time. Random decision forest reduces the model variance, hence effectively corrects the overfitting issues. It repeatedly samples from the training data, and train a random tree learner based on the sample data once a time. After the recursion of random trees training, random forest averages the bunch of learners, thus neutralizes the overfitting effect that results from relying on single learner. We employed the random forest with 10 random decision trees featuring leaf size of 5.

To develop the random forest trading robot, I used the same technical indicators as we introduced in the Manuel Strategy project. The 4 technical indicators, which serves as the independent variables, are listed as follows:

- Price/SMA indicator, which compares the stock price with the 20-day moving average;
- BB indicator, which compares the stock price with the Bollinger Bands with 20-day moving average and $\pm$ two 20-day moving standard deviation;
- $MACD(12,26,9)$ indicator, which is the difference between the MACD line and the signal line; and
- RSI indicator with a 14-day moving window.

There are 3 types of classification:

- +1 Buy;
- 0 Hold; and
- -1 Sell

The classification indictor is based on the 5-day return of the underlying stock. The construction of the classification indicator is formulized as follows:

- Firstly, we calculate the forward-looking 5-day return of the underlying stock.
$$r(t) = \frac{stock(t+5)}{stock(t)} - 1$$
- Then, we adjusted the 5-day return by considering the impact factor.
$$adjusted\_r(t) = \begin{cases} \max(r(t) - 2\times impct, 0) & if\ r(t) > 0 \\ \min(r(t) + 2\times impct, 0) & if\ r(t) < 0 \end{cases}$$
The impact plays a role in diluting the return of trading the underlying stock. A high impact may discourage the trading frequency and an increasing/decreasing stock is not necessarily profitable unless the stock can cover the cost of market impact. For example, when the 5-day return $r(t)$ is negative, ideally, we can short the underlying stock and hold the position for 5 days, earning exactly the return of

- $r(t)$. However, given the presence of the impact factor, realistically, we cannot earn the profit as much as - $r(t)$. Therefore, we adjust the return by 2 times impact, as it requires two transactions to complete one trading strategy.

- Subsequently, we compare the adjusted return to the predetermined threshold. If the 5-day adjusted return exceeds a certain value, we classify the sample as +1 "Buy". Conversely, if the 5-day adjusted return falls below a certain value, we classify the sample as -1 "Sell". And we classify the rest as 0 "Hold". In the strategy learner, the threshold is set as $\pm$ 1%

Note the stock we use is the JPM stock between January 1, 2008 to December 31, 2009.

The outcome, which is the holding position is derived by the random forest trading robot. As the random forest algorithm takes the average of the outputs of random tress. The output of the random forest classifier ranges between -1 to 1. Therefore, we come up with a threshold to transfer the continuous output into the classification. If the output is above 0.2, we classify it as +1 "Buy" and we hold +1000 position of the underlying stock; if the output is below -0.2, we classify it as -1 "Sell", we hold -1000 position of the underlying stock; otherwise, it is classified as 0 "Hold", and we hold the same position as the previous day.

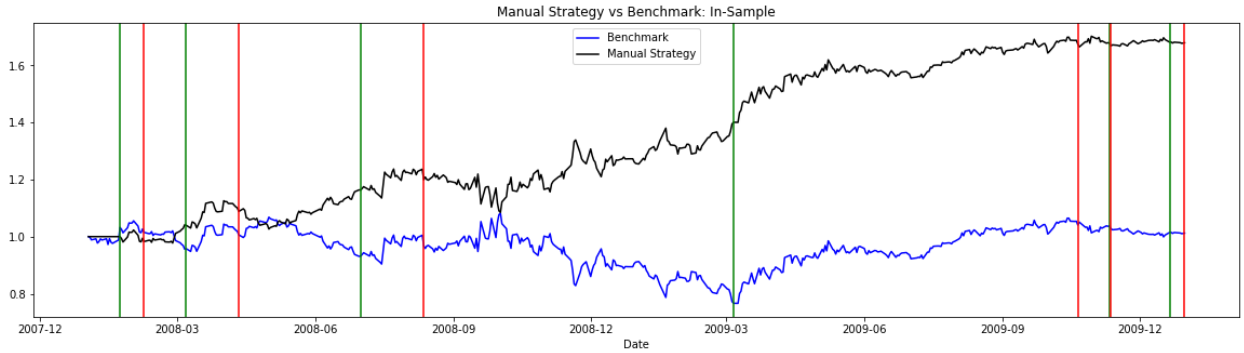
2. Experiment 1

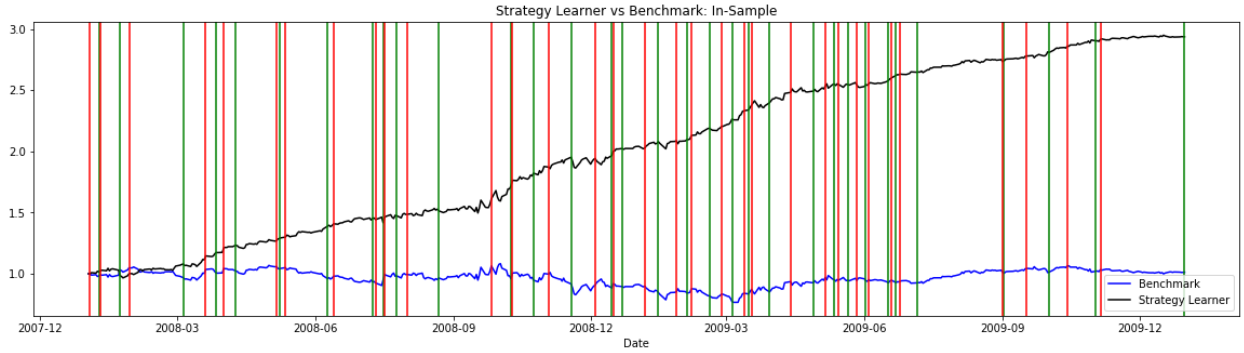
In the first experiment, we test our strategy learner against the Manuel strategy and the benchmark.

- Strategy Learner: as described in the section 1
- Manual Strategy: this strategy determines the positions by examine all technical indicators. For example, if the majority indicators refer to a LONG position, we will go for long and vice versa.
- Benchmark: buy and hold +1000 position from day 1

In this experiment, all the trading strategy is based on the JPM stock between the in-sample period, which is between January 1, 2008 to December 31, 2009. Besides, we didn't incorporate any impacts and commissions fee. Also, we assume only three positions in this experiment, which are long +1000 stock, cash and short -1000 stock position and the start value for each strategy is $100,000.

Graphs below illustrates the comparison between these three strategies. Y axis presents the is the normalized value of portfolio. Green vertical lines indicate the long position while the red vertical lines indicate the short position. We can observe that both of manual strategy and strategy learner outperforms than the benchmark. Besides, the strategy learner trades way more frequently and has a better performance than the manual strategy. And as shown by the table below the strategy learner has higher cumulative return, average daily return and lower standard deviation of daily return in the in-sample period.





	Benchmark	Manual Strategy	Strategy Learner
Cumulative Return	1.23%	63.82%	193.66%
Mean of Daily Return	0.0168%	0.111%	0.218%
Standard Deviation of Daily Return	1.705%	1.259%	0.938%

The results from the first experiment could be expected in any training datasets, as the random forest model learn from the sample data, aiming at achieving the best performance (i.e. maximize the profit) based on the training period. Besides, given no impact and transaction fee in this hypothetical environment, there is nothing limits the trading frequency. Therefore, the strategy learner will always overshadow the manual strategy in the in-sample period

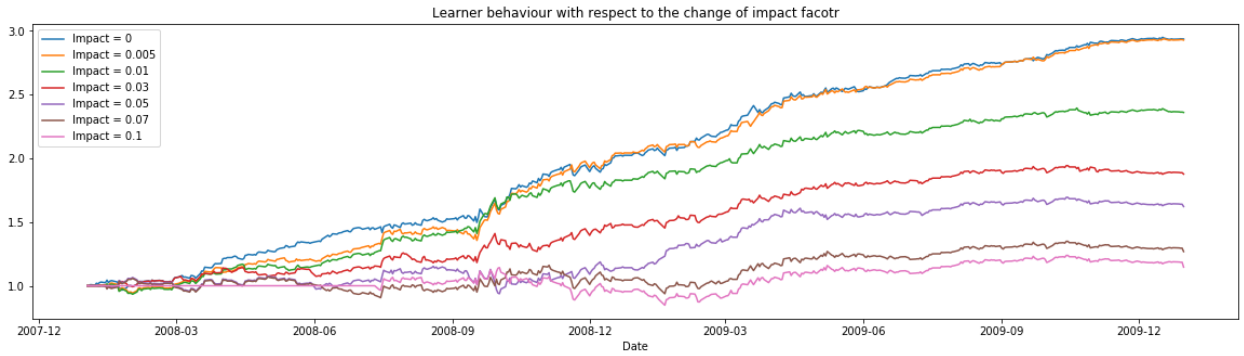
3. Experiment 2

This experiment investigates the behavior of the trading robot with respect to the change of impact factor, which is a charged when executing a trade. A high impact may discourage the trading frequency and an increasing/decreasing stock is not necessarily profitable unless the stock can cover the cost of market impact.

In this experiment, we shift the impact factor from 0 to 0.1, and re-train the trading robot given different level of impact factor. Note that all the trading strategy is based on the

JPM stock between the in-sample period, which is between January 1, 2008 to December 31, 2009.

The graph and table below illustrates that, the trading frequency and the profit are declining with respect the increase of the impact factor. The trading robot becomes inactive when confronting the environment with higher impact factor. It is reasonable as a high impact will cost more for each trade, so the trading robot hesitates when executing a trade as it will be penalized by the high impact cost.



Impact	Cumulative Return	# of Trade
0	193.66%	66
0.5%	192.94%	65
1%	136.05%	63
3%	87.67%	45
5%	62.16%	22
7%	26.86%	8
10%	14.62%	4