Predicting stock prices using Machine Learning vs Manual Strategy

Implementation of Machine Learning Strategy:

I chose to use a **Classification-based learner**, a **BagLearner** with an ensemble of **RTLearners** (Random Forest Tree Learners) for the trading problem.

These are the following steps I implemented to frame the trading problem,

Steps:

1. Created an instance of the BagLearner with **leaf size** = 5 and **bags** = 40, with an ensemble of RTLearners.
2. Modified the corresponding code in the learners to use **mode** instead of mean to calculate Y values, since this is a classification problem.
3. In the **training phase** of the Strategy Learner, calculate the values for the indicators –
   - Price/SMA ratio,
   - Bollinger Band Percentage (BBP)
   - Momentum

   using the **lookback** window of 14 days (the same ones used in the Manual Strategy project). Here, I pull data from the past to calculate indicator values for the first 14 days also.
4. Calculate the Nday future returns as,
   N day future returns = Price[today+Ndays]/Price[today] - 1.0
   using a Nday value of 14 days.
5. Compute the trade decisions based on when the Nday future return values reach a specific threshold such as,
   If Nday returns is greater than YBUY + impact, go LONG.
   If Nday returns is lesser than YSELL - impact, go SHORT.
   I used the threshold values for **YBUY** = 0.02 and **YSELL** = -0.02, deduced it by looking at the Nday return values and tuning it according to performance with trial and error.
   We need to take into account the impact while making the trading decisions, since they can have an effect on how the stocks should be traded.
6. Train the Learner with the indicators as **X** data and the trade decisions as **Y** data. The learner classifies the output as: +1(LONG trades), -1(SHORT trades) & 0 (NO trades)
7. The learner takes the **indicators as the features** to look for and the trade decisions as the output generated for those indicator values on that day. Since we use Random Learners here, it chooses a feature randomly while building the decision tree.
8. In the **testing phase** of the Strategy Learner, we calculate the values of the indicators in the given testing period and query the baglearner. The baglearner in-turn queries the Random Learners and gives the output as trade decisions for each day.

9. Since we are allowed to hold a maximum of 1000 shares long or a 1000 shares short, I compute the trades dataframe using the output from the learners, in a similar fashion in the Manual Strategy.

Discretization or standardization of the data was required for the Strategy Learner. For the trading problem, I had to convert the regression learner into a classification learner. The training Y data is calculated based on the Nday future returns, classifying Y to have one of the three discrete values +1(LONG), -1(SHORT) or 0(NO trades). I discretized the data using the following conditions,

If Nday returns > YBUY with impact: trainY = +1(LONG)

If Nday returns < YSELL with impact: trainY = -1(SHORT)

Otherwise, do nothing: trainY = 0 (NO trade)

Experiment 1: Comparing Manual Strategy with Machine Learning Strategy In Sample

Parameters:

Symbol: JPM

Testing Period: January 1, 2008 to December 31 2009.

Start Value: $100,000.

Commission: $0.00

Impact: $0.005

Indicators: Price/SMA ratio, Bollinger Band Percentage (BBP), Momentum

Lookback window: 14 days

Leaf Size: 5

Bags: 40

Ndays: 14 days

I have set a seed for the RTLearner to maintain consistency in results.

Assumptions:

Allowable holdings at any point: 1000 shares LONG, 1000 shares SHORT or 0 shares.

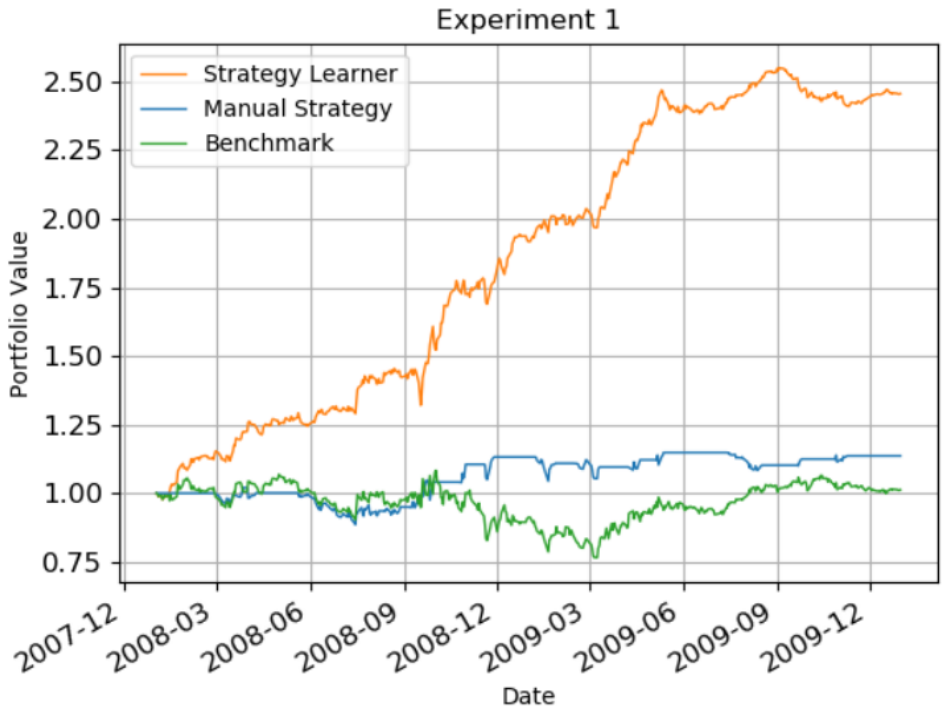
Benchmark: Starting with $100,000 cash, investing in 1000 shares of JPM on the first day and holding that position.

I have used the same values for Manual Strategy and the Strategy Learner.

Outcome:

Looking at the chart below, we can clearly state that the **Strategy Learner outperforms the Manual Strategy**. The strategy learner equipped with the indicators and the trade decisions based on the Nday returns handles the trading problem yielding astounding results. We can understand that this is an expected result especially with the testing being in sample. We can describe the comparison of the results as,

Strategy Learner outperforms **Manual Strategy** outperforms **Benchmark**.



Experiment 1 - Statistics

	Strategy Learner	Manual Strategy	Benchmark
Sharpe Ratio	3.02985757532	0.486858442805	0.1571908415
Cumulative Return	1.45627614717	0.13621	0.0123237046459
Standard Deviation of Daily Returns	0.00958924255858	0.00981804588136	0.0170394293305
Average Daily Returns	0.00183023243632	0.000301111604178	0.000168726001316
Final Portfolio Value	245155.15	113621.0	101037.65

Taking a look at the statistics from the experiment, we can see the Strategy Learner yielding a final portfolio return double that of the Manual Strategy. Even all the other statistics such as the Sharpe ratio, cumulative returns, average daily returns are better in the Strategy Learner compared to the Manual strategy.

Would we expect this relative result every time with in-sample data?

I infer that we should expect the Strategy Learner to outperform the Manual Strategy in sample. This is because when the Manual Strategy employs certain threshold rules to use the indicators in making trading decisions, the Classification Learners are trained using both the data and the indicators and are better equipped to make right trading decisions in sample. Although we should note here that we are using a Random Tree Learner which chooses the indicators randomly while classifying, it might introduce few inconsistencies in results between different periods. Even then the Strategy Learner should be expected to outperform the Manual Strategy in sample.

Experiment 2: How changing the impact value should affect in sample trading behavior?

Parameters:

Same as Experiment 1.

Impact: I have used 3 different impact values to perform the experiment, such as $0.005, $0.015 and $0.025

Hypothesis:

Impact is the costs faced by a portfolio while buying or selling stocks. It is kind of a transaction cost incurred while making a trade in the market.

I hypothesize that, as the **impact** value **increases** the **performance** of the Strategy Learner should **deteriorate**. Considering Portfolio returns as a metric, I deduce that the returns should decrease as the impact value increases. As the next metric, I use the trading decisions. I deduce that the number of trades made must decrease as the impact value increases. The other metrics to note are, Sharpe Ratio and Cumulative Return of the portfolio, which must decrease as the impact value increases.

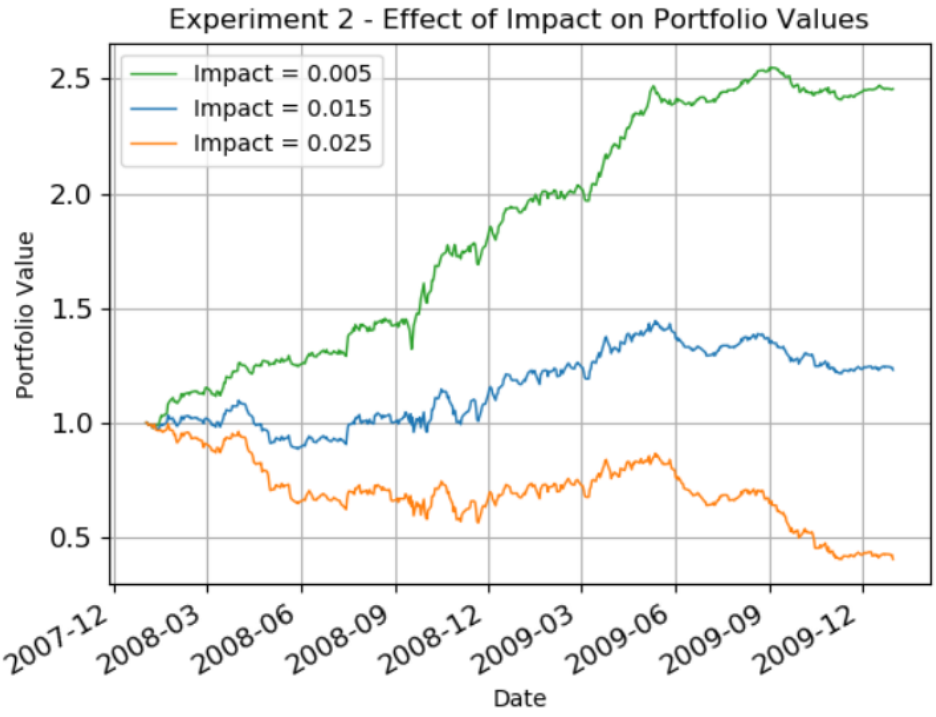
Test the Hypothesis:

1. Train and query the learner with the 3 different impact values.
2. Compute the portfolio values and statistics for the 3 scenarios.
3. Compute the number of days LONG and SHORT trades were made for the 3 scenarios.

Results:

From the graph below, we can clearly see that the performance of the Portfolio decreases as the impact value increases. This is obviously because of the costs incurred while making trades.

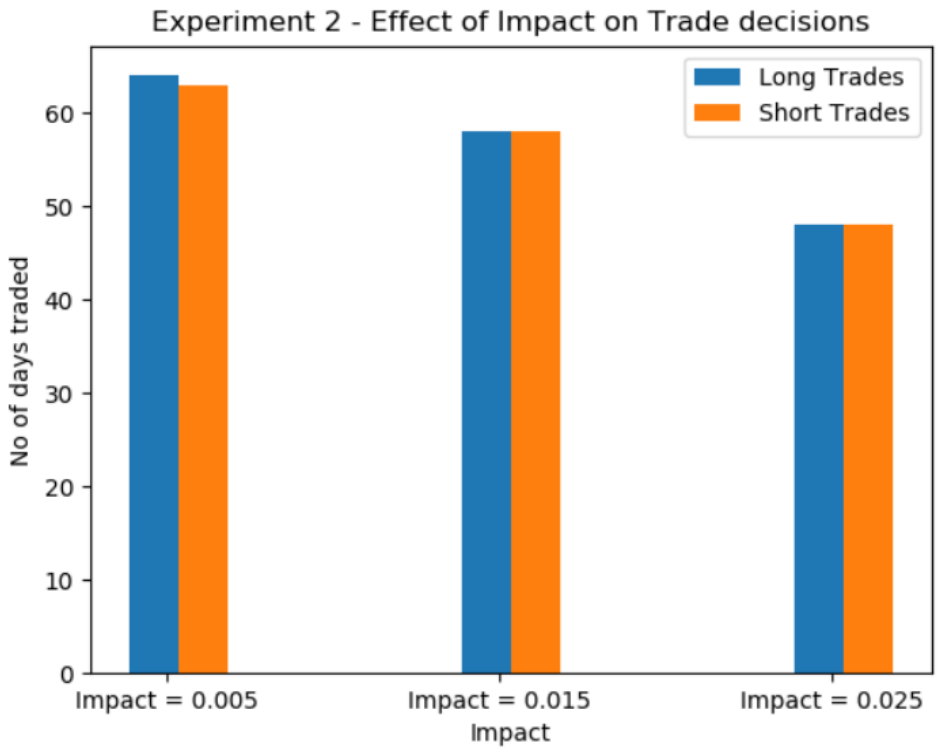
Every time we make a trade of the stock in the market, we face the effect of the impact cost in the portfolio. And if we are not making the right order of trading decisions, we might lose more than the profit we gain due to the impact.



From the bar chart below, we can see that the **number of days traded** decreases as the impact value increases. This is because, in the Strategy Learner's training phase where we form the trade decisions based on the Nday returns, we factor in the impact cost while calculating the threshold for the Nday returns. As the impact value increases, in cases where even if the Nday return indicates that it's a good decision to buy or sell, the effect of the impact affects the trading decision and we end up not trading that day. This leads to lesser trades. We make trading decisions only when the Nday returns accommodates the impact costs also.

The bag of RTLearners are trained using this data, which leads to the learner having lesser trade decisions in the tree. So, when we query the learner, there are lesser scenarios which can lead to trades and in other scenarios taking the mode leads to no trading (since, no trading should be the most occurring output value in the decision tree)

We can also note here that, if the symbol traded is more volatile, that is, if the symbol goes up and down dramatically by huge differences, it might encompass the impact cost during trading decisions and the strategy learner might make more trading decisions. Even then, if we compare the same symbol with a lesser impact value, it should make the most trades.



Experiment 2 – Statistics

	Strategy Learner with Impact = 0.005	Strategy Learner with Impact = 0.015	Strategy Learner with Impact = 0.025	Result
Sharpe Ratio	3.02985757532	0.574370053641	-1.00737076377	Decreases as impact increases
Cumulative Return	1.45627614717	0.230210429282	-0.596466516725	Decreases as impact increases
Standard Deviation of Daily Returns	0.00958924255858	0.0140981806744	0.0238683672169	Increases as impact increases
Average Daily Returns	0.00183023243632	0.000510099138715	-0.00151464823449	Decreases as impact increases
Final Portfolio Value	245155.15	122311.15	39965.25	Decreases as impact increases

When we look at the third scenario where impact = 0.025, we can see that the final portfolio value drops way below. This must be because, as we recall we used threshold values for YBUY = 0.02 & YSELL = -0.02 for making trading decisions to train the learner. The impact value is almost equal to these values. This makes the learner miss most of the promising LONG, SHORT trade combinations, leading to such a poor portfolio value.