Technical Analysis Trading

Jessica Buzzelli

1 INTRODUCTION

In this study, a portfolio managed by a Random Forest algorithm is compared to that of a "manual" stock trading strategy applying simple rules to predefined technical indicators. For the sake of simplicity, the two methods may only trade one asset and they may only take one of three positions at any time: 1000 shares long, 1000 shares short, or 0 shares. Both models will compete against a benchmark of the same initial cash value ($100,000) where the benchmark portfolio purchases 1000 shares of the asset on the first day of the testing period and sells on the final. All analyses comparing the traders uses a market impact value of 0.005 and a trade commission fee of $9.95.

In the case of the Random Forest trader, the underlying learner will first train off of a prior period where buy/sell signals during the training window are manually determined using rolling daily returns. When applied to the testing timeframe, the learner will compare the technical indicators to that of the training window to infer a recommended position (e.g. long or short 1000 shares). No training is required for the manual strategy since the technical indicators will be compared to hardcoded thresholds to determine if a purchase or sale should take place.

The manual strategy has been optimized to outperform the benchmark during the training date range, but its performance on the out-sample will be limited by how reflective the in-sample price fluctuations are of the out-sample. The strategy learner will be beholden to the same limitation, but I further hypothesize that the Random Forest will have a tendency to overfit the training data and, as a result, will perform relatively worse on the out-sample than the manual strategy.

2 INDICATOR OVERVIEW

To account for a variety of price behaviors, three technical indicators were chosen to inform the two trading methodologies. Both strategies used the same technical indicators over the same lookback period of 10 days as determined via trial and error with the manual strategy trader.

In the case of the Random Forest trader, the training dataset was composed of

each indicator's values and the algorithm was responsible for learning which indicator value ranges translated to an advantageous long or short position. For the manual strategy, hardcoded cutoffs were chosen for each indicator to translate the values into long/short signals.

The three indicators chosen were percent of simple moving average (PSMA), Bollinger Band percentage (BBP), and momentum. PSMA and momentum are both measures of a price's deviation from prior patterns whereas BBP is a proxy for volatility.

In Python, the indicators were calculated as follows:

```
sma = prices.rolling(window=n, center=False).mean()
smstd = prices.rolling(window=n, center=False).std()

psma = prices / sma

upper_band = sma + (2 * smstd)
lower_band = sma - (2 * smstd)
bbp = (prices - lower_band) / (upper_band - lower_band)

momentum = (prices / prices.shift(-n)) - 1
```

The signal thresholds used by the manual strategy trader were tuned on the in-sample date range to grow cumulative return and were configured as follows:

Indicator	Buy Threshold	Sell Threshold
PSMA	$\geqslant 0.05$	$\leqslant -0.05$
BBP	$\geqslant 0.80$	$\leqslant 0.20$
Momentum	$\geqslant 0.03$	$\leqslant -0.03$

Table 1—Manual strategy buy and sell thresholds.

3 MANUAL STRATEGY

3.1 Methodology

The manual strategy trader begins by reading in a matrix of each technical indicator on each trading day. From there, the program iterates one day at a time and checks the indicators' values compared to the buy/sell signal thresholds

previously described. The trader then determines if a trade should be made according to the following rule:

```
if current position == 0 shares:
    if PSMA <= -0.05 or BBP <= 0.20 or Momentum <= -0.03:
        if last traded price < today's price:
            sell 1000 shares
            current position = -1000 shares
            last traded price = today's price

    if PSA >= 0.05 or BBP >= 0.20 or Momentum >= 0.03:
        if last traded price > today's price:
            buy 1000 shares
            current position = 1000 shares

if current position == -1000 shares:
    if PSA >= 0.05 or BBP >= 0.20 or Momentum >= 0.03:
        if last traded price > today's price:
            buy 2000 shares
            current position = 1000 shares

if current position == 1000 shares:
    if PSA <= -0.05 or BBP <= 0.20 or Momentum <= -0.03:
        if last traded price < today's price:
            sell 2000 shares
            current position = -1000 shares
```

Since the portfolio should begin and end the trading period with 0 shares, the first and last trades will be either 1000 shares long or short. Afterwards, the trader will only recommend trades of -2000 or 2000 shares to bring the portfolio to a total of -1000 or 1000 at any subsequent point in time. In the market, there may be times at which is advantageous to exit a position for a while (i.e. hold 0 zero shares), but this trading method was too sensitive to over-alerting from the indicators to profit off of a strategy where certain value ranges signaled an exit rather than a hold, especially given trade fees.

Additionally, the above rule incorporates profit-aware trading; by keeping track of the last price the algorithm traded on, the strategy will not leave its cur-

rent position if doing so at that time would incur a net loss. Of course, this implementation does not account for losses due to commission and/or market impact since those parameters are not passed to the ManualStrategy object per the project description.

The manual strategy trader was tuned using the random forest trader's in-sample data; as such, its performance trading J.P. Morgan Chase & Co. stock from January 1st 2008 to December 31st 2009 was stronger than the benchmark, ending with a cumulative return 0.0654 points higher:

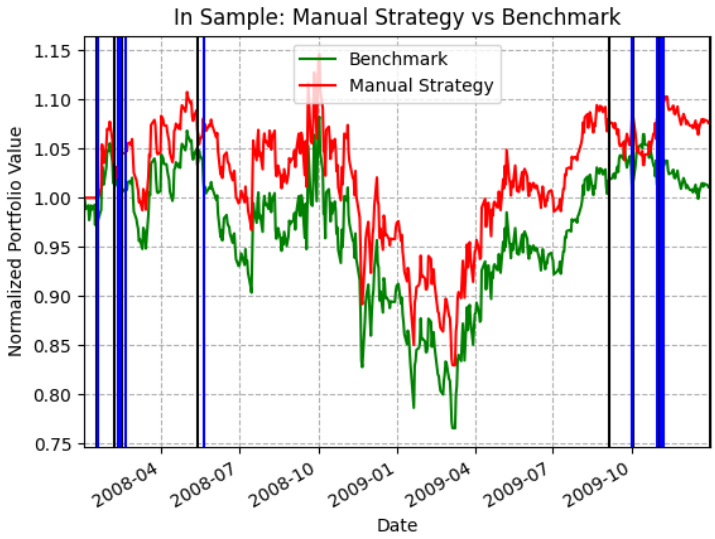

Figure 1—The manual trading strategy outperformed the benchmark throughout the trading period. Blue lines denote days with stock purchases, black lines denote sales.

The trader's success was further extended to the out-sample period of January 1st 2010 to December 31st 2011. While my original expectation was that the model may be overfit to the trends present in the in-sample trading period, the manual strategy trader not only held water against the benchmark, but posted a net gain:

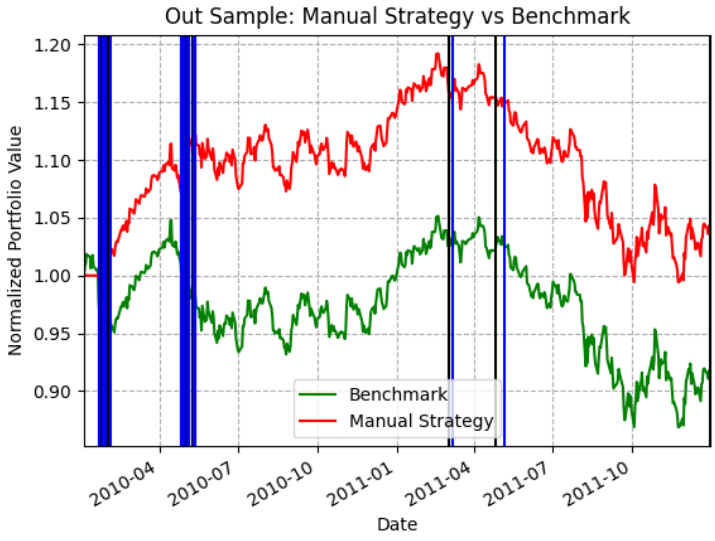

Figure 2—The manual trading strategy outperformed the benchmark and managed to turn a profit over the trading window despite experiencing two windows of rising and falling prices.

Given that the in-sample period contained more noise and quicker volatility than the out-sample, it makes sense that the trader may fare better in the out-sample. On the other hand, the relatively low indicator thresholds that worked well to pick up on minor pattern changes among the in-sample prices should not have been so extensible to the out-sample; for this reason, I attribute the success of the trader to the price-aware trading rules more-so than the choices of indicators or the thresholds used to convert them to buy/sell signals.

Model	Cumulative Return	Avg. Daily Return	Std. Dev. of Daily Returns
Manual Strategy	0.0754	1.955E-5	0.0158
Benchmark	0.0102	-0.0001	0.0170

Table 2—In-sample portfolio outcomes.

Model	Cumulative Return	Avg. Daily Return	Std. Dev. of Daily Returns
Manual Strategy	0.0401	5.161E-5	0.0073
Benchmark	-0.0853	-0.0002	0.0085

Table 3—Out-sample portfolio outcomes.

4 STRATEGY LEARNER

The strategy learner implementation centered around a random forest prediction model where the "forest" was composed of 5 random tree learners, each with a minimum leaf size of 5. Multiple leaf sizes were considered, but since the majority of the test cases for this project hinge around the strategy learner's in-sample performance, the smallest leaf size permitted by the project description (5) proved most advantageous.

In that same line of thought, it would then make sense to use 1 tree in the learner to fit the in-sample data as closely as possible, but I chose 5 as a good-faith compromise as there was not a minimum "bag size" specified.

As for the internal mechanics of the learner, the implementation centers around reading in technical indicator values for a given training period and fitting a trade signal to them (i.e. -1, 0, and 1 coded to short, hold, and long respectively). To fit a trade signal for a given day d, the model looks at $d + 1$'s daily return where a return greater than 0.02 - market impact signals that the trader should take a long position on day d. Likewise, a $d + 1$ daily return less than -0.02 - market impact indicates the trader should take a short position. All other daily return values are treated as signals to hold the current position.

Once the trader has fitted a random forest regression model to the training sample, technical indicators for a testing period are then queries against the model to retrieve an array of values corresponding to positions the trader should take. Since the random forest regression returns a floating point number for each day in the testing window, the values were converted into buy signals if greater than 0 and sell signals if less than 0. Given a series of encoded position recommendations, the algorithm operates similarly to the manual strategy trader such that it iterates day by day through the test period and follows the day's recommendation so long as it obeys the project's holding constraints.

5 EXPERIMENT 1

5.1 Methodology

This experiment compares the in-sample portfolio values of a benchmark, manual strategy, and random forest trader (strategy learner).

The benchmark's trades consisted of buying 1000 shares of JPM on the first

day of the in-sample date range (Jan. 1st 2008) from an initial cash amount of $100,000 and closing its position on the last day (Dec. 31st 2009). The manual strategy and the strategy learner both used the three technical indicators and the overall configuration described above. All portfolios were subject to a market impact value of 0.005 and a per-trade commission fee of $9.95.

Prior to this experiment, my initial hypothesis was that the manual strategy would out-perform the strategy learner and the benchmark since the manual strategy accounts for the price of the asset when trading and, as a result, does not permit trades that result in net losses. However, given that the strategy learner was exposed to data it had seen before, its portfolio's value quickly outpaced the manual strategy:

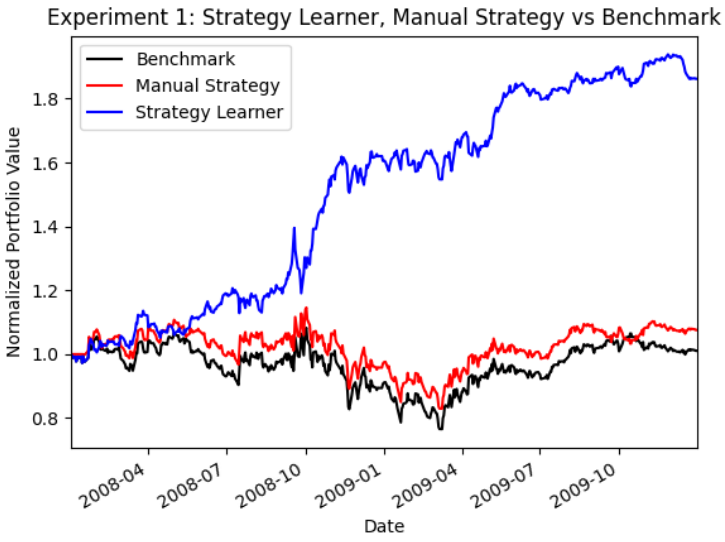

Figure 3—Measured on in-sample trading days, the strategy learner significantly outperformed both the manual strategy and the benchmark. Both models surpassed the cumulative return of the benchmark.

Given that this experiment was conducted on in-sample data, I would confidently expect the same result on most other in-sample data experiments: the manual strategy is price-aware and will beat the benchmark so long as it trades at least twice and the random tree learner should beat the manual strategy since it reading in the training data as-is to inform its recommendations rather than

comparing the indicator values to static buy/sell cutoffs.

6 EXPERIMENT 2

In this experiment, the strategy learner portfolio is assessed at varying market impact levels to determine the impact of impact. All configuration from Experiment 1 remains the same with the exception of commission has been reduced to $0/trade to better isolate the effects of market impact.

Prior to this experiment, I hypothesized that the portfolios working under a larger market impact would be less profitable given that a larger impact means fewer training datapoints would meet the threshold to be considered an advantageous buy/sell point and fewer trades would be conducted.

In regards to cumulative return, the strategy learner performed better at lower levels. As shown below, the model failed to post a positive return after raising the rate to 0.02:

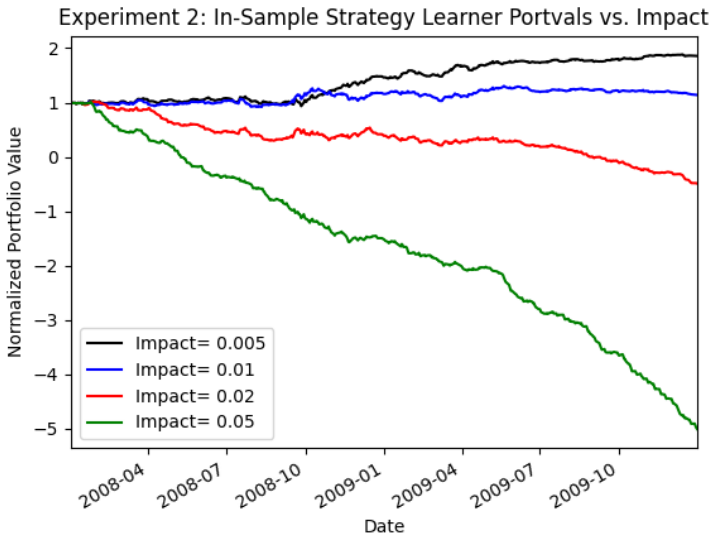

Figure 4—Measured on in-sample trading days, the strategy learner was significantly impacted by market impact values, with cumulative return falling as the trader's market impact increases.

Also in line with my hypothesis, the number of trades decreased with rising market impact levels. Fewer dates were seen as a buy or sell date when training the

model due to higher impact levels raising the signal thresholds and the model's test predictions reflected the increasing rarity of a buy/sell signal relative to the in-sample test technical indicators. Unexpectedly, despite a falling number of executed trades, the ratio of executed trades to that of detected buy/sell training signals rose sharply as impact increased.

Market Impact	Cumulative Return	# of Trade Signals in Training Set	# of Trades Made
0.005	0.7869	376	196
0.01	0.1871	350	192
0.02	-0.9350	310	178
0.05	-5.5611	227	175

Table 4—In-sample portfolio statistics.