Equity Security Trading

A Monthly Investment Strategy Driven by Machine Learning

Brian Cote

April 2019

Table of Contents

I. Introduction

The stock market is one of the most prolific industries in modernity and yet simultaneously one of the least understood. In 2016, all markets across the globe had a combined worth of $69 trillion dollars, a market worth greater than the combined GDP of every nation on the planet combined, all owned privately. These markets are composed of dozens of different forms, from pension funds, to insurance, mutual and index funds, and direct stock ownership. According to a 2017 Gallup Poll[1], although participation in the market is markedly decreasing in recent years, still 54% of Americans report owning stocks, usually in participation of a 401(k) retirement account. These forms of securities, long-standing retirement accounts or 'safe' bonds/mutual funds/etc. which are meant to appreciate over decades is the extent most Americans participate in the market. The lesser appreciated aspect of the market however, especially in consideration of the above numbers, are short-term trading plays. These are trades with timelines not measured in decades or even years, but months, weeks, days, or even hours.

This is for good reason; the stock market is inherently the riskiest form of investment whose risks are usually offset for most by holding onto the equity for a long period of time. For short-term plays however, the risk is amplified considerably. The stock market is a statistician's worst nightmare; equities (stocks) are generally dynamic, non-parametric, chaotic, and noisy in nature. Even worse is that the Efficient Market Hypothesis, which has a whole body of literature supporting it, posits that equity prices in the present perfectly reflect all available information such that each day of market trading is a random walk. However, recent literature by Jagadeesh & Titman[2], and Jacobsen & Zhang[3] indicates faults exist in the EMH. The former argues that short term behavior of equity prices may exhibit exploitable momentum, while the latter argues that centuries of market data indicate equities follow significant seasonal trends.

These concepts have led to the development of the Semi-Strong Efficient Market Hypothesis, stating that all past information is indeed accounted for in equities, but unexpected anomalies and insider information provide advantages to some. In keeping with this, explored in this paper is a study absent of the standard features relating to the underlying company, economy, and sentiment, but modeling based on a function of movement and volatility of a stocks response to anomalies.

II. Data Description

The data collection is relatively straightforward. As one might expect with a $69 trillion industry, the bookkeeping is extraordinary. Yahoo Finance is actually one of the most reliable sources for market data, and the site contains market data on a per-day basis for any stock in any market. I used the Python package DataReader which trivializes the import of market data. By stating the start and end date and inputting a stock ticker (ex: Apple is 'AAPL'), a web-scraping algorithm pulls data from the site for the relevant data. In this case, on a per-day basis from 01-01-2011 to 12-31-2018, I pulled the daily High, Low, Opening, and Closing Price for my chosen stocks along with the total shares traded, called 'Volume'. The descriptive statistics for this data are likewise surprisingly irrelevant for discussion. For instance, what use does "average opening price" present if a stock is consistently rising in price? As stocks essentially behave as random-walk markov chains day-to-day, these statistics in their own right serve no meaningful purpose in my view and any meaning that might be found in them would be spurious.

---

[1] https://news.gallup.com/poll/211052/stock-ownership-down-among-older-higher-income.aspx

[2] Jagdeesh, N. & Titman, S., "Momentum", Annual Review of Financial Economics, Vol. 3, (December 2011)

[3] Jacobsen, B. & Zhang, C., "The Halloween Indicator, 'Sell in May and Go Away': Everywhere and All the Time", American Economic Review, Vol. 92, No. 5, (December 2002)

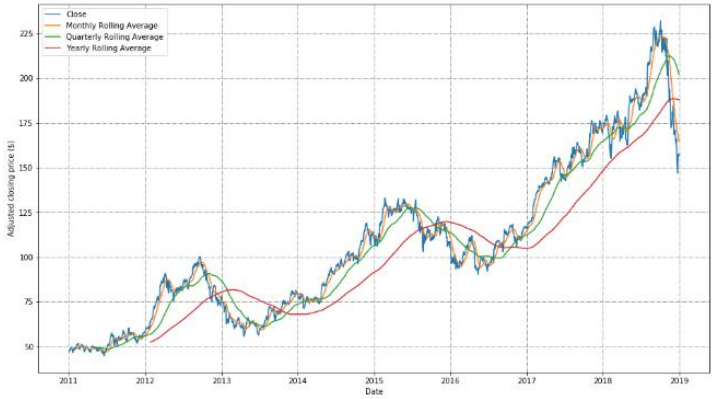
Whereas most prior work on this study is based on fitting multivariate time series models to predict price directly, we will be treating it as a *classification* problem based on stock direction. It is known that forecasting, especially with chaotic, noisy data, may generate impossible outlier points. For instance, it is possible to forecast a commodity price equaling to $0. From an economic point of view, this is essentially impossible. On the contrary though, a classification model provides a probabilistic view of trend and thus is overall safer and more trustworthy in its results. This creative solution was not my own, though. I build this report on top of the success of Saahil Madge[4] and Suryoday Basak et. Al[5], two studies which investigated the optimal manner to build machine learning models around equities, and both viewed it in this classification perspective.

Both also came to two more important conclusions:

1. Information relating to a stock itself is insufficient. We need features relating to the stock itself alongside information relating to the sector a stock belongs to at large. We use what is called an Exchange Traded Fund (ETF), which is a stock that is a collection of various sector stocks.

2. Based on testing models for every possible permutation of trading windows, the short-term information (5 trading days, 1 week) of the stock along with the long-term information of the corresponding ETF (90 trading days, 1 fiscal quarter) lead to ideal model performance.

I will be analyzing a test-case as a proof of concept and then move onto more interesting stocks. These will all be technology-sector stocks. My test case will be Apple Inc. ($AAPL) whose behavior along with rolling moving averages are visualized below:



Just noting the behavior of the plot can make my reasoning for this as a test case clear. $AAPL, exhibits consistent cyclical yet upward behavior. This is what I might consider a 'typical' stock behavior in this sector. My proof of concept model wishes to confirm the hypothesis that a machine learning model can inform consistent gains on a stock. The ETF we will be using for this stock is the S&P 500 Index ($SPY), a technology sector ETF of which $AAPL is a member.

My strategy is to pull in the history of both at once – $AAPL and $SPY – and compose the relevant features for each. Then, create one singular dataset with the two combined. The output will be generated as looking at the $AAPL closing price for a given day and comparing it to the price 20 days ahead, assigning a +1 if it is higher, and -1 otherwise. This means the same features of momentum and volatility are calculated for both the stock and ETF along different time frames, and both used.

---

[4] Madge, Saahil, "Predicting Stock Price Direction using Support-Vector Machines", PhD Thesis, (Spring 2015)

[5] Basak, Suryoday et. Al, "Predicting the direction of stock market prices using tree-based classifiers", The North American Journal of Economics and Finance, Vol. 47, (January 2019)

III. Feature Creation

The features utilized are defined in the following section. These are the most popular indicators for momentum and volatility in this area of research, and recall we will define these for both our stock and ETF among their different time frames. We define *n* in all cases as our 'lookback window.' In the case of a stock it will be a 5 trading-day window, and an ETF a 90 trading-day window.

The first two features are the Stochastic %K and %D Oscillator's, credited to Lane 1984[6]. These measure the day's closing price compared to the expected high and lows in a window of time.

$$\%K = 100 * \frac{C - L_n}{H_n - L_n}$$

$$\%D = Simple\ Moving\ Average(\%K)\ over\ n\ days$$

$$C = current\ closing\ price$$
$$L_n = lowest\ price\ in\ past\ n\ days$$
$$H_n = highest\ price\ in\ past\ n\ days$$

Williams %R is an alternative to the %K. Here we compare the closing price to the highest price in the "look-back period" rather than the lowest. Some scholars see this, even if correlated, useful to include.

$$\%R = -100 * \frac{H_n - C}{H_n - L_n}$$

Average True Range (ATR) is one of the most powerful volatility indicators in use today. It was invented by Wilder Jr. 1978[7] and works by decomposing the price range of a stock in a chosen period defined as,

$$TR = \max\{(high - low), abs(high - Close_{prev}), abs(low - Close_{prev})\}$$

$$ATR_t = \frac{(n-1)ATR_{t-1} + TR_t}{n} \quad , \quad ATR_1 = \frac{1}{n}\sum TR_i$$

Moving Average Convergence-Divergence, credited to Appel 2005[8], is an exponential moving average of prices based on two date points. Usually this is used when compared to another moving average baseline, called a signal line, but here we will use the raw moving average and let the model decide if it is a significant signal at a given point or not.

$$MACD = EMA_{26}(C) - EMA_{12}(C)$$

$$EMA_n = n - day\ exponential\ moving\ average$$

Relative Strength Index (RSI), also from Wilder Jr. 1978, is a popular momentum indicator that investigates over/underselling of a stock. That is, if demand is unjustifiably raising/lowering the price of a stock and a crash/spike is likely to occur. RSI ranges from [0,100] and generally RSI over 70 indicates a stock is overbought while under 30 indicates underbought,

$$RSI = 100 - \frac{100}{1 + RS}$$

$$RS = \frac{Average\ Daily\ Gain\ Over\ past\ n\ days}{Average\ Daily\ Loss\ Over\ past\ n\ days}$$

---

[6] Lane, Jan-Erik, "Public Finance Variations: A New Approach", Scandinavian Political Studies, Vol. 7 (1984)

[7] Wilder Jr., J. Welles, "New Concepts in Technical Trading Systems", Trend Research, (June 1978)

[8] Appel, Gerald, "Technical Analysis Power Tools for Active Investors", Financial Times Prentice Hall (2005)

Price Rate of Change (PROC) is a popular technical indicator which finds a percentage change in price of today compared to a window of time. I also formulated an additive version myself, which I thought might be useful. These are used as volatility indicators, formulated as,

$$PROC_t = \frac{C_t - C_{t-n}}{C_{t-n}}$$

$$Momentum_t = C_t - C_{t-n}$$

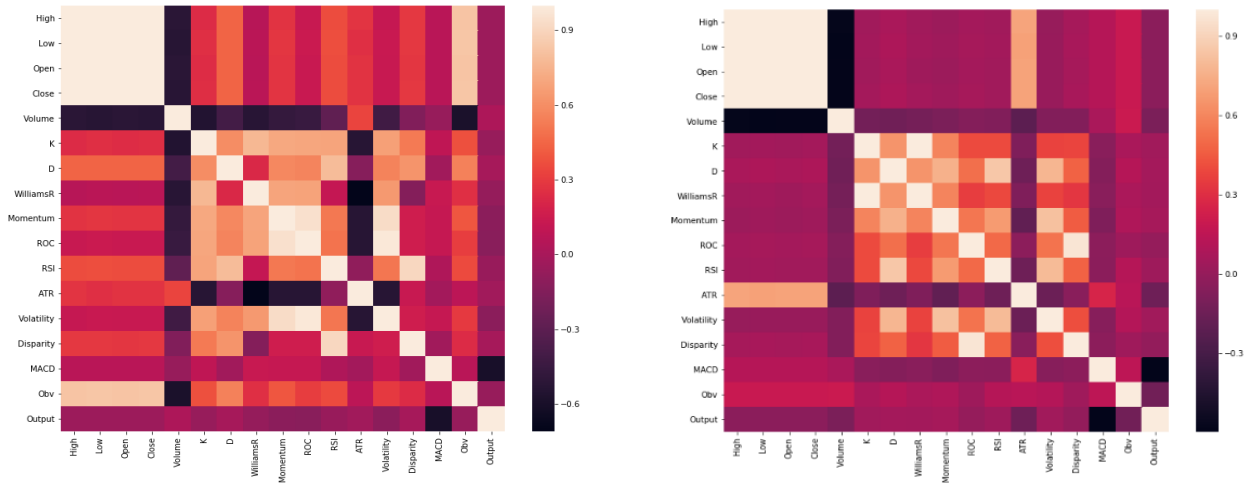
On-Balance Volume (OBV) is a slightly complex formulation invented by Granville 1976[9]. It, simply, considers cumulative volume of a stock. It will add total volume on days where stock price increases, subtract volume on negative days, and stay the same if the price remains unchanged. The Disparity Index investigates the relative position of todays stock closing price divided by a simple moving average over a window. Finally, a popular volatility indicator which measures the average daily change over our window,

$$Volatility = 100 * \frac{\sum \frac{C_t - C_{t-1}}{C_{t-1}}}{n}$$

IV. Feature Selection

With so many features relating to the performance of momentum/volatility, we should expect pretty high multicollinearity. Below is a correlation analysis to prune egregious cases of correlation from the $SPY and $AAPL feature list respectively. I concluded to remove the naïve statistics (High/Low/Open/Close) due to a near perfect correlation, $SPY's Disparity and ROC variables, and $AAPL's William's %R variable. Even with these removed, there are still a considerable number of variables with low to moderate levels of intercorrelation.

It might seem redundant to include these, but it is important because they each measure momentum/volatility in slightly different aspects. I am considering a tradeoff. There is a high amount of noise to account for in the daily movement of a stock. I believe taking on the risk of including slightly-correlated variables we might otherwise remove in a standard modeling exercise will be outweighed by the increased performance in reducing noise and catching momentum and volatility signals. Further, machine learning techniques can be applied to offset correlated variables, which we investigate in the modeling section on the following pages.



[9] Granville, Joseph E., "Granville's New Strategy of Daily Stock Market Timing for Maximum Profit", Prentice-Hall, Inc. (1976)

V. Model Selection & Tuning

With the background established, our modeling goal can then be stated in plain English: We will build a classification model to predict a stocks price direction 20 trading-days (1 month) in the future using features relating to the short-term momentum/volatility of the stock itself along with the long-term momentum/volatility of the sector it belongs to as a whole. I propose two methods of prediction, the Support Vector Machine Classifier (SVC) and Random Forest Ensemble Method (RF).

The SVC has a distinct advantage for this area of study, only the few points along the support vector margin fit the model. Therefore, additional points on a day-to-day basis and random noise will not affect the separating hyperplane when fit. This will massively offset the noise issue. I think the SVC actually would perform the best the shorter the trading window gets for this reason, as the closer we predict, the more noisy a stock prediction gets. It likewise has considerably faster fitting time compared to an ensemble method, even when using an RBF kernel in the tuned case, which would be advantageous in smaller trading windows (i.e.: day-trading). I compare the performance below table of the Linear SVC using an arbitrary *C*-constant and a tuned SVC using an RBF kernel, using a 5-fold cross-validation grid-search on the 2011-2016 training set to optimize both *C* and σ, and then tested on the 2016-2018 set.

Comparison Metric	Linear SVC	Tuned SVC
Total Predictive Accuracy	74.80%	79.01%
Sensitivity	84.92%	82.14%
Specificity	69.50%	77.39%

This is a promising predictive quality. We easily outperform the 50/50 coinflip of the hypothesized random walk, thus we can usually predict direction on a monthly level. What is concerning is that even with the tuning, the SVC tends to favor Sensitivity. The SVC favors making good-plays and predicting upward movement but lacks in catching bad-plays, that is, missing down months. Since this method of classification only gives direction but not magnitude of directional change, we want to avoid bad-plays as much as possible due to not knowing how severe a mistake could cost. It is better to sell and miss some profits rather than make every single good-play possible while also making every bad-play.

Since we are not day-trading and do not care all too much about fit times since this is a *monthly* strategy, Random Forests are also considered for an important reason: decorrelation. Since RF's inherently decorrelate by selecting a random subset of predictors to fit per tree, the mild-moderate correlation still remaining above is heavily mitigated while still keeping the noise-reduction of having so many different measures of momentum and volatility. The Tuned RF is based on a 5-fold cross validation grid search.

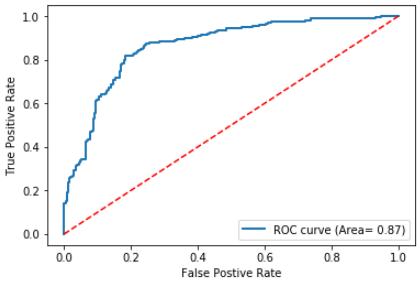
Validation Set Score	Arbitrary RF	Tuned RF
Validation Set 0 score	64.14%	72.51%
Validation Set 1 score	72.11%	85.66%
Validation Set 2 score	78.09%	76.89%
Validation Set 3 score	64.80%	82.00%
Validation Set 4 score	62.25%	68.67%

My optimum was 311 trees, $\sqrt{p}$ features per tree, a max depth of 2, minimum samples per leaf of 10, and minimum samples per split of 15 with a 3 minute 42 second runtime. The performance of the trees on the same 5-fold cross validation sets is compared above, with the tuned RF massively outperforming the

arbitrary RF on the validation data. With this in mind I went to test its efficacy on the testing dataset from 01-01-2016 to 12-31-2018, with the results summarized below:

Comparison Metric	Arbitrary RF	Tuned RF
Total Predictive Accuracy	79.02%	82.29%
Sensitivity	80.95%	77.78%
Specificity	76.14%	84.65%

We likewise can see a ROC curve of our tuned RF, which optimized our discrimination thresh-hold.



I briefly note that the reason I did not test GBM's is from the aforementioned paper of Suryoday Basak. He explicitly compared performance of GBM's to RF's, and while in longer term predictions they performed similar, in shorter trading windows (5-30 days), RF outperformed its GBM ensemble counterpart. This is maybe due to GBM's being more susceptible to overfitting noisy data than RF.

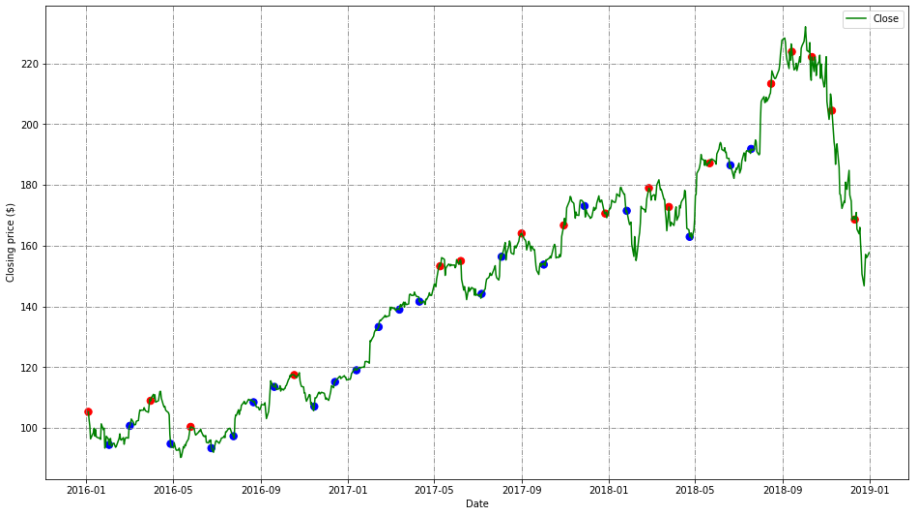
With all of this presented, I believe the RF is the optimal model to use. It naturally optimizes specificity, meaning it prioritizes avoiding down-trending months. Again, I believe this long-run time is only valid because of the fact this is a monthly trading strategy and we are only modeling based on a per-day basis. Day-trading, where we are looking at minute-by-minute prices and volumes with success measured in making trades seconds before competition, would likely find an SVC more appropriate. Here though we have the luxury of waiting for a better-tuned ensemble method with ideal parametrization.

VI. Model Results: The Ideal Stock

The next question though is what does this look like? That is, these tables are useful for assessing overall theoretical accuracy, but they do not inform an actual trading as if I were actually using it to attempt making money. For this reason, my final analysis of model performance is not based on a statistical comparison, but on a simulated trading session if I were a real person using this model.

The first component of the trading simulation is visualized on the following page, where monthly buy/sell signals from the model are overlaid on the price graph. Blue represents a buy signal and red a sell signal, with direction being predicted once per month. This is a more realistic and practical view of model performance. It has advantages over raw numbers too. For instance, we can see the model has slight issues handling "sideways" months (ex: 11-2017 to 02-2018). However, it is exceptional in detecting inversion points – where a growth period turns red, and vice versa. I find this plot promising as, if I were using this model, whenever I see a buying opportunity, I can be confident at worse it will be a near neutral change.

To continue the exercise, I want to see what would occur in dollar amounts. I ran a naïve trading strategy as follows: Starting at zero shares held, if I see a buy signal, I buy a single share. If I see a sell-

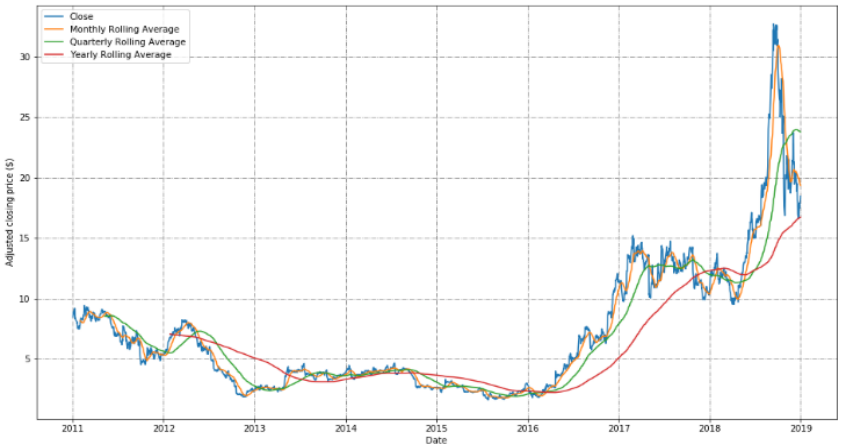


signal, I liquidate my entire position. The plan is to ride upward trending waves then dump at the end of in wait for the next, starting fresh every time. Between 01-01-2016 to 12-31-2018, the simulated random forest model earned a net profit of **$367**. This is great! It indicates that machine learning can secure a profit. However, the *real* question is not if the machine learning focused strategy makes money. The question to ask is does it beat a baseline buy-and-hold strategy?

This is how I ultimately will evaluate the model: Is modeling the data better than just buying and doing nothing. The buy-and-hold, commonly called dollar-cost-averaging, is simple: At a consistent interval, buy shares for an asset no matter what the trend. Over long periods of time, spikes and dips average out if there is continued overall growth. If we were to use buy-and-hold in the same $AAPL case, buying a share every month no matter what, we earn a total profit of **$2247**.

VII. Model Performance: Non-Ideal Performing Stocks

This might seem like a critical blow against machine learning. Why bother doing all of this complex trading strategy if just buying doing nothing will outperform? Consider that we can not *guarantee* an asset is going to appreciate by a considerable amount like $AAPL did between 2016 and 2018. In reality, we don't know how a stock will behave in the future! $AAPL is a dream case of being one of the wealthiest corporations in the world with some of the most meteoric growth in recent years.



How about stocks that did not perform as well? Let's turn our attention to another technology sector stock in the S&P 500 Index, Advanced Micron Devices ($AMD) for which I also fit a tuned Random Forest in the same way as above. To the left is its price graph over the training and testing period. As before, output below is the test trading strategy between 2016 and 2018 for $AMD:

This is a far more interesting case than $AAPL. During its training period, it is mostly sideways, whereas in the testing period, it has exponential growth in price. This situation informs another important question: Can machine learning, trained on stable data, then perform well on highly volatile data? The answer is yes. Even better, the profit differential between the random forest model and buy-and-hold strategies are closer. The random forest model earned a profit of **$127** whereas buy-and-hold Strategy **$349**.

This gave me to one final hypothesis worth investigating: I think more sideways or negative trending a stock performs, the more useful the monthly random forest model becomes. Let us consider one more case, AT&T Inc. ($T), a telecommunications corporation part of the S&P 500 Index, visualized below:

Note in its testing period, unlike the prior two cases, $T experiences a net loss. There are considerable spikes in profit though as it continues its inevitable spiral down, which serves as a profit opportunity buy-and-hold would have missed out on. Consider the trading strategy outlined below:

In this case, the random forest model would have generated a total profit of **$28**. If this seems like a meager amount, consider that the buy-and-hold strategy would have experienced a net loss of **-$243**. This provides strong evidence for my hypothesis. A tuned random forest model will outperform buy-and-hold if the underlying stock has sideways or negative behavior.

VIII. Review

Let us briefly recall everything done up until this point. Based on literature we found that features relating to a stock's underlying company and the economy at large can not be used due to the Efficient Market Hypothesis. Literature suggested features relating to the short-term momentum and volatility of a stock in combination with the long-term momentum and volatility of the sector at large a stock belongs to can create an accurate machine learning model informing the stocks direction. We generated a list of features to capture this information for both a 5-day trading window of a chosen stock along with the 90-day trading window of its corresponding ETF, which acts a market-sentiment measure. These are combined into one dataset with a binary output of +1/-1, if the price in 20 days is higher or lower than that day.

With this information, and some of the features pruned, we built two classification models based on the $AAPL proof of concept. We found while an SVC performs quicker, the tuned Random Forest provides more accurate overall predictions and better optimized specificity. While the model accurately predicted movement, and also informed a net-positive trading strategy, it performed worse than a simple buy-and-hold strategy. We likewise looked at more interesting cases of lesser performing stocks, $AMD and $T, finding that the less upward price growth a stock experiences, the better the random forest model performs compared to buy-and-hold. In fact, during a net loss over a period of time, a random forest can provide a net gain.

IX. Conclusion

If one can guarantee a stock will consistently experience growth over a period of time, a simple buy-and-hold strategy will always outperform any sophisticated trading strategy. As an entity, the market as a whole tends upward even despite recessions. Though on a company-by-company basis we cannot be so certain. Companies file bankruptcy or decline in sales and the market in general may be irrational. One might also need to liquidate their position soon, such as a near retiree, and would not have the luxury of waiting for a post-recession rebound. When any of these occur, buy-and-hold fails. I view machine learning informed strategies then as a short-term hedging method. If a trader wishes to make their portfolio recession proof for the next few years, or just wants their assets resistant to devaluation, a monthly machine learning model can guarantee at least meager profits at the expense of lowered potential gains.