Stock price forecasting using machine learning with event detection on candlesticks (NASDAQ and NYSE) $

Research · November 2020

DOI: 10.13140/RG.2.2.32116.17285

Salman Marvasti
Sharif University of Technology

Sepehr Abdous
Johns Hopkins University

Stock price forecasting using machine learning with event detection on candlesticks (NASDAQ and NYSE) ☆

Salman Marvasti[a,1,*], Sepehr Abdous[b], Negin Majidi[b]

[a]*Rizpardakht - Davum Capital, Ordnance House 31 Pier Road St Helier Jersey JE4 8PW*
[b]*Sharif University of Technology, Azadi Ave, Tehran, Iran PO Box: 11365-11155*

Abstract

In this paper we present a novel approach that uses daily candlestick prices, event detection along with machine learning to create a computational technical analyst. We establish a statistical significance to our approach. Our trend forecasting system through the combined use of a unique event and trend detection algorithm for unsupervised learning prior to the application of Artificial Neural Network (ANN). Other financial forecasting papers have typically used either classical economic models such as Capital Asset Pricing Models, or time series models such as Auto Regressive Moving Average models, and AR Conditional Heteroskedasticity or neural networks directly on pricing data with little cross sector applicability. Most such attempts have either used just price or other derived metrics.

Our learning system differs by the use of an event detector which is trained on past data using more than just close prices on a large group of individual stocks. We apply our model on individual American stocks(441 stocks) which is a larger space than most public research studies involving ANNs. Indexes are much more predictable because averages tend to remove noise, but we are interested in single names. In addition to candlestick prices we add a derived metric (RSI) and show that little value. We show that it is possible to predict correctly trends from candlesticks upto 70% of the time across all individual stocks, suggesting a correlation exists between inflection candles and future stock movements; this system has potential for profitable exploitation in markets and is evidence for structural inefficiency in single names.

Keywords: neural networks, machine learning, event detection, finance, technical analysis, unsupervised learning

---

☆paper title

*I am corresponding author

Email addresses: `smarvasti@rizpardakht.com` (Salman Marvasti), `sepehrabdous1375@gmail.com` (Sepehr Abdous), `negin.majidi45@gmail.com` (Negin Majidi)

URL: `www.github.com/SalmanMarvasti/Research` (Salman Marvasti)

[1]Dubai

1. Introduction

Forecasting and predicting financial market trends is controversial if not impossible due to large amounts of noise present in market data and an unclear dependency between future and past prices. [12]. Artificial Neural Networks (ANN) have been used with and without combinations of heuristic features from pricing information [20]. Until recently even ANN, were used either for predicting prices or on some derivative measures such as returns and correlation analysis. We are using ANN for classification of potential inflection points, rather than directly predicting price values. We intend to replicate the human technical stock chart expert.

Predicting prices is not usually accurate because stock prices typically are affected by a mixture of random factors. Older studies such as [19] used feed forward NN's to study the IBM daily returns. However this had a problem with over-fitting. After White there were many attempts, either using pricing data direct to predict future prices such as [16], or using technical and other manual identification of second order features to train the networks. The more recent studies have used ANN with genetic algorithms with echo state networks [16] [14]. There are two viewpoints in literature: either that there is no signal or correlation to be found thus markets are perfect random walks or there is a weak signal mixed with randomness that neural networks may be able to pick out. Price trend models which derive from auto-regressive models have shown that prices follow mean reversion [11], which is a known phenomena used by professional traders [9]. Yet there is no publicly available system for reliable forecasting to use for portfolio optimization. Others such as [13], [8] and [12] have used AI techniques including neural networks but have grappled with the problem of choosing the optimal network structure.

We intend to test the hypothesis that there is some information contained in past prices about future behaviour through a hybrid system trained using deep learning using the same input as a basic technical analyst: candlesticks. There is an underlying predictable movement despite effecient market theory probably because markets involve humans psychology [2][1]. When things get cheap some people jump in, and when they get expensive relative to what they paid, they pull out; this is known as profit taking. So there should be elements of predictability that a well designed ANN should be able to pick out [13]. We use the basic elements of technical analysis(TA) [8] and setup multi-layer neural networks to learn from candlesticks.

1.1. Replicating a human expert and modeling price movements

Technical analysts (a.k.a chartists) assume that immediate past values of the time series contain information required to predict its future behavior, but they rely on rules that are closer to art than to reproducible science. Chartists should not be able to predict major movements with a profitable (greater than 50%) probability as pricing theory is usually based on mathematical martingales; in other words historical price movements should not be related to future price movements[5]. This is also a weak form of the efficient market theory [3]. An extension of this theory is the Random Walk(RW) model. The RW model assumes that not only all historic information is summarized in the current value, but also that increments – positive or negative – are uncorrelated(random), and balanced, that is, with an expected value equal to zero. In other words, in the long run there are as many positive as negative fluctuations. The random nature of this model makes accurate predictions difficult. The second hypothesis interprets sudden movements as the result of delayed or incomplete information that impacts prices. External phenomena such as political turmoil or climatic changes influencing agricultural yield for instance, may affect subsequent stockmarket prices. Accurate predictions are difficult because it is not possible to model, quantify or even know apriori such external phenomena. However the way external events impact prices, we hypothesis, depends on past pricing behavior. After all, if an event makes you fearful and you are in profit you are more likely to sell than if you are in deep loss. Thus human psychology, makes the martingale presumption not necessarily accurate. /par In this respect, the basic tools of a chartist involve candlesticks and certain aggregate indicators, such as the RSI. However, no public study exists linking candles using neural networks to predict major inflection points. Here the ANN would emulate chartists by observing candles over a long enough time frame. A candle, in trading and pricing terms is a shape that is obtained by observing the open, close, high and low of price during a defined period. The candles were labeled by observation of future movement in price, up, down or steady. Each of these are defined mathematically in the Event section.

2. Data and Methods

We used 10 years of stock market data form Apr 01, 2007 to Apr 01, 2017 with 441 stocks of which 150 symbols are listed below. This decade of data is sufficient to get a statistical significant result ; after event detection pruning we had 15461 inflection points(see figure 1). For each day of any stock there are four data points representing the stock prices in the highest and lowest point and also at the time of opening and closing (Open, Close, High, Low in short). Data was obtained through Yahoo finance via a curl command. 80 percent of our data was used for the training set (12369 rows of 12 dimensions or 3 days of candles), and 20 percent (3092 rows) is held over for testing analyzed through Python NumPy with tensor flow and Keras libraries.

ADSK	WFC	NSC	WY	BLL	AFL	HES	COL	UST	ED
TER	DGX	FTI	FAST	MYL	SCHW	SBUX	JCP	AKAM	WFT
VAR	HPQ	PFE	XRAY	TEX	FMCC	FITB	SHW	WHR	HUM
HON	CSCO	MBI	BC	PNC	HOG	AMGN	STI	RRC	LYB
EXPE	CTXS	FNMA	BBBY	WMT	AMT	HRB	LH	LLY	EFX
AMP	GOOG	MAS	GD	CLX	ADBE	TYC	NOV	BWA	SII
SJM	UIS	S	ROK	RTN	JWN	HBAN	KR	WYE	PCP
ORLY	MTG	VLO	MUR	WAMUQ	CF	POM	ITW	TGT	IRM
FII	L	ANF	MET	APH	XRX	CAT	C	FTR	LMT
EXC	EBAY	LIFE	AMD	ROST	EQT	GCI	PX	CAH	EQR
KBH	DVN	XTO	CTAS	DHR	RSG	EXPD	CTSH	CMS	WFR
CI	NRG	VFC	PXD	NYX	APOL	ZMH	PBG	SLE	FLS
VZ	LM	EQ	PTV	HAL	CVX	LUV	LRCX	DISCA	AET
CBE	DE	RIG	OMC	MCO	FFIV	MCHP	MI	CB	EK
AGN	XLNX	KO	JNJ	MCD	RRD	GPC	CMI	BDK	ASH

Table 1: The symbols of 150 of the total 441 stocks used. The symbols of all 441 stocks are listed in `https://github.com/SalmanMarvasti/ResearchCode`

A chartist typically zoomes in on an area of interest in the stock chart. He then uses TA patterns to decide if it is a good entry point or not. The NN at this post event stage is trained on data only and has no prior knowledge of technical analysis rules. We use 3 days of candles at these locations to feed to the NN to predict longer term trend prediction. We use 3 days as it seemed to be a more reliable predictor based on the work by Tharavanji[18]. We labeled algorithmic-ally all the 441 stocks for the 10 years in question and feed that to the ANN. The ANN only received sets of 3 day candles and labels for the training set.

3. Event Detection And Labeling Trends

Our approach involves automated event detection[14], for labelling and training stage. Subsequently we use deep learning neural networks. The inspiration for event detection came from the way technical analysts work by only looking at data which show significant movement. The objective is to replicate part of what a technical analyst does using back-propagating neural networks. We tailored the event detector to achieve (sensitivity and specificity over 95 percent relative to a technical expert) . The first step for detecting the events is to divide the data into windows, using a significant rise or fall in the stock price on the returnized data as defined using the QSTK library provided by Georgia Tech [15]. As a result the first step is to detect the windows based on the special events in the data.

- Identifies trend windows of significance using first order derivatives as described below:

- Thresholds where price moves significantly is based on an entropy measure of the stock price calibrated over the entire price history relative to the apple stock over 10 years. We use Apple's standard deviation as reference and adjust for any expected differences in other stocks. Here variation is calculated on the zero mean returnized data (calculated return based on daily roll-over see next item).

$$T = T_m \times |1 - \log_{10} \frac{apple\_deviation}{stock\_deviation}| \times AVG_{abs}$$

  Where T_m was trained to be 4 and AVG_{abs} is the average of the absolute amount of the returnized prices are calculated as below:

$$r_i = \frac{P_i}{P_{i-1}} - 1$$

  Where P_i is the prices of the day i.
- no window can be less than 3 days
- only daily movement of price is used as it was shown to be the best time scale for prediction [13]

In order that the event detector does not miss movement following the end of a window, a certain amount of overlap is considered between consecutive windows. The overlap was obtained through testing with our data set:

- Our current window length is greater than $4 + \alpha$, where α is the number of days that the windows overlap.
- The difference between the stock price at the current point and the previous point is greater than a threshold(T) then an event is started/ended. The deviation is calculated from the days' low point and the fixed Threshold(T_m).
- The difference between x[n+1] and x[n-1] should also be greater than the threshold T to prevent temporary jumps or impulses from confusing the event detector (x is the array of stock data).
- $|AVG1 - AVG2| \geq 0.5 \times T$
  where AVG1 is the average of stock prices at the current point and the previous one and AVG2 is the average of stock prices of the next two points.

The data was thus divided into windows which have a constant trend, usually either up down or steady. For this purpose for each line we calculate a variable β as following:

$$\beta = \frac{AVG3_2 - AVG3_1}{AVG3_1}$$

where $AVG3_2$ is the average of the last 3 points of window mapped to the line and $AVG3_1$ is the average of first 3 points of window mapped to the line. Regarding the value of β we consider the future trend of the last 3 points of each line. For determining the trend we consider a variable 'Tolerance' which is calculated as below :

$$Tolerance = max(0.04, 0.9 \times stock\_prices\_deviation)$$

The reason to put 0.04 for the variable if the deviation was so small is that for Tolerance less than 0.04 the trend determination would be somehow unrealistic. The general trend for each window is specified as following:

$$overall\_trend = \begin{cases} Down & \beta \leq (-1) \times Tolerance \\ Up & \beta \geq Tolerance \\ Steady & Otherwise \end{cases}$$

We extract the candles before each window and map it to the state of the trend(up, down, steady) calling this matrix A. Subsequently the data is normalized for further processing (kmeans, tensor flow). The normalizing algorithm we utilized is a dynamic normalization algorithm which for each data in A it finds the lowest and highest prices in the last 14 days:

$$C_i = \frac{A_i - l_i}{h_i - l_i}$$

where the values are as following:

- A_i: the i'th element of array A
- l_i: the lowest value in the last 13 days before the day mapped to the i'th element of A and the day itself
- h_i: the highest value in the last 13 days before the day mapped to the i'th element of A and the day itself

After the normalization stated above, C would be the normalized version of A. In summary , first the events on each stock is detected and windows are extracted, then the trend(up, down, steady) is determined for each window, resulting in data with 13 dimensions including the normalized prices(open, high, low, close) of the last 3 days of each window in addition to the RSI (14 days see Appendix) is outputted to a text file. RSI is included optionally to determine whether derived indicators are of any use for the ANN.

3.1. Using KMeans and PCA for visualization

Below you can find attempts at simple linear separation techniques involving KMeans and Principle Components Analysis projection to separate strong up trends from downtrends. In the figure below you can see the candles before a significant up movement and those before a down movement are somewhat separable but not always for a long period of Apple stock. As the data for stock prices have got 4 dimensions: low, high, open and close and we needed data with 2 dimensions to visualize the Kmeans, we utilized PCA to reduce data to 2 dimensions and then apply Kmeans to them.

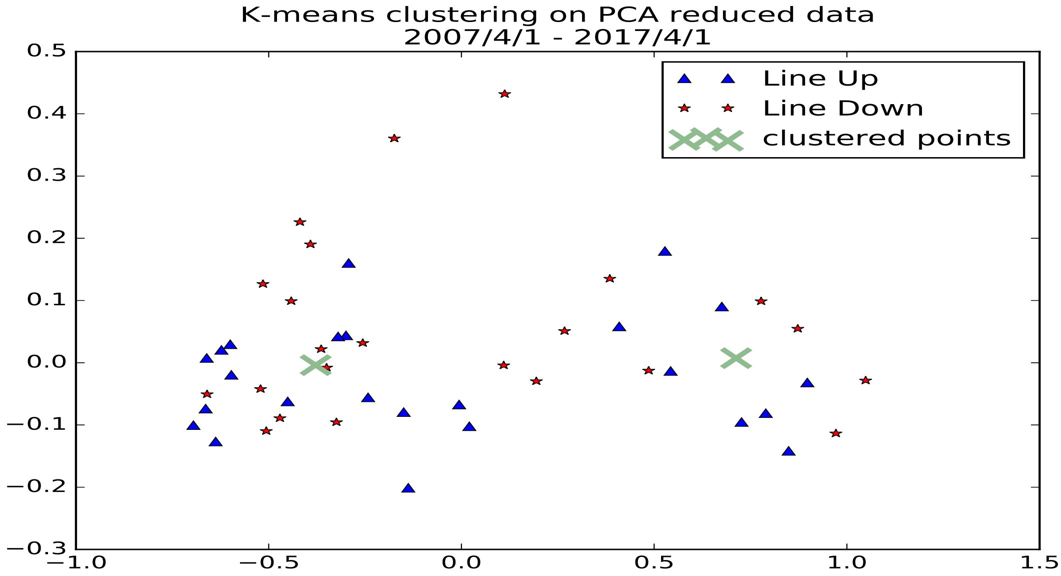


Figure 1: Candlesticks at inflection points for Apple (just before significant stock movement) after projection to 2 dimensions using PCA

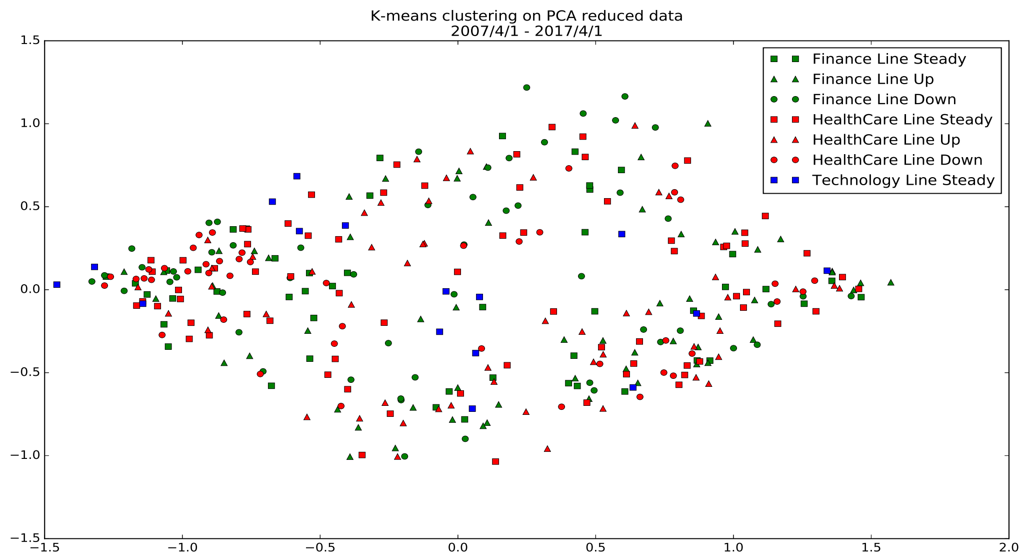


Figure 2: There was no simple linearly separable group in all the sectors in our data set of inflection points. Again after projection on two dimensions with max variance

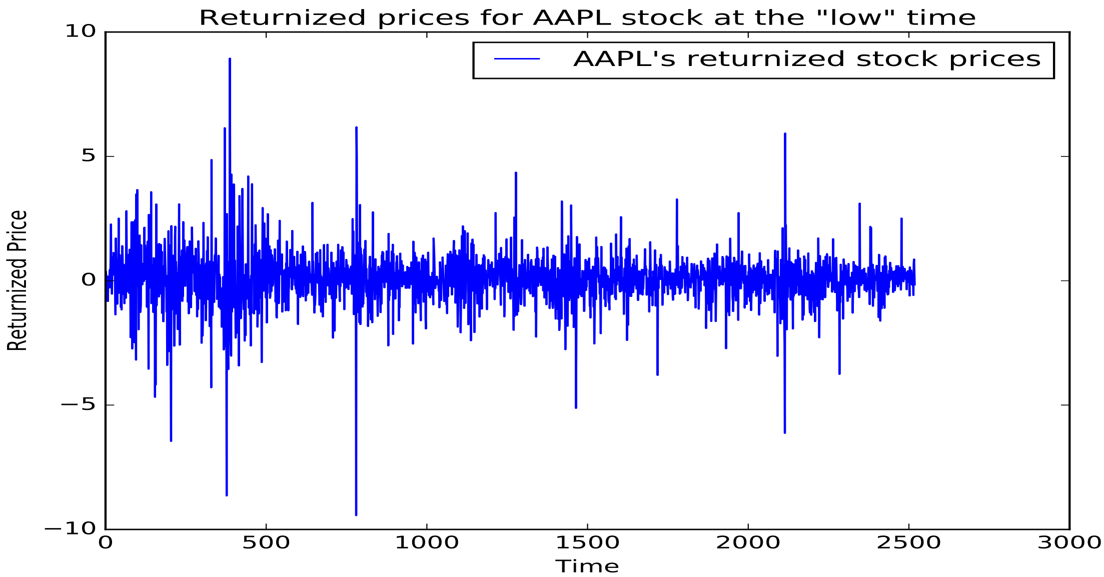


Figure 3: event detect- returnized price used to determine windows length

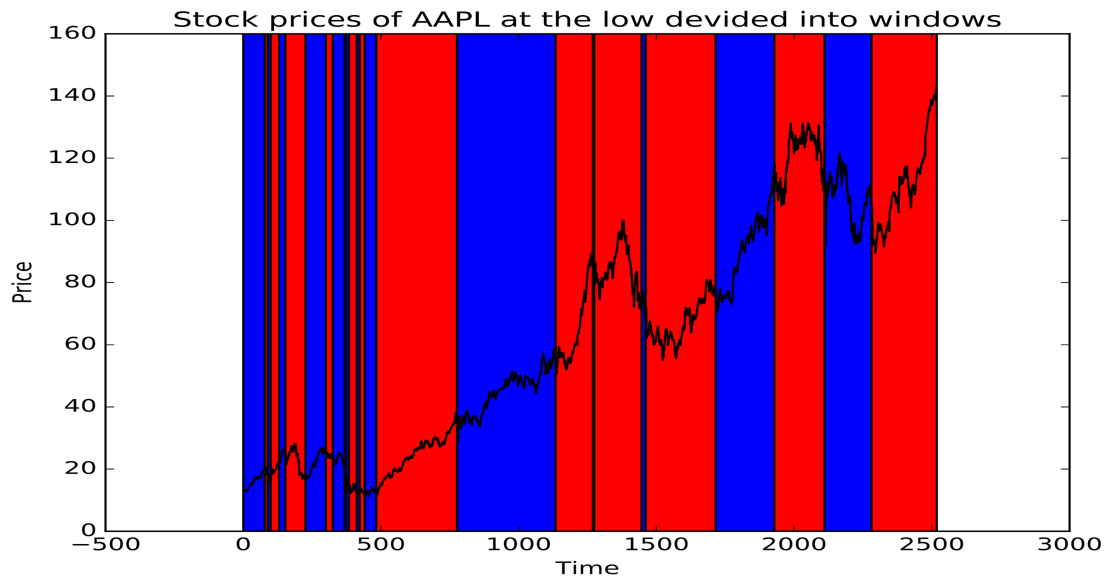


Figure 4: event detect- returnized price used to determine windows length

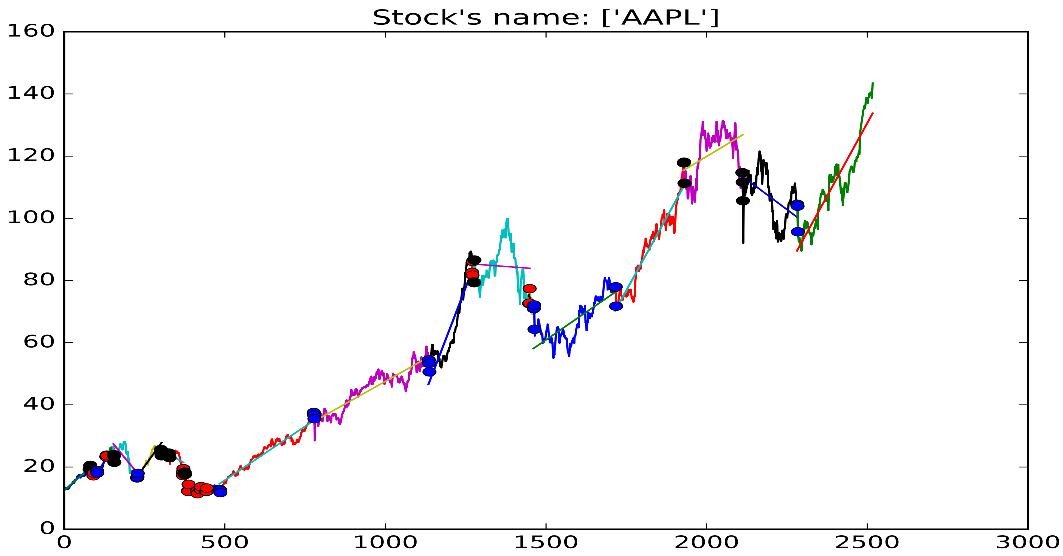
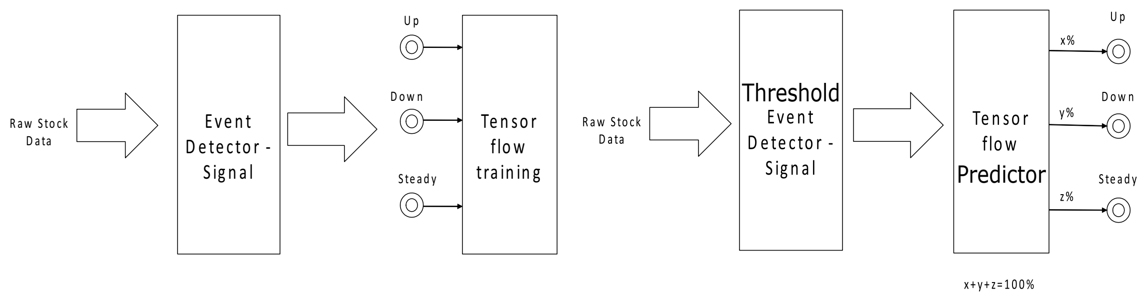


Figure 5: Event detection showing trend lines and inflection points that are passed onto ANN for APPLE (AAPL symbol) stock

. Blue points - uptrend, Black - down trend, Red - no trend.



(a) Train (b) Test

Figure 6: An alternative way TensorFlow machine learning could be used. In our initial system for part b, all candlesticks were fed to the ANN. The Event Detector was just used for unsupervised labelling.

From the above figures, it is clear that a somewhat non linear separation technique is needed if we are to get any partially useful system. This is where deep learning on multi-layer networks help.

3.2. Training the neural network, deep learning

With the labeled set of candles and resulting trends, we trained various neural networks to try and separate the data which preceded upward movements and the points which preceded downward movements to see if the network can find a causal link. During our deep learning [7], which in our case was automatic discovery of structure using a feed forward network, we ran tests with and without RSI. We normalize RSI by subtracting the mean and dividing by the standard deviation value. We split our dataset randomly into training data and testing data (See data section). We fit our model on the train data to examine its accuracy on the test data and also to prevent over-fitting on the training data.

Firstly, we implemented a feed forward network with six layers [13] including five hidden layers of 20, 40, 40, 40, and 40 neurons and an output layer with one neuron. With a recurrent network, we needed much more data to get a high level of accuracy. Initially the best accuracy was 84.27% on the training set and 60.8% accuracy on the testing data. This result on the test set indicated that the model is over-fitting on the training data. Thus, to prevent over-fitting, we use the "dropout" approach for regularization. Dropout is a method in which we randomly drop (ignore) some neurons in some layers with a specified probability when training[17].

We use "ReLu" activation function in each hidden layer. ReLu is computationally less expensive than tanh and sigmoid because it involves simpler mathematical operations. It is an approximation of $f(x) = log(1+\exp(x))$. As our model is a classifier, in the output layer we use sigmoid activation function[7]. We use Adam optimizer for training our network. Adam combines the best properties of AdaGrad and RMSProp and is computationally efficient. We use values proposed in the original Adam paper for β_1 and β_2 which are 0.9 and 0.999, respectively.[10]

4. Results

We used a genetic approach to create random variations in the hyperparameters of our ANN model such as the number of neurons in each layer, the number of layers, dropout probability, adding and removing dropouts between two specific layers, and learning rate, to see if we can reach a better result. We examined the networks defined in Table 2.

Network Number	Layers and number of neurons	Learning rate	Training set accuracy (%)	Testing set accuracy (%)
1	$20, 40, 40, 40, 40, 1$	0.001	84.27	60.8
2	$20, d(0.2), 40, 40, 40, 40, 1$	0.001	70.2	64.29
3	$20, d(0.2), 20, 20, 20, 20, 1$	0.001	70.48	64.42
4	$20, d(0.2), 20, 1$	0.001	68.60	66.40
5	$20, d(0.2), 40, 1$	0.001	69.32	66.36
6	$20, d(0.2), 40, 20, 1$	0.001	69.68	65.04
7	$20, d(0.2), 40, d(0.2), 20, 1$	0.001	69.17	65.72
8	$20, d(0.3), 40, d(0.3), 20, 1$	0.001	68.26	65.78
9	$20, d(0.2), 20, d(0.2), 1$	0.001	68.28	66.40
10	$20, d(0.2), 40, d(0.2), 1$	0.001	68.33	66.04
11	$40, d(0.2), 40, 40, 40, 40, 1$	0.001	74.74 (73.7)	65.75 (65.9)
12	$40, d(0.3), 40, 1$	0.001	70.05	65.69
13	$40, d(0.3), 40, d(0.3), 1$	0.001	69.59	66.72
14	$50, d(0.4), 50, d(0.4), 1$	0.0001	68.14	66.04
15	$50, d(0.4), 50, d(0.4), 1$	0.01	67.48	66.01
16	$50, d(0.4), 50, d(0.4), 1$	0.001	70.11 (68.8)	68.37 (67.7)

Table 2: The examined network structures. The term "$d(x)$" represents a dropout which drops neurons with a probability of x. For example, the fourth network of the table represents a neural network of three layers with 20 neurons in each of the two hidden layers and a single neuron in the output layer. A dropout is used between the first and the second layer, and the neurons are dropped with a probability of 0.2. The figure in parenthesis is accuracy without using RSI Indicator.

After examination of several models defined in the Table 2, the best structure appeared to consists of an input layer of 13 neurons for the test set, two hidden layers of 50 and 50 neurons, and an output layer of one neuron. We use dropout between the first and second hidden layer and between the second hidden layer and the output layer. In both cases, we drop neurons with a probability of 0.4. If we consider a 4% error in the test set, the best network would be the 5 layer network.

With this network, we got the best results, nearly 70% accuracy in predicting uptrends both at training and testing just with data from the candles.

4.1. Discussion

The results showed accuracy is dependent also on the network structure. Nonetheless, with any structure the lowest accuracy we achieved was 64%, thus it appears to show that there is some information in the candles just when the price initiates a move. Without the event detection we got around 55 percent with the same neural network. Across all the various industries (442 stocks) we saw the ANN was able to predict trend statistically significant amount of times. The fact that the best network had few hidden layers but many neurons indicates there was a complex relationship between the three candles, thus approximating nonlinear transformation. There is also indication of little information being added by the indicator of RSI, as our best network without RSI was only 1% less accurate. Thus, given enough data points a neural network can replicate part of the functions of a technical analyst; the code can be found on Github `https://github.com/SalmanMarvasti/ResearchCode`.

The drawback with our approach is that outside transaction costs we need to use the threshold part of our event detection algorithm as well as the trained ANN. The event detection also acts as a noise removal filtering as we generally see large trend movement and not random or minor movements at event locations. Using the threshold part of our event detection algorithm, would work as follows: Our ANN requires either a 12-dimension data as input or 13 dimension including the candles for 3 days and the 14 day RSI. Lets make the following assumptions:

- D_i denotes the price at the low point in the day 'i' and therefore D_{i-1} is the price at the low point in the day before 'i'.
- M_i denotes the set of prices of 30 days before 'i' at the low point:

$$M_i = \{D_{i-31}, D_{i-30}, ..., D_{i-1}\}$$

- R_i denotes the returnized price of day 'i' at low point.
- RM_i denotes the set of returnized prices of M_i.

At each day 'i' our algorithm deals with D_i, D_{i-1}, D_{i-2} and M_{i-2} and goes through the following steps:

- Like the original algorithm T is calculated for M_{i-2} as below:

$$T = T_m * |1 - \log_{10} \frac{apple\_deviation}{M_{(i-2)}\_deviation}| * AVG_{abs}$$

  Where T_m was trained to be 4 and AVG_{abs} is the average of the absolute amount of RM_i prices.
- Then the algorithm calculates the following amounts:

$$V_1 = D_{i-2} - D_{i-3}\ ,\ V_2 = D_{i-1} - D_{i-3}\ ,\ V_3 = D_i - D_{i-3}$$

- If V_1, V_2 and V_3 are bigger than T and they all have the same sign (all positive or all negative) then an event is detected.
- When an event is detected (for instance on day 'i') then all prices(high, low, open, close) for the last 3 days (i-2, i-1 and i) are dynamically normalized as was explained before and the input to our ANN would be the normalized values of these 3 days plus the RSI value.

After the detection is done we have the required 13-dimension input and can get the output from the trained ANN.

4.2. Related Work

First Gately's book [6] and then Kutsurelis[4] work only worked on indexes not individual stocks and did not use candlesticks just prices. Indexes essentially act like low pass filters , in that they remove high frequency noise and while individual stocks are much more noisy. We more closely mimic the behavior of a technical analyst and are concerned with inflection points on each individual stock. We work with a large variety of stocks and show the general applicability of ANN on almost any stock not just relatively stable indexes.

5. Conclusion

It can be seen from the results that there is information in the candles 3 days before a major up or down trend and we can predict over 69% of the up/down trends using the described methods with a 3 layer neural network structure and a threshold based event detector. This must be due to significant market inefficiency and thus in practice markets are not completely efficient. The condition for our neural network to work is correctly detecting 'events', and the one described here does exactly that. For a technical analyst, a 70% win rate is good enough to generate returns on capital.

Some individual stocks are more predictable than others, but that can be a basis for future work. We used 10 years of data on many stocks, but for profitable exploitation of this learned trend, we need to also consider transaction cost relative to movement.

6. Acknowledgment

We would like to thank Fatemeh Abbasian Abyaneh for input on the use of candlesticks and ANN. We would like to thank Bonyad Melli Nokhbegan for funding part of this research.

Appendix A. Useful definitions

The relative strength index (RSI) is calculated using the following formula:

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

Where $RS = \frac{Average\ Gain\ of\ up\ periods}{Average\ Loss\ of\ down\ periods}$ during the specified time frame. The RSI provides a relative evaluation of the strength of a security's recent price performance, thus making it a momentum indicator. RSI values range from 0 to 100. The default time frame for comparing up periods to down periods is 14, as in 14 trading days. For more details see [20].

References

[1] Bollen, B., Inder, B., 2002. Estimating daily volatility in financial markets utilizing intraday data. Journal of Empirical Finance 9 (5), 551–562.

[2] BONDT, W. F. M., THALER, R., 1985. Does the stock market overreact? The Journal of Finance 40 (3), 793–805.

[3] Brock, W., Lakonishok, J., LeBaron, B., December 1992. Simple Technical Trading Rules and the Stochastic Properties of Stock Returns. Journal of Finance 47 (5), 1731–1764.

[4] E. Kutsurelis, J., 2001. Forecasting financial markets using neural networks: An analysis of methods and accuracy. Ph.D. thesis, Naval Postgraduate School, Monterey, California.

[5] Fama, E. F., 1970. Efficient capital markets: A review of theory and empirical work. The Journal of Finance 25 (2), 383–417.

[6] Gately, E., 1995. Neural Networks for Financial Forecasting. John Wiley & Sons, Inc., New York, NY, USA.

[7] Goodfellow, I., Bengio, Y., Courville, A., 2016. Deep Learning. MIT Press, `http://www.deeplearningbook.org`.

[8] Hulme, D., Xu, S., 2001. Application of genetic algorithms to the optimisation of neural network configuration for stock market forecasting. In: Stumptner, M., Corbett, D., Brooks, M. (Eds.), AI 2001: Advances in Artificial Intelligence. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 285–296.

[9] Jorgic, D., 2010. Can man groups ahl strategy return to its dizzy heights. CityWire 36 (2, Part 2), 3761 – 3773, `http://citywire.co.uk/new-model-adviser/news/can-man-groups-ahl-strategy-return-to-its-dizzy-heights/a380710`.

[10] Kingma, D. P., Ba, J., 2014. Adam: A method for stochastic optimization. CoRR abs/1412.6980.

[11] Kwan, J. W. C., Lam, K., So, M. K. P., Yu, P. L. H., "1999". Forecasting and trading strategies based on a price trend model. Journal of Forecasting 19 (6), 485–498.

[12] Lai, R. K., Fan, C.-Y., Huang, W.-H., Chang, P.-C., 2009. Evolving and clustering fuzzy decision tree for financial time series data forecasting. Expert Systems with Applications 36 (2, Part 2), 3761 – 3773.

[13] Larsen, J. I., 2010. Predicting stock prices using technical analysis and machine learning. Master's thesis, Institutt for datateknikk og informasjonsvitenskap.

[14] Lin, X., Yang, Z., Song, Y., 09 2011. Intelligent stock trading system based on improved technical analysis and echo state network. Expert Syst. Appl. 38, 11347–11354.

[15] Quant Software Toolkit, 2014. Quant software toolkit— Wikipedia, georgia tech. [Online accessed 29-September-2017].
URL `wiki.quantsoftware.org/index.php?title=QuantSoftware_ToolKit`

[16] Sitte, R., Sitte, J., May 2002. Neural networks approach to the random walk dilemma of financial time series. Applied Intelligence 16 (3), 163–171.

[17] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R., 2014. Dropout: A simple way to prevent neural networks from overfitting. Journal of Machine Learning Research 15, 1929–1958.
URL `http://jmlr.org/papers/v15/srivastava14a.html`

[18] Tharavanij, P., Siraprapasiri, V., Rajchamaha, K., 2017. Profitability of candlestick charting patterns in the stock exchange of thailand. SAGE Open 7 (4).

[19] White, July 1988. Economic prediction using neural networks: the case of ibm daily stock returns. In: IEEE 1988 International Conference on Neural Networks. pp. 451–458 vol.2.

[20] Wilder, J., 1978. New Concepts in Technical Trading Systems. Trend Research.
URL `https://books.google.fi/books?id=WesJAQAAMAAJ`