Mach Learn (2015) 101:187–209
DOI 10.1007/s10994-014-5480-x

Day trading profit maximization with multi-task learning and technical analysis

Zsolt Bitvai · Trevor Cohn

Received: 31 December 2013 / Accepted: 30 November 2014 / Published online: 25 December 2014

Abstract Stock price movements are claimed to be chaotic and unpredictable, and mainstream theories of finance refute the possibility of realizing risk-free profit through predictive modelling. Despite this, a large body of technical analysis work maintains that price movements can be predicted solely from historical market data, i.e., markets are not completely efficient. In this paper we seek to test this claim empirically by developing a novel stochastic trading algorithm in the form of a linear model with a profit maximization objective. Using this method we show improvements over the competitive buy-and-hold baseline over a decade of stock market data for several companies. We further extend the approach to allow for non-stationarity in time, and using multi-task learning to modulate between individual companies and the overall market. Both approaches further improve the predictive profit. Overall this work shows that market movements do exhibit predictable patterns as captured through technical analysis.

Keywords Multi-task learning · Technical analysis · Stock market trading

1 Introduction

A central tenet of financial theory is the Efficient Markets Hypothesis, which states that the market price reflects all available knowledge and accordingly no risk-free returns can be realized without access to non-public information. Despite its prevalence, this theory

Editors: Vadim Strijov, Richard Weber, Gerhard-Wilhelm Weber, and Süreyya Ozogur Akyüz.

Z. Bitvai (✉)
Department of Computer Science, University of Sheffield, Sheffield, UK
e-mail: z.bitvai@shef.ac.uk; bitvaizs@gmail.com

T. Cohn
Computing and Information Systems, The University of Melbourne, Melbourne, Australia
e-mail: t.cohn@unimelb.edu.au

is not universally accepted, as it cannot explain several small but significant examples of inefficiencies commonly exhibited in markets. A large body of technical analysis techniques claim that recurring patterns can be identified from historical market data, which can be used to realize risk-free profits (Lo et al. 2000). Such techniques are widely used as part of active portfolio management, which aims to outperform passive 'buy-and-hold' management strategies. It is an open question as to the scientific validity of these claims, namely whether active trading and technical analysis can reliably realize above market returns. In this paper we seek to test the question of whether markets are completely efficient by empirical validation using predictive modelling. We demonstrate that historical market movements and technical analysis contain predictive power for forecasting daily returns in an active trading scenario.

This paper proposes a novel learning algorithm for profit maximization in an active trading scenario, where stocks are bought and sold on a daily basis. We show how trading can be modelled using a similar formulation to logistic regression, permitting a simple gradient based training algorithm and a straight-forward means of prediction on unseen data. This technique allows for recent market data, represented using technical analysis basis functions, to drive investment decisions. Compared to the manual use of technical analysis, where a trader interprets the identified patterns, here our training algorithm assists the trader in deciding which technical analysis indicators are important, and how these should be used in their trading decisions.

To reflect the idiosyncrasies of individual companies, we present a multi-task learning approach that is suitable for jointly modelling several companies on the stock market. This trains a series of per-company models, subject to a mean regularization term which encourages global parameter sharing. We show that this improves over independent modelling of each company, or joint modelling of all companies with tied parameters. A second question is how to handle changing market conditions over time, which is of particular importance in our setting as speculative opportunities are likely to change over time as they have been identified and removed by market participants. For this purpose we use a simple time based regularizer which permits model parameters to vary smoothly with time, which is shown to result in further improvements in predictive profit.

This paper seeks to answer several research questions. The first is regarding market efficiency, namely whether there are systematic inefficiencies that can be exploited using statistical models. To answer this, we show that excess profits are achieved using our active trading models when compared to simple baseline methods, such as buy-and-hold. A second aspect of this research question is whether technical analysis can improve active trading models compared to using only recent price values, which we also show to be the case, although this difference is less dramatic. Together these results provide an empirical justification of active trading and technical analysis, refuting the efficiency arguments of financial theory.

The second set of research objectives concerns modelling. Our approach develops a model of financial trading, and a training method for optimizing trading profits. To test the validity of explicit profit maximization, we compare against squared error loss, the most common regression objective, and show significant outperformance. Our modelling includes multi-task learning over several different companies and over time, which we show leads to substantial benefits in profit over baseline models trained on individual companies, pooling together the dataset, or ignoring the effect of time. Overall these results suggest that fine grained modelling is useful, including modelling non-stationarities, but there is information from the global data. Multi-task learning is an effective means of balancing these two criteria.

The remainder of the paper is structured as follows. In Sect. 2 we review the financial literature on market efficiency, as well as the few papers applying machine learning techniques to stock price prediction or portfolio optimization. In Sect. 3 we present our novel

profit maximization model, and extensions to enable joint multi-task learning over several companies as well as temporal adaptation to handle non-stationarities. Next we turn to the evaluation, starting in Sect. 4 with a brief overview of technical analysis before presenting the validation methodology and comparative baselines in Sect. 5. Experimental results are presented in Sect. 6, evaluating the algorithm in several realistic trading scenarios.

2 Literature review

2.1 Risk-free profits

Most theoretical finance works maintain that markets are efficient and as such Modern Portfolio Theory (Markowitz 1952) and the Capital Asset Pricing Model (Sharpe 1964) state that no risk-free excess profits can be made. That means the best one can do is maximize the returns for a given level of risk. In practical terms for markets to be fully efficient the following must be true: universal access to high-speed and advanced systems of pricing analysis; a universally accepted analysis system of pricing stocks; an absolute absence of human emotion in investment decision-making; the willingness of all investors to accept that their returns or losses will be exactly identical to all other market participants.

There is some evidence that actively managed funds under-perform passively managed index funds by their added expenses (Jensen 1968). Therefore, a simple model with maximum diversification that spreads the risk and invests equally in all assets yields better returns than a complex model that aims to select stocks by active analysis. Stock markets are highly chaotic systems with very high levels of noise (Magdon-Ismail et al. 1998; Ghosn and Bengio 1997). Therefore, the price movement of companies on the market are fundamentally unpredictable (Magdon-Ismail et al. 1998). The Efficient Market Hypothesis states that the price of a stock already contains all the available information about the asset, therefore the market is informationally efficient (Malkiel 2003). According to this theory, in the long term one cannot beat the market consistently through speculation.

However, Malkiel notes that in the real world there are market phenomena that can be interpreted as signs of inefficiencies. One such example is short term momentum and underreaction to new information (Lo et al. 2000). This may be attributed to the psychological bandwagon effect (Shiller et al. 1984). Another example is long-run return reversals, which means that stock prices are expected to revert to their mean (Kahneman and Tversky 1979). A third source is seasonal patterns, for example in January higher returns could be achieved due to tax filing in December (Haugen and Lakonishok 1987). According to the Size effect, smaller companies tend to outperform larger companies (Bondt and Thaler 1985). The equity risk premium puzzle shows that investors prefer bonds to common stocks even when that results in lower risk adjusted returns for them (Weil 1989). The 1987 market crash and the 1990s Internet bubble can be regarded as short term market inefficiencies (Malkiel 2003). Prospect theory and advances in behavioral economics have shown that humans are subject to cognitive and emotional biases and therefore they are prone to make sub-optimal financial decisions (Tversky and Kahneman 1974; Kahneman and Tversky 1979). Nevertheless, it has been shown that soon after publishing the discovery of such patterns that may enable excess risk adjusted profits to be made, these opportunities are quickly exploited by investors (Malkiel 2003). These observations suggest that active portfolio management could outperform passive management by exploiting inefficiencies in the market.

Although the Efficient Market Hypothesis rejects its validity, technical analysis is commonly applied to stock market forecasting. Kim et al. (2002) measured the distributional

differences of a select few technical indicators that were chosen by expert knowledge instead of automated statistical analysis. They used profit per trade as a measure to evaluate performance of a trading system with transaction cost. Cha and Chan (2000) proposed a system that output buy, sell and hold trading signals for stocks that were not part of the training set. In his system he extracted local maxima and minima of prices and trained a neural network to predict these points. They used a dataset of around 800 trading days with three companies on the Hong Kong Stock Exchange, proposing to invest proportional to the strength of the trading signal, but leaving the implementation of such an objective to future work. In our system, profit per trade was used as a measure of performance but we relied on automated learning methods to extract relevant information from the dataset, instead of expert knowledge. Similar to their work, we consider investment based on the predictive signal determined by a learning algorithm that invests based on the strength of the signal after squashing it through a sigmoid function. Our modelling approach is different to theirs as we do not explicitly model peaks and troughs, but consider making daily trades based on the recent historical market context. Moreover we train on several thousand data points across almost a hundred companies on the London Stock Exchange, a much larger dataset than that used in Cha and Chan (2000).

2.2 Joint modelling

Multi-task learning has been investigated as an effective way of improving predictions of machine learning models. Ghosn and Bengio (1997) studied whether sharing hidden layers of neural networks between companies could improve the selection of high return stocks. They trained neural net parameters on one company and used them to produce predictions for other stocks, and also examined whether selective parameter sharing of various neural net layers could aid prediction. They report significant above-market returns with a trading system based on this model. Bengio (1997) experimented with portfolio optimization over 35 stocks using one model for all companies, finding that the best results were obtained by sharing neural network parameters across companies. Finally, Cha and Chan (2000) explored domain adaptation by training a model on all stocks but one, and testing on the held-out stock. These papers show that relationships between companies contain valuable information that can be mined for trading. Therefore, in our experiments separate models for different companies were related to one another during training via multi-task learning. Our working assumption is that the mechanisms for predicting individual stock movements will be highly correlated through model regularization towards a common shared component.

2.3 Temporal variations

Multi-task learning can also be used for temporal data. Caruana (1997) explored time series prediction where he fitted a neural network with shared parameters to produce outputs for multiple simultaneous tasks. He noted that this kind of learning is especially useful for harder longer term predictions where best results can be obtained when the middle task is predicted from previous and later tasks. To take temporal changes into account Bengio (1997) trained on a window of data, which he shifted through time. This is a naive way for accounting for time as it assumes non stationarity but sacrifices information sharing that goes beyond the window. In another experiment, Bengio used a recurrent neural network, with five macro and micro economic variables as external inputs. In this paper we consider a linear model, with its outputs mapped to trading actions via a non-linear sigmoid function. This is akin to a one layer neural network. Here we account for time using a regularization method which

ensures smoothness between adjacent time periods, where model parameters were stratified by month. Our experimental setup focused on extrapolation a month into the future, rather than the easier problem of interpolation, as done in Caruana (1997).

2.4 Model formulation

Minimizing squared loss is inappropriate for the reason that profit and prediction accuracy are often not significantly correlated in a financial context. Instead Bengio (1997) proposes replacing the prediction criterion in training with a financial criterion. Bengio identified optimal weights to determine the share of each stock or asset in the portfolio at a given time step which was one month. We optimized for profit which is a different type of financial criterion suited to our setting of day trading.

3 Model

An active trading scenario is considered, where a trader makes an investment every day which is then reversed the following day. Trades are based on the outputs of a linear model over technical analysis features encoding recent market history, as described in Sect. 4. Then the problem is how to trade based on the real valued prediction, considering the type of trade (buy or sell) and the magnitude of the trade. A linear model is chosen due to its simplicity, although the approach would accommodate a more complex non-linear modelling approach, such as a multi-layer neural network.

The invested money on each day is proportional to the confidence level of the prediction, which is denoted by the absolute sigmoid value. The sigmoid is a useful function because it is easily differentiable, thus resulting in an error term which can be easily minimized. The hyperbolic tangent sigmoid function is applied to the simple linear model of the input,

$$p(\mathbf{x}, \mathbf{w}) = \tanh(\mathbf{w}^\intercal \mathbf{x}) \,, \tag{1}$$

resulting in a prediction value $p \in [-1, 1]$ which determines the trade type (buy or sell, depending on $\text{sign}(p)$) and magnitude of investment ($|p|$). Figure 1 illustrates the trading actions resulting from several different inputs. Here a negative prediction denotes sell the stock whereas a positive prediction denotes buying the stock, with a maximum investment of £ $\pm$ 1.

In order to train such a model, we use as the response variable the relative price movement from the stock. If the stock price falls by 50 %, then the three scenarios illustrated in Fig. 1 result in profits of roughly £-0.5, £0 and £0.5. More formally, the target of the prediction $y(d) \in [-1, \infty)$ is the return on investment,

$$y(d) = \frac{s_{\text{close}}(d+1) - s_{\text{close}}(d)}{s_{\text{close}}(d)} \,,$$

where d is the trading day and $s_{\text{close}}(d)$ is the closing price of the stock on day d.

Consider now a linear regression baseline algorithm where the optimum weights are found that minimize the error of the predictions as measured by the squared difference between the targets and the predictions. This has the benefit of a closed form solution for the optimal weights (Rogers and Girolami 2011), however it has two short-comings. First, it is not clear how to trade based on the signal, which can vary over a wide range of values, and secondly the training loss is highly dissimilar to trading profit. To account for the first problem, we can adjust the model predictions by first normalizing by the standard deviation of the prediction,

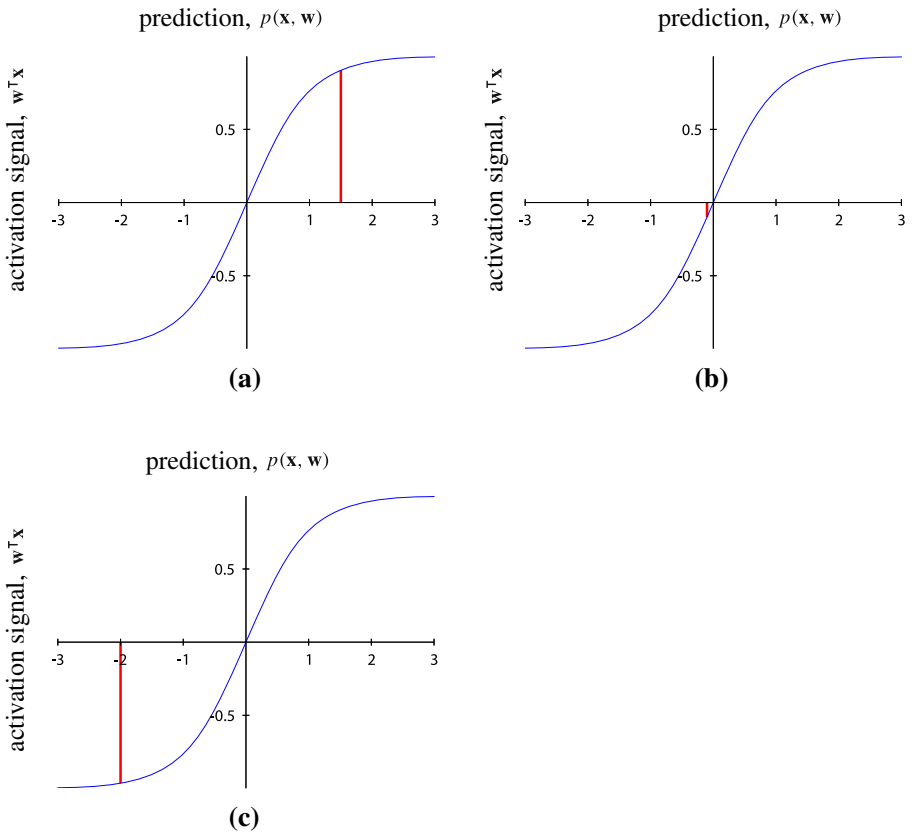

Fig. 1 Prediction signal: confident buy, unconfident (short) sell, confident (short) sell

after which we apply the hyperbolic sigmoid in (1) to obtain a trading action. This is a rough heuristic to ensure the scale of predictions are comparable to that of other models.

The second problem of the mismatch between the training and the testing loss is more insidious, as the sum of squared errors does not resemble profit. To illustrate the difference between the two objectives, consider the case when the stock price rises. The squared error loss penalizes predictions which do not exactly match the price rise, irrespective of whether they are above and below the true price movement. This makes little sense, as all buy predictions should be rewarded, including extremely high values. A similar effect occurs for negative predictions, which attract heavy penalties out of keeping with the loss from trading.

A better approach is to integrate the sigmoid into the loss function such that optimization can be performed directly for profit instead of prediction accuracy. For this reason instead of minimizing the squared error of the prediction, as for the linear regression baseline, we seek to maximize profit directly. This gives rise to the profit (utility) objective,

$$u(\mathbf{x}, \mathbf{w}, y) = p(\mathbf{x}, \mathbf{w})y = \tanh(\mathbf{w}^\mathsf{T}\mathbf{x})y$$

which multiplies the prediction $p(\mathbf{x}, \mathbf{w})$ by the relative price movement of the stock, y. This corresponds to the profit realized over a single trade.

Table 1 Notation

Notation	Explanation
X	Training data of technical analysis indicators, $\mathsf{X} \in \mathbb{R}^{C \times M \times D_{c,m} \times T}$
Y	Targets, relative price movement of stocks, $\mathsf{Y} \in \mathbb{R}^{C \times M \times D_{c,m}}$
W	Model parameters, $\mathsf{W} \in \mathbb{R}^{C \times M \times T}$
λ_c	Regularization coefficient between companies
λ_m	Regularization coefficient between trading months
λ_l	Regularization coefficient for weights
C	Number of companies
M	Number of trading months
$D_{c,m}$	Number of trading days in company c and month m
T	Number of technical indicators
R_c	Company regularizer term
R_m	Time regularizer term
R_l	L-2 regularizer term
U	Utility or profit in entire dataset
S	Raw market data

It is assumed that fractions of stocks can be traded and that stocks that are not currently possessed can be short sold. Further, trades can be executed at the closing price of the stocks of the companies each day for the invested amount. Last, transaction costs are not factored in the prediction signal, although this could be implemented using a form of hinge loss. These assumptions are designed to be reasonable while still yielding a simple and differentiable objective for training the model. Table 1 gives an explanation of the notation used throughout the paper.

The overall utility or profit U is obtained by aggregating the profit over all time periods, for which we compare two alternative methods:

$$U(\mathsf{W}) = \sum_{c=1}^{C} \sum_{d=1}^{D_c} \tanh(\mathbf{w_c}^\intercal \mathbf{x_{c,d}}) y_{c,d} \tag{2}$$

$$U'(\mathsf{W}) = \ln \prod_{c=1}^{C} \prod_{d=1}^{D_c} \left(\tanh(\mathbf{w_c}^\intercal \mathbf{x_{c,d}}) y_{c,d} + 1 \right). \tag{3}$$

The first objective in (2) simply considers the total aggregate profit, which is appropriate if each trade is performed independently based on the same investment budget. The second objective (3) allows for compounding of profits whereby the proceeds from day d are reinvested and subsequent trades are based on this revised budget. In Eq. (3) the constant 1 is added to the daily utility to reflect the multiplicative change in bank balance, and the logarithm is applied to simplify the formulation.

The training objective is to maximize the utility, as defined in (2) or (3), including an additive L_2 regularizer term to bias weights towards zero,

$$R_l = \lambda_l \sum_{t=1}^{T} \|\mathsf{W}_{\cdot,\cdot,\mathrm{t}}\|_F^2, \tag{4}$$

where $\|\cdot\|_F$ is the Frobenius norm of the weights and the coefficient λ_l modulates the effect of the regularization term relative to the profit. In the optimization, the loss function which measures the negative utility is minimized with respect to the weights W in order to learn the optimal model weights,

$$\hat{\mathsf{W}} = \underset{\mathsf{W}}{\operatorname{argmin}} - U(\mathsf{W}) + R_l(\mathsf{W})\,. \tag{5}$$

The optimization in (5) is solved using the L-BFGS algorithm (Byrd et al. 1995), a quasi-Newton method for convex optimization. Note that the objective in (5) is non-convex, and therefore gradient based optimization may not find the global optimum. However, our experiments showed that L-BFGS was effective: it consistently converged and was robust to different starting conditions. The L-BFGS optimizer requires first order partial derivatives of the objective function. The gradient of each component of the objective are as follows:

$$\begin{aligned}
\frac{\partial}{\partial w_{c,t}} U &= \sum_{d=1}^{D} x_{c,d,t}\, y_{c,d}\, \cosh^{-2}(\mathbf{w_c}^\intercal \mathbf{x_{c,d}}) \\
\frac{\partial}{\partial w_{c,t}} U' &= \frac{\sum_{d=1}^{D} y_{c,d} x_{c,d,t} \cosh^{-2}(\mathbf{w_c}^\intercal \mathbf{x_{c,d}}) \prod_{d'=1, d' \neq d}^{D} y_{c,d'} \tanh(\mathbf{w_c}^\intercal \mathbf{x_{c,d'}}) + 1}{\prod_{d=1}^{D} y_{c,d} \tanh(\mathbf{w_c}^\intercal \mathbf{x_{c,d}}) + 1} \\
\frac{\partial}{\partial w_{c,t}} R_l &= 2\lambda_l w_{c,t}\,,
\end{aligned}$$

where the first two equations are the partial derivatives of the two alternative loss functions—for non-compound (2) and compound profits (3), respectively—and the last equation is the gradient of the regularizer term.

3.1 Multi-task learning

The algorithm makes a prediction for each company c each trading day d based on the data vector $\mathbf{x_{c,d}}$. Therefore, it produces multiple simultaneous outputs, one for each company. As new data points are acquired for each daily time step, a prediction is made for each company at the same time. We assume that different companies operate under different conditions, which in turn affect their trading on the stock market. Therefore each company is best modelled using different parameters. However as our dataset consists of companies which operate in the same market and are all 'blue-chip' stocks with high market capitalization, we would also expect that their price behaviors are linked.

Balancing these two effects is achieved by mean regularized multi-task learning (Evgeniou and Pontil 2004). This method learns multiple related tasks jointly, which can improve accuracy for the primary task learned. Despite its simplicity among other multi-task learning methods, it can be highly effective. This is implemented by including a new penalty term, the 'company regularizer',

$$R_c = \lambda_c \sum_{c=1}^{C} \|\mathsf{W}_{c,\cdot,\cdot} - \frac{1}{C} \sum_{c'=1}^{C} \mathsf{W}_{c',\cdot,\cdot}\|_F^2\,, \tag{6}$$

which penalizes differences between company specific weights and the mean weights across all companies. When the company regularizer coefficient, λ_c, is set to zero there is no regularization between companies, such that each company is modelled independently. Conversely, setting $\lambda_c = \infty$ means all parameters are tied, i.e., all company data is pooled together.

3.2 Non-stationarity

Markets are dynamic systems since inefficiencies will be eventually discovered and exploited by traders, and thus exploitable signals in the market data may fade over time. Besides evolving speculative behaviors, it is likely that the underlying dynamics of financial systems and changes to external economic conditions (unknown to our model) result in a non-stationarity process. Although the model may need to change with time, it is unlikely to change rapidly over a short period, but rather evolve smoothly with time.[1]

To encode the assumption of smooth variation with time, we elect to use an additional time regularization term. This is related to other temporal modelling methods such as the Fused Lasso (Tibshirani et al. 2005), which penalizes absolute changes in weights between adjacent time intervals using the L_1 norm. Here instead we use a L_2 term,

$$R_m = \lambda_m \sum_{m=2}^{M} \|\mathsf{W}_{\cdot,\mathsf{m},\cdot} - \mathsf{W}_{\cdot,\mathsf{m}-1,\cdot}\|_F^2 \,, \tag{7}$$

where m is the trading month, used as the granularity of our temporal modelling. The regularizer penalizes weight differences for adjacent trading months, m and $m-1$, such that weights smoothly vary over time. The extreme behavior of the time regularizer represents either pooling ($\lambda_m = \infty$), allowing for no temporal variations, or independent modelling of each month ($\lambda_m = 0$). The final optimization objective, including both company and time regularization term, is now

$$\hat{\mathsf{W}} = \underset{\mathsf{W}}{\operatorname{argmin}} - U(\mathsf{W}) + R_l(\mathsf{W}) + R_c(\mathsf{W}) + R_m(\mathsf{W}) \,.$$

This objective discourages large differences between parameters for adjacent trading months, and also discourages large differences between the weights for individual companies versus the average over all companies. These biases are balanced against data fit, and consequently we expect the model to learn different parameter values for each task as needed to maximize training profit. The partial derivatives of the regularization terms are as follows:

$$\frac{\partial}{\partial w_{c,m,t}} R_c = 2\lambda_c \left(w_{c,m,t} - \frac{1}{C} \sum_{c'=1}^{C} w_{c',m,t} \right)$$

$$\frac{\partial}{\partial w_{c,m,t}} R_m = \begin{cases} 2\lambda_m (w_{c,m,t} - w_{c,m-1,t}) & \text{if } m = M \\ 2\lambda_m (w_{c,m,t} - w_{c,m+1,t}) & \text{if } m = 1 \\ 2\lambda_m (2w_{c,m,t} - w_{c,m-1,t} - w_{c,m+1,t}) & \text{otherwise} \end{cases} .$$

The multi-task regularizers facilitate the sharing of data statistics across task models, while also limiting overfitting the weak signal characteristic of noisy financial datasets.

4 Technical analysis

Now we consider experimental validation of our trading model. Our dataset was sourced from Google Finance, taking market data over the period 2000–2013 for all companies constituting the FTSE index of the top 100 stocks by market value in the United Kingdom.

[1] Note that market crashes and other sudden events will not be handled well by this approach, a more sophisticated method would be needed to handle such cases.

The FTSE index composition changed during this period, due to companies being excluded or included, and we retain the 64 companies which remained in the index for the full period (see "Appendix 1" for the company ticker symbols).[2] The market data was cleaned to account for incorrect adjustment for dividends, stock splits or merges by manually verifying all daily price movements above 10 %, and excluding any improperly adjusted prices.

The dataset is divided into individual 'tasks' by company and by trading month. The daily market data is transformed by applying a suite of technical analysis features, which are then each normalized and standardized before used in the learning algorithm.

Technical analysis can reveal behavioral patterns in trading activity on stock markets. It has the potential to uncover behavioral cues of market participants and capture psychological biases such as loss aversion or risk avoidance. In its arsenal, sound statistical and quantitative analysis methods can be found in addition to heuristic pattern analysis such as candlestick pattern matching functions. The three pillars of technical analysis are history tends to repeat itself, prices move in trends, and market action discounts everything (Lo et al. 2000; Neely et al. 1997). This means that all past, current and even future information is discounted into the markets, such as emotions of investors to inflation or pending earnings announcements by companies. Therefore, technical analysis treats fundamental analysis, which analyzes external factors such as company performance reports and economic indicators, redundant for making predictions.

We apply technical analysis to the market data for each company using the TA-lib technical analysis library.[3] This transforms raw market data series, S, of open, high, low, close and volume values with a family of technical analysis functions, ϕ, to obtain the technical indicator series representation of the data, $\mathsf{X} = \phi(\mathsf{S})$. Functions are applied from the family of overlap studies, momentum, volume, cycle, price transform, volatility and pattern recognition indicators. For the list of technical indicators, please see "Appendix 2".[4] Figure 2 depicts an example of a few technical indicator values calculated on a single stock. Bollinger Bands return two time series that are two standard deviations away from the moving average (MVA), which is a measure of volatility. Moving Average Convergence Divergence (MACD) subtracts the 26-day exponential moving average (EMA) from the 12-day EMA. This is used as a rough heuristic by traders as follows: when the MACD is above the signal line, it recommends buying while when it is below, it recommends selling. A dramatic rise suggests a small speculative bubble and end of the current trend. Williams' %R compares the closing price to the high-low range over 14 days. A high value indicates that a stock is oversold, while a low value shows that it is overbought.

The first six months of the dataset is reserved to bootstrap the indicator calculation. Most indicators rely on a history of market data for analysis so as to give forecasts. As the TAlib suite provides a large range of technical analysis indicators, we process them in a simple and agnostic manner to derive the feature representation of our data. The parameters of most indicators were left at their default values. In cases where an indicator returns multiple value series, one feature was created for each series, and indicators returning invalid or constant values were discarded. Overall, this resulted in a dataset with $T = 133$ features, representing the market conditions for a given company on a given trading day.

[2] This confers a survivor bias to our results, as any companies suffering extremely poor performance or bankruptcy during the period have been excluded. However note that this bias also affects our baselines, in particular inflating the performance of the passive 'buy-and-hold' strategy.

[3] Technical Analysis library—http://ta-lib.org/.

[4] Statistical functions, mathematical transforms and operators are not used because these are helper functions of other more complex functions. In addition, MAVP, MACDFIX and ROCR100 are omitted due to the fact that equivalent functions, such as scaled versions, are already present in the library.

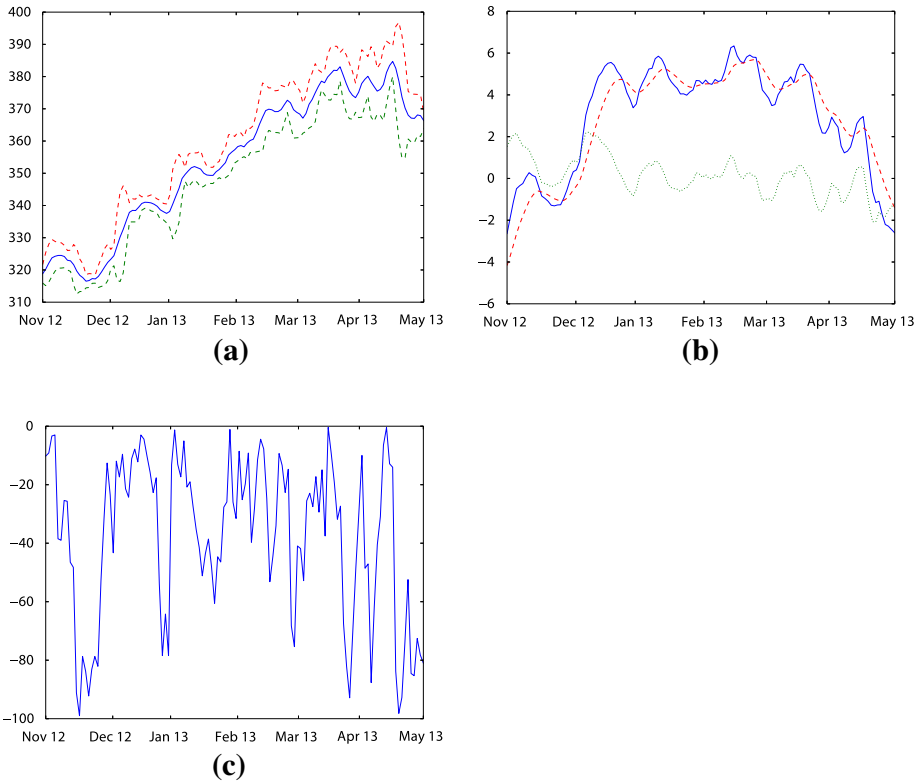

Fig. 2 Bollinger bands, **a** MVA (*blue solid*), +2STD (*red dashed*), -2STD (*green dashed*); **b** MACD (*blue solid*), EMA (*red dashed*), DIV (*green dotted*); Williams' %R; **c** for Tesco, Nov 2012–May 2013 (Color figure online)

5 Validation

In order to evaluate the performance of the algorithm, a sliding window experimental setup is used, as illustrated in Fig. 3. For each evaluation round, one and a half years of data across all companies is used for training, the following month is used for validation and the subsequent month is used for testing. In the case of time-varying models, the weights from the most recent month are used for validation and testing. Validation is only performed on the first window of data for efficiency reasons, and is used to select the three regularization coefficients which control the importance of the penalty terms of the loss function. After the first validation and evaluation, the training and testing window is shifted by one month and the process is repeated until the end of the dataset is reached in 2013. This evaluation setup is designed to match a trading scenario, where short term extrapolation predictions are needed to guide investment decisions.

The model has several parameters: the weight, λ_l, company, λ_c, and time, λ_m, regularizer coefficients (see Eqs. 4, 6, 7). These need to be tuned to control the relative effect of the data fit versus the regularization for weight magnitude, deviation from the market mean, and weight change with time, respectively. These parameters are automatically tuned by grid search using a logarithmic scale between 10^{-10} and 10^{10}. In addition, 0 and infinity are added to the possible values to simulate independent task learning and pooling. The performance

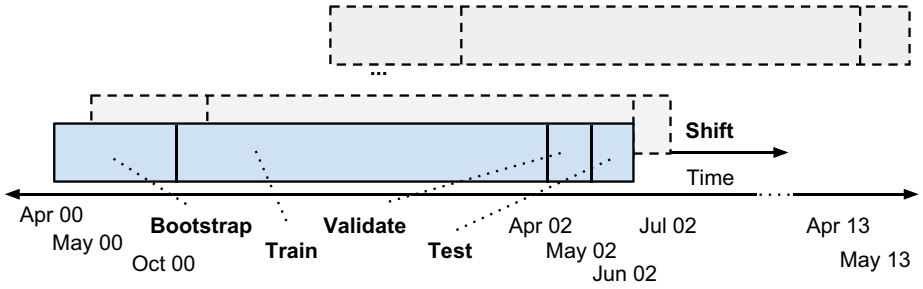

Fig. 3 Sliding window evaluation

of the model is measured on the validation set, and the values that give the highest validation profit are selected.

We evaluate against several baseline measures:

Random: In random trading, the predictions are produced by a uniform distribution between -1 and $+1$. In this case the expected predictions are 0 and consequently expected trading profit is also 0.

Buy-and-hold: Setting all predictions to +1 is similar to a long term buy-and-hold position, but without reinvestment of daily profits or losses. A buy and hold strategy can be seen as spreading the risk between all possessed assets, in this case the FTSE index, which confers diversification benefits as advocated by Modern Portfolio Theory.

Short-and-hold: There is an always sell strategy which is the inverse of buy-and-hold, above. This corresponds to a long term short position.

Ridge regression: To test the importance of profit maximization, we consider also ridge regression (Hoerl and Kennard 1970) which instead minimizes squared error subject to an L_2 regularization term. In this case we do not perform multi-task learning over companies or time, but instead fit a single regression model per company and ignore temporal variation. For fairness of evaluation, we use the sliding window evaluation and validation method for fitting the regularization hyper-parameter, as described in Sect. 5.

6 Evaluation

Our experimental validation seeks to provide empirical answers to several research questions: whether our approach outperforms simple baselines, the importance of using a profit objective, the importance of technical analysis features, and the how multi-task learning affects performance, both over individual companies and over time. We now address each of these questions in turn.

The overall profit results are shown in Table 2,[5] where the models in the top portion of the table using our techniques for profit maximization, and in the bottom portion, the baseline techniques. It is clear that the profit-trained models (profits of £45.3–£64.9) outperform the baselines (−£26.2–£26.2) by a large margin. During the testing period the market went through peaks and troughs, including two market crashes, with an overall gradual rise, leading to a positive profit from the buy-and-hold strategy. Despite this our method was able to achieve excess profits of up to £38.7. The best model predictions significantly outperform random

[5] We present other evaluation metrics, including annual return in Sect. 6.1.

Table 2 Trading results based on a £1 budget, evaluated on 2002–2013 FTSE data over 62 companies

Method	Absolute profit
Multi task, tech anal, comp-time reg	£64.9
Multi task, tech anal, comp-time reg, compound prof	£62.1
Multi task, tech anal, comp reg	£61.1
Multi task, tech anal, time reg	£55.9
Single task, tech anal	£49.3
Single task, window of returns	£45.3
Buy-and-hold	£26.2
Ridge regression	£9.0
Random	£0.0 ± 4.6
Short-and-hold	£-26.2

trading.[6] Our method has identified important patterns in the data to achieve excess profits, demonstrating that market inefficiencies do exist in historical FTSE market data and can be exploited. Comparing the ridge regression baseline against the equivalent model `Single task, tech anal` trained with a different loss function, shows that optimizing for profit outperforms squared error loss by £40.3. In fact, ridge regression performed poorly, significantly outperforming only random trading and Short-and-hold. Using the appropriate loss function was clearly the single most important modelling decision in terms of net profit. Note however that using the compounding or non-compounding formulation of profit made little difference, with compounding under-performing by £2.8, although this did speed up training.

A related question regards the utility of technical analysis features (as described in Sect. 4). Our intuition was that many of these features could be useful, and using many together would provide a rich and expressive basis for (non-linear) modelling and thus outperform a linear autoregressive model. The rows in Table 2 labelled `Single task, tech anal` and `Single task, window of returns` differ in their feature representation: the former uses our 133 technical analysis features, while the window of returns uses as features the market history for the past 90 days, i.e., an autoregressive model with a 90 day time lag. The difference in profit between the two systems is modest but statistically significant, which shows evidence of the utility of technical analysis. An advantage of using technical indicators is that they can perform non-linear transformations on the market data. Therefore, using this basis allows our linear model to learn non-linear relationships over this time-series data. An interesting extension would be to allow for non-linear functions, which could remove the need for hand-engineered technical analysis features.

Next, we consider the importance of multi-task learning. Starting with the `Single task, tech anal` we can see that multi-task learning over companies (`Multi task, tech anal, comp reg`) provides a statistically significant improvement of £10.8 over independent models per company. Moreover, the temporal regularizer (`Multi task, tech anal, time reg`) also results in a significant gain of £6.6, showing that our method for modelling non-stationarity is effective. Together the two multi-task regularizers (`Multi task, tech anal, comp-time reg`) provide a significant profit increase

[6] Significance hereinafter is accessed using the Wilcoxon ranked-sign test, $p < 0.01$.

Table 3 Top indicators for Capita plc and BG Group plc with multitask learning

Company	Date	Top positive	Top negative
CPI	2004-03-31	MACDEXT	CDLCLOSINGMARUBOZU
CPI	2005-08-31	CDLEVENINGSTAR	CDLCLOSINGMARUBOZU
CPI	2008-08-31	CDLHIKKAKE	CDLHOMINGPIGEON
CPI	2009-07-31	CDLPIERCING	CDLCLOSINGMARUBOZU
CPI	2009-11-30	ADXR	CDLCLOSINGMARUBOZU
CPI	2010-03-31	CCI	CDLCLOSINGMARUBOZU
BG	2004-07-31	CDLHANGINGMAN	CDLCLOSINGMARUBOZU
BG	2006-06-30	CDLHANGINGMAN	CDLCLOSINGMARUBOZU
BG	2006-08-31	CDLHANGINGMAN	CDLCLOSINGMARUBOZU
BG	2008-09-30	HT DCPERIOD	HT PHASOR
BG	2011-02-28	HT SINE	STOCHRSI
BG	2012-10-31	HT DCPERIOD	STOCH

of £15.6 over the single task method.[7] The magnitude of this improvement suggests that the two regularization methods work in complementary ways, and both identify important aspects in our data.

By examining the feature weights during testing, it is possible to determine which weights were important and how they contributed to predictions. For the top positive and negative indicators by weights sampled at random intervals for randomly selected companies CPI and BG, see Table 3. The fact that some of the top indicators are different in various tasks justifies having an individual model for each company separately and relating them to the market average model via the company regularizer. It also shows that the time regularization can provide additional flexibility in the model. However, note that the top negative feature persisted for a long time for many companies. The tasks were not forced to share the same weights as in single task learning, but rather they could learn their own weights which provided better predictions during testing.

The Hilbert Transform is a technique used to generate in-phase and quadrature components of a de-trended real-valued signal, such as price series, in order to analyze variations of the instantaneous phase and amplitude. An interesting note is that HT PHASOR, Hilbert Transform—Phase components, frequently appeared in the top four positive indicators for companies during 2002–2003 while it also appeared in the top four negative indicators for some of the same companies during 2005–2006. The HT DCPERIOD, or Hilbert Transform—Dominant Cycle Period, is an adaptive stochastic indicator. Given a cyclic price signal, it attempts to identify the beginning and end of the cycle. CCI, or Commodity Channel Index, measures the current price level relative to an average price level over a period of time. It can be used to track the beginning of a new trend or warn of extreme conditions. Both HT DCPERIOD and CCI had positive weights which suggests that they can provide information with regard to the beginning of new price trends.

Another interesting observation is the inclusion of behavioral pattern matching indicators starting with the CDL prefix. CDLCLOSINGMARUBOZU frequently appeared in the top negative indicators for many companies during various periods. Marubozu is positive when

[7] The values learned for the regularizer coefficients were $\lambda_c = 10$ (company), $\lambda_m = 10^4$ (time), and $\lambda_l = 10^{-4}$ (weight magnitude). Note that none of these values are extreme, illustrating that multi-task learning is favored over independent learning or pooling.

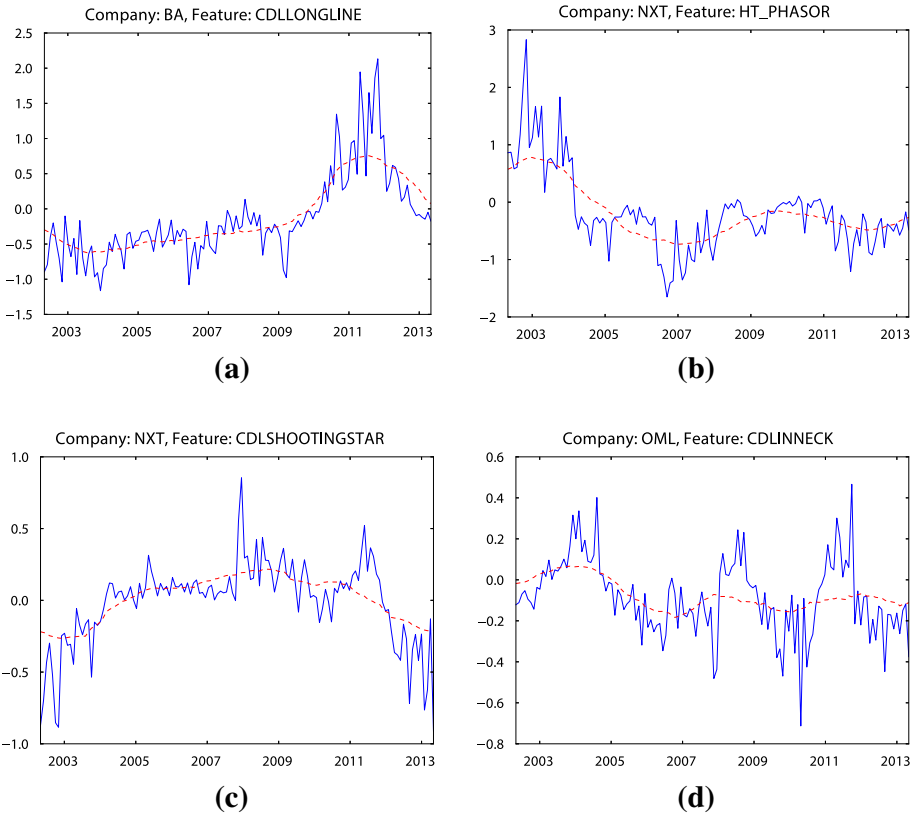

Fig. 4 Time variations in feature weights (*blue line*) with 30-month moving average (*red dotted line*) with multi-task learning (Color figure online)

the closing price was at a high during the period, indicating a bull market, and negative when it was at a low, indicating a bear market. Hence, a negative weight would contribute to a SELL signal when positive and to a BUY signal when negative. This could indicate trend reversal in stock prices or a reversion to longer term average returns. CDLHANGINGMAN was another top positive feature. It indicates that despite a large sell-off, the buyers remain in control and manage to push the prices further up in the short term before an eventual drop. With a positive weight it has the effect of buying stock shortly before the peak of the price trend, which is still a good time to do so.

Next, the change of the feature weights is examined over time. Figure 4 demonstrates that the predictive weight of feature CDLLONGLINE was fairly negative during the initial testing period, but later it became positive. On the other hand, HT PHASOR behaved the opposite way. Similarly, the weight of CDLSHOOTINGSTAR started with negative value, then during the middle of the testing period it became positive, and at the end it went back to negative. On the other hand, the weights of CDLINNECK seemed to vary periodically across the time scale. These examples justify the use of time regularization as they show that there are temporal changes in market conditions to which the feature weights can adapt.

Last, the importance of each feature group is determined, which could result in the omission of certain features and a simpler model consequently. Table 4 contains ablation analysis of each technical analysis feature group in terms of predictive power. This shows that the Pattern

Table 4 Trading with selected feature groups

Feature group	Absolute profit
All	£64.9
Pattern recognition	£54.8
Momentum indicators	£47.4
Overlap studies	£37.8
Volume indicators	£27.1
Cycle indicators	£21.2
Price transform	£9.0
Volatility indicators	£-5.7

Matching indicators had the most predictive power during the experiments, with Momentum indicators coming close second. On the other hand, Volatility indicators were not very useful in predicting profitability, but all indicators together still gave a solid boost in performance.

6.1 Trading simulation

The experimental results reported above show the efficacy of our proposed modelling technique. However the above evaluation used a simplistic trading setting which does not correspond to the conditions an investor would face on the market. Now we seek to augment the evaluation to cope with real market conditions, by (1) maintaining a running budget to determine the amount invested each day, (2) disallowing short-selling and (3) including transaction costs.[8] Together these changes provide a more realistic evaluation of the trading profits and losses.

First, consider the trading amount, which previously was fixed to range between £-1 and £1. Here instead we allow the budget accumulate throughout the testing period, and scale all trades by the funds available each day.[9] This limits the downside during a run of poor performance, as investments become proportionally smaller, while also increasing the profits (and risk) after sustained successful trading. The results of trading with a cumulative balance are shown in Fig. 5. Even with two significant market crashes, the algorithm made a loss only temporarily. On average the algorithm made a profit of 101 % which is equivalent to a 6.53 % annual return, exceeding the baseline buy and hold strategy which returned 22 % profit overall and 1.8 % annually.

Secondly, although short selling is permitted on many financial markets, it is a controversial practice and is subject to several restrictions and costs. In brief, short selling involves borrowing a stock from a broker, which is immediately sold on the market. The transaction is closed later by repurchasing the stock to repay the debt to the broker. This strategy can be used to profit in a falling market, however, as it involves borrowing with an unlimited potential for loss, it is not a widely available service and typically incurs significant costs and collateral requirements. For these reasons we now consider evaluation where short-selling is disallowed. This allows for the algorithm to applied to a much wider range of markets and stocks. To support this change, we also allow stocks to be held in long positions, and

[8] Note that we allow for fractional stocks to be traded, which is also unrealistic, but would have only a negligible effect when trading with a sufficiently large budget.

[9] This evaluation method corresponds to the compounding training objective U' in (3). However for simplicity in this section we evaluate only the model trained without compounding, denoted `Multi task, tech anal, comp-time reg` in Sect. 6.

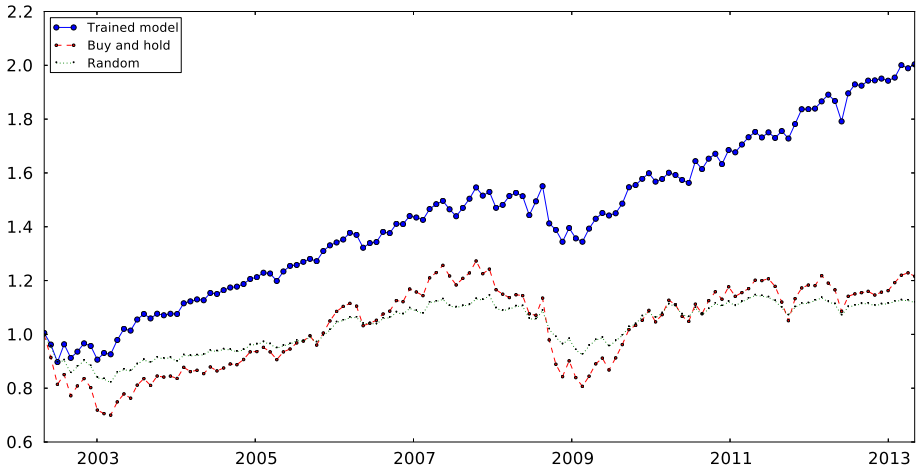

Fig. 5 Balance evolution, or cumulative profits, of various trading strategies

maintain daily cash and stock balances for each security. This contrasts with the evaluation in Sect. 6 where each trade was reversed on the following day, such that no ongoing positions were held. The revised evaluation process starts with a budget of £1 for each security, split evenly between stock and cash. Buy predictions $p(\mathbf{x}, \mathbf{w}) > 0$ are applied to the cash balance, proportionally to the prediction magnitude, which is then used to increase the investment in the stock. Sell predictions $p(\mathbf{x}, \mathbf{w}) < 0$ are applied proportionally to the stock balance, thus depleting the stock and increasing the cash balance. Note that as both balances are tracked separately and will often differ in value, and consequently the same magnitude prediction for buy versus sell may result in a different value trade. This trading strategy is denoted **trading position**. For comparison we also present two modifications to this trading strategy: first **fixed lot** trading which scales each trade, by a fixed constant, e.g., 1 %. This helps to limit extreme trading behavior where all or none of the budget is invested. The other modification is **rebalancing** which equalizes the value of stock and cash in each account before applying the trade. This restores the symmetry of buying and selling, and limits the exposure to large losses or gains. Note that **fixed lot** is orthogonal to **rebalancing**, and we evaluate using both techniques together.

The final change to our evaluation method is to include **transaction costs** at 0.6 % of the transaction value. The results of the trading strategy simulations are shown in Table 5, where each trade was executed using the closing price of each day. As expected transaction costs tend to erode the profits, however this was not the case with some trading strategies. In particular, with a fixed 1 % lot size, the algorithm still made a substantial profit. When combined with the rebalancing strategy the profits were even greater than positional, which is even more surprising considering the costs levied on the rebalancing transactions. What this suggests is that it is important to realize gains and losses quickly. With a positional trading strategy with 100 % lot size, early trades can have a large affect on the leverage of later trades. While rebalancing is also similarly affected by compounding balances, it is overall more conservative and maintains a more diversified portfolio. The algorithm may have predicted small changes in the relative price movement accurately, but these changes were not enough to offset the transaction costs, which were not included in the model's training objective. However, when the transaction costs were zero, then there was no drawback of

Table 5 Trading simulations with £1 starting balance, 2002–2013, 62 companies

Transaction cost (%)	Trading position	Lot size (%)	Prediction model	End balance
0	Positional	100	Trained model	£2.01
0	Positional	100	Buy-and-hold	£1.22
0	Positional	100	Random	£1.11
0	Positional	1	Trained model	£1.18
0	Rebalance	1	Trained model	£1.33
0.6	Positional	100	Trained model	£0.00
0.6	Positional	100	Buy-and-hold	£1.21
0.6	Positional	100	Random	£0.00
0.6	Positional	1	Trained model	£1.12
0.6	Rebalance	1	Trained model	£1.20

predicting small price changes and thus a 100 % lot strategy produced more profit than 1 % lot. Note that the algorithm came very close to the buy and hold strategy in the transaction cost case even though it suffered significant transaction costs compared to the practically no transaction costs of the former. In future work, we plan to extend the model objective to allow for transaction costs.

6.1.1 Performance metrics

Modern Portfolio Theory gives various measures to evaluate the performance of a trading strategy. According to the Capital Asset Pricing Model (Sharpe 1964) in order to evaluate whether an investment is worth the capital, the investor must be compensated for the time value of his capital and for the risk of losing the investment, also known as risk premium.

Jensen's alpha (Jensen 1968), α_a, can be interpreted as how much the fund strategy outperforms the baseline investment strategy risk adjusted, $\alpha_a = r_a - (r_f + \beta_a(r_m - r_f))$, where r_a is the returns of portfolio or own strategy, r_f is the risk-free returns, 3 month UK treasury bond yields in sterling,[10] r_m is the market or baseline returns, always buy or buy and hold. A value above 0 means that the algorithm was able to beat the market in the long term without taking excess risk. Beta, β_a, measures how volatile or risky the strategy is compared to the baseline, $\beta_a = \frac{\sigma(r_a, r_m)}{\sigma^2(r_m)}$, where σ is the covariance function. The Sharpe ratio measures excess return adjusted by risk (Sharpe 1998), $S = \frac{r_a - r_f}{\sigma(r_a)}$.

The results are summarized in Table 6. An annual rate of 4.3 % was calculated for alpha, meaning that the algorithm generated risk-free excess profits in the long term, putting it in the 95 % percentile of funds as measured by a t-distribution of 115 fund performances over 20 years (Jensen 1968). Beta was measured on the monthly time series of the buy and hold and the own portfolio returns, where 1 denotes the market risk. An annualized value of 0.54 means the algorithm returns were considerably less risky than the market returns. The Sharpe ratio of 1.52 was computed based on the monthly standard deviation of returns, noting that 1 is considered good, 2 very good and 3 excellent (Lo 2002). In summary, all three measures confirm the presence of risk adjusted profits from our algorithm's trading predictions.

[10] We used the monthly average 3-month sterling treasury bill discount rate data for 2002–2013 as reported by the Bank of England, http://www.bankofengland.co.uk/.

Table 6 Annualized performance metrics of own strategy compared to baseline buy and hold, risk adjusted by 3 month sterling UK treasury bills discount rate, 2002–2013

Metric	Value
Alpha	4.3 %
Beta	0.54
Sharpe	1.52

7 Conclusions

This paper has demonstrated that stock market price movements are predictable, and patterns of market movements can be exploited to realize excess profits over passive trading strategies. We developed a model of daily stock trading of several stocks, and a means of training to directly maximize trading profit. This results in consistent risk-free profit when evaluated on more than a decade of market data for several UK companies, beating strong baselines including buy-and-hold and linear regression. Beyond individual stock modelling, we presented a multi-task learning approach to account for temporal variations in market conditions as well as relationships between companies, both demonstrating further improvements in trading profit. Technical analysis indicators, in particular from the pattern matching and momentum family, were found to have better predicting power than plain historical returns calculated on a window of adjacent trading days. Finally, we demonstrated in realistic trading scenarios that the algorithm was capable of producing a profit when including transaction costs.

Appendix 1: List of company symbols

The following companies were used in the experiments reported in the paper. Shown below are the stock symbols on the London Stock Exchange.

```
CPI REX BG AGK BA AZN AAL AMEC CNA LGEN BARC BP REL STAN CRH
GSK PRU LAND MGGT RR DGE SAB NG BLND RIO WTB SMIN RB RDSB
LLOY SGE SHP NXT IMI BLT MKS MRW RBS WEIR WOS HMSO SRP IMT
GKN ADN SVT BSY VOD TSCO SDR UU GFS HSBA RSA OML TATE SN AV
KGF PSON BATS ULVR
```

Appendix 2: Technical indicators

Overlap studies

```
BBANDS              Bollinger Bands
DEMA                Double Exponential Moving Average
EMA                 Exponential Moving Average
HT_TRENDLINE        Hilbert Transform - Instantaneous
                                Trendline
KAMA                Kaufman Adaptive Moving Average
MA                  Moving average
MAMA                MESA Adaptive Moving Average
MAVP                Moving average with variable period
MIDPOINT            MidPoint over period
```

MIDPRICE	Midpoint Price over period
SAR	Parabolic SAR
SAREXT	Parabolic SAR - Extended
SMA	Simple Moving Average
T3	Triple Exponential Moving Average (T3)
TEMA	Triple Exponential Moving Average
TRIMA	Triangular Moving Average
WMA	Weighted Moving Average

Momentum indicators

ADX	Average Directional Movement Index
ADXR	Average Directional Movement Index Rating
APO	Absolute Price Oscillator
AROON	Aroon
AROONOSC	Aroon Oscillator
BOP	Balance Of Power
CCI	Commodity Channel Index
CMO	Chande Momentum Oscillator
DX	Directional Movement Index
MACD	Moving Average Convergence/Divergence
MACDEXT	MACD with controllable MA type
MFI	Money Flow Index
MINUS_DI	Minus Directional Indicator
MINUS_DM	Minus Directional Movement
MOM	Momentum
PLUS_DI	Plus Directional Indicator
PLUS_DM	Plus Directional Movement
PPO	Percentage Price Oscillator
ROC	Rate of change : ((price/prevPrice)-1) *100
ROCP	Rate of change Percentage: (price-prevPrice)/prevPrice
ROCR	Rate of change ratio: (price/prevPrice)
RSI	Relative Strength Index
STOCH	Stochastic
STOCHF	Stochastic Fast
STOCHRSI	Stochastic Relative Strength Index
TRIX	1-day Rate-Of-Change (ROC) of a Triple Smooth EMA
ULTOSC	Ultimate Oscillator
WILLR	Williams' %R

Volume indicators

AD	Chaikin A/D Line
ADOSC	Chaikin A/D Oscillator
OBV	On Balance Volume

Cycle indicators

HT_DCPERIOD	Hilbert Transform - Dominant Cycle Period
HT_DCPHASE	Hilbert Transform - Dominant Cycle Phase
HT_PHASOR	Hilbert Transform - Phasor Components
HT_SINE	Hilbert Transform - Sine Wave
HT_TRENDMODE	Hilbert Transform - Trend vs Cycle Mode

Price transform

AVGPRICE	Average Price
MEDPRICE	Median Price
TYPPRICE	Typical Price
WCLPRICE	Weighted Close Price

Volatility indicators

ATR	Average True Range
NATR	Normalized Average True Range
TRANGE	True Range

Pattern recognition

CDL2CROWS	Two Crows
CDL3BLACKCROWS	Three Black Crows
CDL3INSIDE	Three Inside Up/Down
CDL3LINESTRIKE	Three-Line Strike
CDL3OUTSIDE	Three Outside Up/Down
CDL3STARSINSOUTH	Three Stars In The South
CDL3WHITESOLDIERS	Three Advancing White Soldiers
CDLABANDONEDBABY	Abandoned Baby
CDLADVANCEBLOCK	Advance Block
CDLBELTHOLD	Belt-hold
CDLBREAKAWAY	Breakaway
CDLCLOSINGMARUBOZU	Closing Marubozu
CDLCONCEALBABYSWALL	Concealing Baby Swallow
CDLCOUNTERATTACK	Counterattack
CDLDARKCLOUDCOVER	Dark Cloud Cover
CDLDOJI	Doji
CDLDOJISTAR	Doji Star
CDLDRAGONFLYDOJI	Dragonfly Doji
CDLENGULFING	Engulfing Pattern
CDLEVENINGDOJISTAR	Evening Doji Star
CDLEVENINGSTAR	Evening Star
CDLGAPSIDESIDEWHITE	Up/Down-gap side-by-side white lines
CDLGRAVESTONEDOJI	Gravestone Doji
CDLHAMMER	Hammer
CDLHANGINGMAN	Hanging Man
CDLHARAMI	Harami Pattern

CDLHARAMICROSS	Harami Cross Pattern
CDLHIGHWAVE	High-Wave Candle
CDLHIKKAKE	Hikkake Pattern
CDLHIKKAKEMOD	Modified Hikkake Pattern
CDLHOMINGPIGEON	Homing Pigeon
CDLIDENTICAL3CROWS	Identical Three Crows
CDLINNECK	In-Neck Pattern
CDLINVERTEDHAMMER	Inverted Hammer
CDLKICKING	Kicking
CDLKICKINGBYLENGTH	Kicking - bull/bear determined by the longer marubozu
CDLLADDERBOTTOM	Ladder Bottom
CDLLONGLEGGEDDOJI	Long Legged Doji
CDLLONGLINE	Long Line Candle
CDLMARUBOZU	Marubozu
CDLMATCHINGLOW	Matching Low
CDLMATHOLD	Mat Hold
CDLMORNINGDOJISTAR	Morning Doji Star
CDLMORNINGSTAR	Morning Star
CDLONNECK	On-Neck Pattern
CDLPIERCING	Piercing Pattern
CDLRICKSHAWMAN	Rickshaw Man
CDLRISEFALL3METHODS	Rising/Falling Three Methods
CDLSEPARATINGLINES	Separating Lines
CDLSHOOTINGSTAR	Shooting Star
CDLSHORTLINE	Short Line Candle
CDLSPINNINGTOP	Spinning Top
CDLSTALLEDPATTERN	Stalled Pattern
CDLSTICKSANDWICH	Stick Sandwich
CDLTAKURI	Takuri (Dragonfly Doji with very long lower shadow)
CDLTASUKIGAP	Tasuki Gap
CDLTHRUSTING	Thrusting Pattern
CDLTRISTAR	Tristar Pattern
CDLUNIQUE3RIVER	Unique 3 River
CDLUPSIDEGAP2CROWS	Upside Gap Two Crows
CDLXSIDEGAP3METHODS	Upside/Downside Gap Three Methods

References

Bengio, Y. (1997). Training a neural network with a financial criterion rather than a prediction criterion. In A. S. Weigend, Y. Abu-Mostafa, & A.-Paul N. Refenes (Eds.), *Decision technologies for financial engineering:Proceedings of the fourth international conference on neural networks in the capital markets (NNCM'96)* (pp. 36–48). London: World Scientific Publishing.

Bondt, W. F., & Thaler, R. (1985). Does the stock market overreact? *The Journal of Finance*, *40*(3), 793–805.

Byrd, R. H., Lu, P., Nocedal, J., & Zhu, C. (1995). A limited memory algorithm for bound constrained optimization. *SIAM Journal on Scientific Computing*, *16*(5), 1190–1208.

Caruana, R. (1997). Multitask learning. *Machine Learning*, *28*(1), 41–75.

Cha, S. M., & Chan, L. (2000). Trading signal prediction. In *International conference on neural information processing-ICONIP*. Citeseer, pp. 842–846.
Evgeniou, T., & Pontil, M. (2004). Regularized multi-task learning. In *Proceedings of the tenth ACM SIGKDD international conference on knowledge discovery and data mining*. ACM, pp. 109–117.
Ghosn, J., & Bengio, Y. (1997). Multi-task learning for stock selection. *Advances in Neural Information Processing Systems, 10*, 946–952.
Haugen, R. A., & Lakonishok, J. (1987). *The incredible January effect: The stock market's unsolved mystery*. Homewood, IL: Dow Jones-Irwin.
Hoerl, A. E., & Kennard, R. W. (1970). Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, *12*(1), 55–67.
Jensen, M. C. (1968). The performance of mutual funds in the period 1945–1964. *The Journal of Finance*, *23*(2), 389–416.
Kahneman, D., & Tversky, A. (1979). Prospect theory: An analysis of decision under risk. *Econometrica: Journal of the Econometric Society*, 263–291.
Kim, S. D., Lee, J. W., Lee, J., & Chae, J. (2002). A two-phase stock trading system using distributional differences. In A. Hameurlain, R. Cicchetti, & R. Traunmüller (Eds.), *Database and Expert Systems Applications* (pp. 143–152). Berlin: Springer.
Lo, A. W. (2002). The statistics of sharpe ratios. *Financial Analysts Journal*, 36–52.
Lo, A. W., Mamaysky, H., & Wang, J. (2000). Foundations of technical analysis: Computational algorithms, statistical inference, and empirical implementation. *The Journal of Finance*, *55*(4), 1705–1770.
Magdon-Ismail, M., Nicholson, A., & Abu-Mostafa, Y. S. (1998). Financial markets: Very noisy information processing. *Proceedings of the IEEE*, *86*(11), 2184–2195.
Malkiel, B. G. (2003). The efficient market hypothesis and its critics. *The Journal of Economic Perspectives*, *17*(1), 59–82.
Markowitz, H. (1952). The Portfolio selection. *Journal of Finance*, *7*(1), 77–91.
Neely, C., Weller, P., & Dittmar, R. (1997). *Is technical analysis in the foreign exchange market profitable? A genetic programming approach*. Cambridge: Cambridge University Press.
Oliphant, T. E. (2007). Python for scientific computing. *Computing in Science & Engineering*, *9*(3), 10–20.
Rogers, S., & Girolami, M. (2011). *A first course in machine learning*. Boca Raton, FL: CRC Press.
Sharpe, W. F. (1964). Capital asset prices: A theory of market eqilibrium under conditions of risk. *The Journal of Finance*, *19*(3), 425–442.
Sharpe, W. F. (1998). *The sharpe ratio. Streetwise—The best of the Journal of Portfolio Management*. Princeton, NJ: Princeton University Press.
Shiller, R. J., Fischer, S., & Friedman, B. M. (1984). Stock prices and social dynamics. *Brookings Papers on Economic Activity*, *2*, 457–510.
Tibshirani, R., Saunders, M., Rosset, S., Zhu, J., & Knight, K. (2005). Sparsity and smoothness via the fused lasso. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, *67*(1), 91–108.
Tversky, A., & Kahneman, D. (1974). Judgment under uncertainty: Heuristics and biases. *Science*, *185*(4157), 1124–1131.
Weil, P. (1989). The equity premium puzzle and the risk-free rate puzzle. *Journal of Monetary Economics*, *24*(3), 401–421.