University of Washington

Computational Finance & Risk Management

Department of Applied Mathematics

Creation of a Systematic Trading Strategy

Tap into the Pulse of the Markets

Author:
Nick Kirk
nkirk01@uw.edu
nk@mintegration.eu

Supervisors:
Brian Peterson
Guy Yollin

August 22, 2015

"I have observed that the selection of food is often influenced by the mood of the person ordering."

Lieutenant Commander Data

1 Introduction

Combining sentiment data with technical indicators, and using a machine learning classification technique named Support Vector Machines (SVM), a systematic trading strategy is created.

The classifier forecasts with reasonable accuracy the future direction of the daily closing prices for a subset of S&P500 stocks. Based on these predictions, indicators are created, signals generated and trading rules make path-dependent actionable decisions to generate orders.

All research and development was implemented in the **R** software environment for statistical computing and graphics [35]. The following R packages were used; quanstrat [21], blotter [20], PerformanceAnalytics [19], TTR [37], kernlab [13], caret [15], xts [28], quantmod [29], doParallel [38] and doMC [1].

End-of-Day (EOD) U.S. stock prices are sourced from QuoteMedia [26] through Quandl's [25] premium subscription.

The chosen source of sentiment is from StockTwits [31] message posts that have been aggregated and scored by PsychSignal [24]. This data too can be obtained through Quandl.

1.1 Financial Crowdsourcing Sites and Sentiments

Crowdsourcing sites like StockTwits [31], Estimize [8], SumZero [32] and Harvest Exchange [10] aggregate stock commentary, earnings estimates and investment research. Then there are data analytics services like PsychSignal [24], Dataminr [6], Gnip [9], Social Market Analytics [2], Knowsis [14] and Market Prophit [23]. They scrub the raw crowdsourced data and extract actionable signals from the noise. See [7] for a comprehensive guide in to using social media as an investment tool.

StockTwits - "Tap into the Pulse of the Markets". Founded in 2008, StockTwits is a financial communications platform for the financial and investing community. StockTwits created the $TICKER tag (i.e. "cashtags") to enable and organize "streams" of information around stocks and markets across the web and social media.

Today, more than 300,000 investors, market professionals and public companies share information and ideas about the market and individual stocks using StockTwits.

Psychologists use time to differentiate between moods and emotions. Emotions occur within brief defined moments of time and are regarded as being unique to individual persons. When emotions extend over time they become moods. A crowd mood can be formed by aggregating many of these individual emotions.

For the crowd to be wise it has to satisfy *four* specific conditions, but once those conditions are met, its judgment is likely to be accurate [33];

- Diversity of opinion: Each person should have private information even if it's just an eccentric interpretation of the known facts.
- Independence: People's opinions aren't determined by the opinions of those around them.
- Decentralization: People are able to specialize and draw on local knowledge.
- Aggregation: Some mechanism exists for turning private judgments into a collective decision.

PsychSignal create a measure of crowd mood by aggregating and scoring emotions from short form, time stamped messages, originating from microblogging sites including Twitter and StockTwits. Once the psychological expressions are identified and aggregated, their proprietary Natural Language Processing (NLP) engine continually scores each emotion or attitude (including anger, sadness and love) to determine the degree of *bullishness* and *bearishness* present.

PsychSignal cover 10,000+ symbols covering a wide variety of financial market products including Stocks, ETFs, Futures, Currencies and even Bitcoin. A majority of the symbols are derived from the StockTwits "cashtags" people use to identify the symbols they are talking about.

I use PsychSignal's sentiment data with StockTwits as the chosen source. The following fields are inputs to my predictive model; *bullish_ intensity*, *bearish_ intensity*, *bull_ scored_ messages*, *bear_ scored_ messages* and *total_ scanned_ messages.*

Field	**Description**
source	The chosen source or aggregated sources responsible for the sentiments data (e.g. StockTwits, Twitter)
symbol	The stock symbol or sector symbol for which the sentiments data refer to
timestamp (UTC)	Date and time of the analysed data
bullish_intensity	PsychSignal's algorithms score each message's language for the strength of bullishness present on a 0-4 scale. 0 indicates no bullish sentiment measured, 4 indicates strongest bullish sentiment measured. 4 is rare
bearish_intensity	PsychSignal's algorithms score each message for the strength of bearishness present in the message on a 0-4 scale. 0 indicates no bearish sentiment measured, 4 indicates strongest bearish sentiment measured. 4 is rare
bull_minus_bear	This indicator simply subtracts bearish_intensity from bullish_intensity [BULL - BEAR] to provide an immediate net score
bull_scored_messages	The total count of bullish sentiment messages scored by the algorithm
bear_scored_messages	The total count of bearish sentiment messages scored by the algorithm
bull_bear_msg_ratio	Ratio between bull_scored_messages and bear_scored_messages
total_scanned_messages	The number of messages coming through PsychSignal's source data feeds and attributable to a symbol regardless of whether PsychSignal's sentiment engine can score them for bullish or bearish intensity

Table 1: PsychSignal's Sentiment Data Per Security

An academic research paper that is related to my strategy is authored by Plakandaras et al [22]. They used StockTwits data as a possible predictor for the future evolution of exchange rates. They trained several Machine Learning models and found that they outperformed both a Random Walk (RW) model and standard Econometric methods and hence argued that sentiment data helps to shape market expectations and drives exchange rates.

Interestingly, they detect a significant lag between when the StockTwits messages were posted and the time of the directional movement of the exchange rates. This is something I explore by asking the question; "Do *today's* public Psychological stance generate any Signals anticipating *tomorrow's* market performance?" (that's where the name "PsychSignal" came from). To help answer this question I evaluate potential delays in the transmission channel of the sentiment to the stock prices with up to 8 lags.

If you know what the public's mood is at any given point in time, then this could be utilized in shaping profitable investment portfolios. And for knowing the optimal number of chocolate sundaes to order !

1.2 The SVM Model

The classifier is chosen to be a non-linear support vector machine (SVM) due to its simplicity and effectiveness [30], [17]

There are alternative methods that might do as good a job as a SVM, but the simplicity of the mathematical functions, and the theory that frames the training of the model as a convex optimization problem make SVMs a preferred option [3]. An important feature of convex optimization problems is a guarantee that there is a single optimal model to fit the data.

The SVM is trained to learn correlations between the features of a stock and the class it belongs to (positive or negative returns).

There are two alternative paradigms; linear versus non-linear [11].

The linear method begins by trying to find a hyperplane that separates and makes the biggest gap or margin between the two classes in the feature space (known as a Maximal Margin Classifier). If it cannot, we get creative in two ways by creating a Support Vector Classifier:

- We can soften what we mean by "separates" by introducing a regularization parameter C, also known as a hyper-parameter.
- We can enrich and enlarge the feature space by including transformations so that separation is possible. This results in non-linear decision boundaries in the original space. The non-linear support vector classifier in the enlarged space solves the problem in the lower-dimensional space.

However, there is a more elegant and controlled way to introduce nonlinearities in support vector classifiers - through the use of **kernels**. If we can compute inner-products between observations, we can fit a **SVM** classifier. Some special kernel functions can do this for us.

I decide to use a **Gaussian kernel** (the Gaussian kernel is an example of radial basis function (RBF) kernel).

$$K(\mathbf{x}, \mathbf{x}_i) = e^{-\frac{\|\mathbf{x}-\mathbf{x}_t\|^2}{2\sigma^2}} \tag{1}$$

Alternatively, it could be written as:

$$K(\mathbf{x}, \mathbf{x}_i) = e^{-\gamma\|\mathbf{x}-\mathbf{x}_t\|^2/M} \tag{2}$$

2 Strategy Constraints, Benchmarks, and Objectives

Without a set of clearly defined goals, there is the possibility of accepting a strategy or backtest that is incompatible with business constraints. It can also lead to adjusting your goal to try to follow the backtest, which can culminate in all sorts of bad decision making, and can increase the probability of erroneously accepting an over-fitted backtest [18].

My business constraints include the following:

- Securities that the software systems can trade are limited to stocks
- Infrastructure and systems can currently support only EOD data
- Total amount of available capital is $100K
- Limited leverage with the broker

As the available capital is limited to only $100K and the Infrastructure and systems can handle only EOD stocks, the assets I select to trade are liquid assets from the S&P500 index.

The choice of benchmark will be the S&P500 index.

The Business Objectives are outlined here and are mostly driven by the constraints:

- Drawdowns must be shallow (e.g. less than 20 percent) and short in duration (e.g. less than a few weeks)
- Orders will consist of directional trades (long or short only) and dollar-neutral trades (hedged or pairs) will not be considered
- A consistency of returns
- An annualized Sharpe Ratio (SR) higher than 1.5. Information Ratio (annualized return over annualized risk) is the measure to use when you want to assess a long-only strategy [5]. The SR is actually a special case of the IR, suitable when we have a dollar-neutral strategy, so that the benchmark to use is always the risk-free rate. In practice, the SR is used even when trading a directional strategy, simply because it facilitates comparison across different strategies. During the last 5 years, the risk-free rate has been very near zero and so the SR is almost equal to the IR anyhow.

3 Data

Both the EOD U.S. stock prices from the S&P500 index and PsychSignal's sentiment data was merged and approximately 4.5 years of data was finally used.

Duration
2011-02-01 to 2015-06-08

The number of *bull_ scored_ messages* and *bear_ scored_ messages* sentiment posts per stock ranged from a minimum of 80(ALLE) to a maximum of 547100(AAPL). 27 stocks with the highest number of these sentiment posts were selected and the rest dropped from the dataset.

Stock	bull_scored_messages	bear_scored_messages	total_scanned_messages
AAPL	305.48	187.10	1230.30
FB	126.06	70.40	468.98
NFLX	36.12	27.44	146.87
AMZN	26.77	17.60	109.89
GILD	28.26	14.00	102.41
BAC	23.60	11.65	80.13
PCLN	20.42	13.48	75.98
YHOO	19.16	8.20	73.48
GMCR	12.06	7.42	42.36
FSLR	11.52	6.23	40.86
MU	11.79	5.47	39.69
MSFT	10.73	4.61	49.95
GS	9.57	5.27	36.62
F	10.10	4.34	38.03
KORS	9.26	4.61	30.76
CMG	7.85	5.25	30.76
IBM	6.81	3.85	27.10
RIG	6.69	3.82	23.60
JPM	6.70	3.63	29.54
C	6.24	3.16	25.89
SBUX	6.42	2.75	22.23
CRM	5.23	3.80	21.22
EBAY	6.04	2.78	23.55
INTC	5.96	2.66	24.20
BBY	5.11	3.09	17.48
AA	5.46	2.16	18.82
CAT	4.60	2.94	17.21

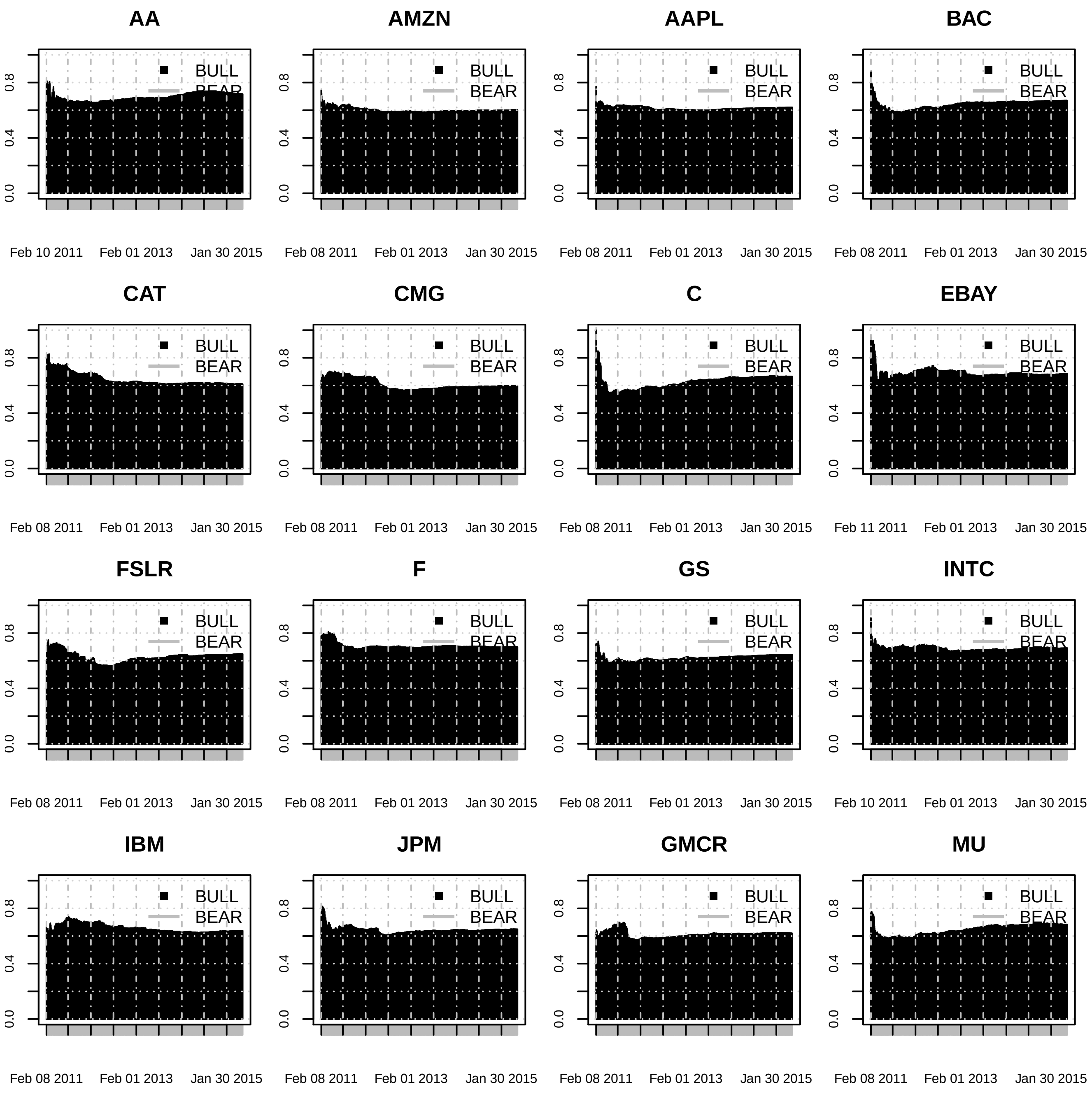
Table 2: Daily Averages for the 27 Stocks with the Highest Number of Sentiment Posts

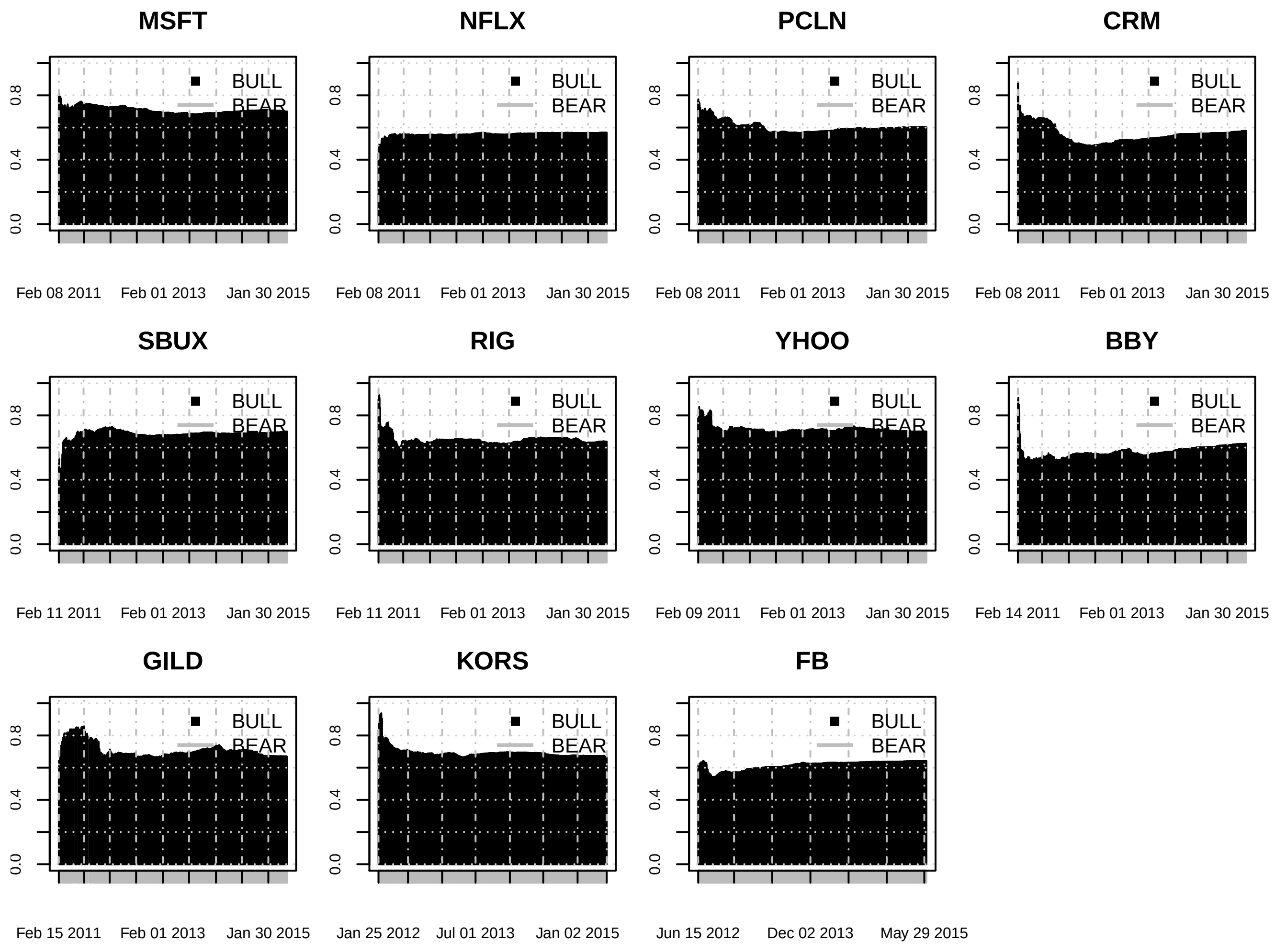
The ratio between the two classes (positive and negative returns respectively) is almost equal for all stocks. Therefore, classification accuracy is used as the performance measure that will be optimized during training.

Stock	Positive	Negative
AA	49.70	50.30
AMZN	51.20	48.80
AAPL	52.10	47.90
BAC	49.60	50.40
CAT	49.50	50.50
CMG	53.10	46.90
C	50.50	49.50
EBAY	50.90	49.10
FSLR	49.00	51.00
F	48.90	51.10
GS	53.20	46.80
INTC	51.20	48.80
IBM	50.10	49.90
JPM	52.10	47.90
GMCR	50.30	49.70
MU	52.50	47.50
MSFT	49.30	50.70
NFLX	50.10	49.90
PCLN	51.40	48.60
CRM	51.50	48.50
SBUX	53.80	46.20
RIG	47.30	52.70
YHOO	51.30	48.70
BBY	50.40	49.60
GILD	56.10	43.90
KORS	51.10	48.90
FB	51.50	48.50

Table 3: Class Ratios

Plotted below are the ratios of the *bull_ scored_ messages* and *bear_ scored_ messages* for each stock. Bullish sentiment is noticeably higher than bearish sentiment during the period 2011-02-01 to 2015-06-08.




MSFT
NFLX
PCLN
CRM
SBUX
RIG
YHOO
BBY
GILD
KORS
FB
BULL
BEAR
0.0
0.4
0.8
Feb 08 2011
Feb 11 2011
Feb 09 2011
Feb 14 2011
Feb 15 2011
Feb 01 2013
Jan 30 2015
Jan 25 2012
Jul 01 2013
Jan 02 2015
Jun 15 2012
Dec 02 2013
May 29 2015

3.1 Input Variables

Like Plakandaras et al [22], I assess multiple independent variables when tuning and training the model. *Input Variable Sets* are formulated that comprise different combinations of these variables. In other words, combinations of the regressors are created to identify the Input Variable Set with the highest forecasting accuracy.

One must be careful to not fall into the trap of data mining bias (i.e. brute force searching of the parameter space). Considering a broader range of candidate signals or factors, and applying a feature selection algorithm such as the QPFS method [27] could be something to consider in the future.

The input variable sets used are shown in Table 4. Regressors include simple autoregressive (AR) models (past returns), PsychSignal's sentiment data and several technical indicators.
The Relative Strength Index (RSI) is a momentum oscillator that measures the speed and change of price movements. Moving averages are used to identify current trends. An Exponential moving average (EMA) calculates an exponentially-weighted mean, giving more weight to recent observations. Several look-back periods are chosen for the technical indicators; RSI with 9 and 14 days and EMA with 5 and 13 days.

Input Set 1	AR(1)
Input Set 2	AR(1), AR(2)
Input Set 3	EMA5, EMA13
Input Set 4	RSI9, RSI14
Input Set 5	EMA5, RSI9, EMA13, RSI14
Input Set 6	bullish_intensity, bearish_intensity
Input Set 7	bullish_intensity, bearish_intensity, total_scanned_messages
Input Set 8	bull_scored_messages, bear_scored_messages
Input Set 9	bull_scored_messages, bear_scored_messages, total_scanned_messages
Input Set 10	bullish_intensity, bearish_intensity, bull_scored_messages, bear_scored_messages
Input Set 11	bullish_intensity, bearish_intensity, bull_scored_messages, bear_scored_messages, total_scanned_messages

Table 4: Input Variable Sets

4 Hypotheses

Testing that the strategy meets its objective (i.e. to make money) is one thing but testing that the model used in the strategy has predictive power is another.

The main hypothesis is that the SVM model is good at predicting the future direction of stock prices.

In order to validate this hypothesis, a model is trained and then validated by checking how accurately the model predicted the future direction of stock prices.

The original data is split into two. One of them is used for **training** and the other one is used for **testing**. The *training* dataset is further split into sets for ***training*** and ***validation*** using a technique called ***cross-validation***.

training	testing
2011-02-01 to 2013-12-24	2013-12-25 to 2015-06-08

Using a 5-fold cross-validation resampling method, the SVM hyper-parameters C and γ are tuned. The model with the highest accuracy is selected as the best.
Other methods also exist. Huerta et al [12] chose to implement the selection of the hyper-parameters by a type of *Reinforcement Learning* [34].

Repeated "looks" at the test dataset (i.e. data snooping) can lead to over-fitting hence resampling with cross-validation by using the training samples helps detect and avoid over-fitting.

For models using Gaussian RBF kernels, it turns out that there is an analytical method for directly estimating the optimal values of the width hyper-parameter γ [4]. It is shown to lie in between the 0.1 and 0.9 quantile of $||x - x'||^2$. Generally the mid-point between these two numbers provide a good estimate for this hyper-parameter. By default, the train function in the caret package uses the *sigest* function from the kernlab package to initialize this parameter. In doing so, this leaves only the Cost parameter C for tuning.
A grid search was performed and the Cost was varied over 10 values; 0.25, 0.50, 1.00, 2.00, 4.00, 8.00, 16.00, 32.00, 64.00 and 128.00.

For an extremely good treatment on the predictive modelling process, in particular data pre-processing, data splitting and the foundations of model tuning, see [11] and [16].

4.1 K-Fold Cross-Validation Algorithm

I randomly split the training data into K distinct blocks of roughly equal size.

1. I leave out the first block of data and fit a model.
2. This model is used to predict the held-out block (aka the validation set)
3. I continue this process until I've predicted all K held-out blocks

The final performance is based on the hold-out predictions by taking the average.

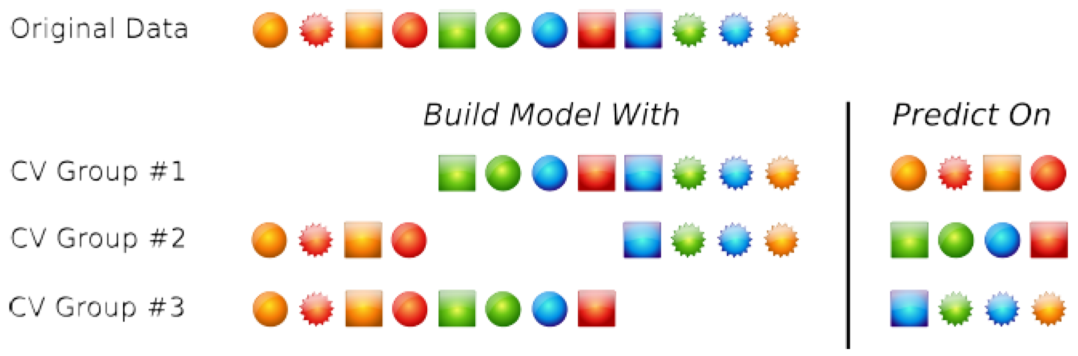


Figure 1: K-Fold Cross-Validation

4.2 Model Accuracy

A 5-fold cross-validation resampling method is implemented per stock, for each Input Set, and at each lag (0-8), where a lag of 0 defines the returns *today* and a lag of 8 being the returns 8 trading days in the *future*.

The best (most accurate) model at each lag from the *in-sample training* is used for the *out-of-sample testing*.

To check the accuracy of the model in the out-of-sample testing, a **Confusion Matrix** is used. This is a cross-tabulation of the true versus the predicted values. Correct positions fall on the diagonal. The off-diagonal matrix cells indicate the cases where the predicted value differs from the actual one.
The overall accuracy rate is given by:

$$\frac{\sum(\text{main diagonal})}{\sum(\text{entire table})} \tag{3}$$

The results from the in-sample training and out-of-sample testing are shown in Table 5.

There is a significant level of accuracy for all stocks at a lag of 0. However, this model cannot be used in my current trading strategy without incurring significant look-ahead bias because the returns are generated at the same time as when the inputs are calculated. It is however very interesting as I plan to look at intra-day sentiment data in the forthcoming weeks.

Not including the lag of 0, there is no lag more dominant than the rest. The Input Sets using Sentiment data occur many more times that those with technical indicators or AR models. Accuracies are mostly positive but not too much higher than 50%. Any additional *edge* is a good one though.

AA	Lag	0	1	2	3	4	5	6	7	8
	InputSet	4	3	4	8	4	11	4	10	6
	Sigma	4.75	11.906	3.771	38.541	5.691	1.179	3.163	1.058	3.045
	Cost	32	64	2	0.5	0.25	32	0.25	128	16
	Accuracy (Training)	0.753	0.56	0.558	0.584	0.558	0.536	0.554	0.571	0.577
	Kappa (Training)	0.506	0.118	0.113	0.169	0.111	0.065	0.098	0.139	0.147
	AccuracySD (Training)	0.043	0.026	0.043	0.08	0.038	0.041	0.066	0.039	0.088
	KappaSD (Training)	0.086	0.053	0.087	0.161	0.079	0.086	0.131	0.081	0.178
	Accuracy (Test)	0.693	0.514	0.532	0.564	0.464	0.464	0.504	0.493	0.493
	Kappa (Test)	0.386	0.005	0.066	0.063	-0.067	-0.073	0.022	-0.015	-0.071
AMZN	Lag	0	1	2	3	4	5	6	7	8
	InputSet	4	10	9	8	5	4	4	8	4
	Sigma	3.855	0.631	10.081	13.429	0.747	3.855	3.855	13.429	3.855
	Cost	32	16	64	4	0.5	4	0.5	1	0.25
	Accuracy (Training)	0.752	0.554	0.547	0.569	0.535	0.545	0.538	0.553	0.533
	Kappa (Training)	0.504	0.109	0.094	0.137	0.069	0.091	0.076	0.105	0.066
	AccuracySD (Training)	0.048	0.03	0.033	0.061	0.027	0.042	0.058	0.025	0.031
	KappaSD (Training)	0.097	0.061	0.065	0.122	0.054	0.082	0.116	0.05	0.061
	Accuracy (Test)	0.729	0.504	0.514	0.579	0.489	0.511	0.496	0.55	0.55
	Kappa (Test)	0.456	0.005	-0.027	0.109	-0.028	0.025	-0.016	0.04	0.093
AAPL	Lag	0	1	2	3	4	5	6	7	8
	InputSet	4	11	2	6	3	1	6	10	5
	Sigma	2.221	0.352	0.188	1.311	6.46	0.25	1.31	0.396	0.609
	Cost	2	1	0.5	4	128	1	128	1	32
	Accuracy (Training)	0.713	0.54	0.545	0.583	0.52	0.519	0.535	0.547	0.533
	Kappa (Training)	0.426	0.079	0.083	0.166	0.04	0.037	0.069	0.093	0.065
	AccuracySD (Training)	0.036	0.047	0.037	0.035	0.038	0.04	0.035	0.047	0.027
	KappaSD (Training)	0.072	0.094	0.077	0.07	0.077	0.078	0.07	0.093	0.054
	Accuracy (Test)	0.743	0.5	0.536	0.543	0.5	0.511	0.543	0.482	0.493
	Kappa (Test)	0.481	-0.025	0.029	0.028	-0.059	0.016	0.03	-0.045	0.035
BAC	Lag	0	1	2	3	4	5	6	7	8
	InputSet	4	3	9	8	9	4	8	5	6
	Sigma	4.087	17.323	2.716	3.35	2.716	4.087	3.35	0.539	1.445
	Cost	0.25	128	1	4	32	128	2	1	0.25
	Accuracy (Training)	0.715	0.538	0.563	0.565	0.585	0.554	0.561	0.558	0.563
	Kappa (Training)	0.429	0.071	0.123	0.129	0.167	0.109	0.12	0.114	0.125
	AccuracySD (Training)	0.042	0.05	0.028	0.035	0.047	0.043	0.031	0.035	0.076
	KappaSD (Training)	0.085	0.098	0.056	0.069	0.093	0.086	0.063	0.072	0.154
	Accuracy (Test)	0.714	0.5	0.536	0.525	0.518	0.482	0.593	0.482	0.571
	Kappa (Test)	0.427	-0.009	0.005	-0.009	-0.023	-0.037	0.115	-0.023	0.091
CAT	Lag	0	1	2	3	4	5	6	7	8
	InputSet	4	5	4	11	7	4	4	4	3
	Sigma	5.385	0.991	5.385	0.421	0.714	5.385	5.385	5.385	20.683
	Cost	128	4	1	1	1	1	2	2	64

	Accuracy (Training)	0.759	0.538	0.544	0.553	0.565	0.553	0.574	0.54	0.535
	Kappa (Training)	0.518	0.078	0.083	0.1	0.131	0.103	0.144	0.08	0.07
	AccuracySD (Training)	0.035	0.048	0.042	0.015	0.036	0.035	0.04	0.037	0.051
	KappaSD (Training)	0.071	0.097	0.088	0.032	0.073	0.07	0.078	0.075	0.1
	Accuracy (Test)	0.693	0.521	0.486	0.493	0.539	0.536	0.5	0.521	0.546
	Kappa (Test)	0.385	0.041	-0.029	-0.016	0.022	0.073	-0.009	0.038	0.091
CMG	Lag	0	1	2	3	4	5	6	7	8
	InputSet	5	4	7	5	10	11	8	1	8
	Sigma	0.515	3.493	0.82	0.515	0.603	0.519	12.845	0.25	12.849
	Cost	32	2	2	4	8	0.5	0.5	0.25	0.25
	Accuracy (Training)	0.717	0.538	0.545	0.569	0.535	0.553	0.547	0.519	0.569
	Kappa (Training)	0.431	0.07	0.077	0.139	0.068	0.098	0.074	0	0.122
	AccuracySD (Training)	0.046	0.03	0.044	0.022	0.034	0.024	0.04	0.002	0.028
	KappaSD (Training)	0.091	0.061	0.087	0.042	0.066	0.05	0.08	0	0.057
	Accuracy (Test)	0.646	0.493	0.582	0.493	0.421	0.521	0.571	0.5	0.543
	Kappa (Test)	0.297	-0.016	0.105	-0.014	-0.157	0.043	0.09	0	0.029
C	Lag	0	1	2	3	4	5	6	7	8
	InputSet	5	1	4	1	8	5	7	5	5
	Sigma	0.78	0.25	4.951	0.25	5.779	0.622	0.772	0.623	0.623
	Cost	128	0.25	8	0.25	16	8	0.5	128	128
	Accuracy (Training)	0.725	0.547	0.533	0.536	0.556	0.542	0.574	0.553	0.531
	Kappa (Training)	0.45	0.093	0.066	0.072	0.112	0.084	0.149	0.106	0.063
	AccuracySD (Training)	0.051	0.056	0.078	0.047	0.043	0.06	0.042	0.017	0.034
	KappaSD (Training)	0.102	0.111	0.155	0.094	0.086	0.12	0.084	0.034	0.068
	Accuracy (Test)	0.707	0.557	0.443	0.507	0.543	0.439	0.55	0.475	0.457
	Kappa (Test)	0.404	0.116	-0.123	0.017	0.021	-0.126	0.047	-0.046	-0.078
EBAY	Lag	0	1	2	3	4	5	6	7	8
	InputSet	4	5	1	4	5	2	2	9	6
	Sigma	4.088	1.055	0.25	4.854	1.053	0.188	0.188	6.582	2.13
	Cost	1	2	0.25	2	0.5	0.25	0.25	0.25	0.25
	Accuracy (Training)	0.718	0.555	0.539	0.542	0.555	0.54	0.551	0.535	0.572
	Kappa (Training)	0.435	0.11	0.079	0.085	0.107	0.079	0.103	0.069	0.145
	AccuracySD (Training)	0.048	0.055	0.031	0.043	0.038	0.049	0.039	0.037	0.035
	KappaSD (Training)	0.097	0.111	0.063	0.085	0.077	0.101	0.077	0.076	0.069
	Accuracy (Test)	0.736	0.532	0.486	0.482	0.504	0.514	0.514	0.536	0.493
	Kappa (Test)	0.47	0.069	-0.028	-0.036	0.006	0.029	0.029	0.012	-0.064
FSLR	Lag	0	1	2	3	4	5	6	7	8
	InputSet	4	9	2	11	7	4	4	6	2
	Sigma	4.58	10.544	0.188	0.77	1.045	4.073	4.074	2.284	0.188
	Cost	2	128	0.25	8	64	128	64	128	0.25
	Accuracy (Training)	0.698	0.531	0.544	0.544	0.576	0.558	0.555	0.504	0.533
	Kappa (Training)	0.396	0.06	0.084	0.087	0.152	0.116	0.109	0.008	0.065
	AccuracySD (Training)	0.054	0.06	0.051	0.041	0.028	0.034	0.033	0.036	0.042
	KappaSD (Training)	0.109	0.118	0.1	0.082	0.056	0.068	0.066	0.072	0.083
	Accuracy (Test)	0.721	0.564	0.486	0.536	0.532	0.511	0.457	0.55	0.536
	Kappa (Test)	0.443	0.077	-0.021	0.071	0.007	0.022	-0.086	0.045	0.071
F	Lag	0	1	2	3	4	5	6	7	8
	InputSet	4	1	7	4	4	8	9	4	6
	Sigma	3.144	0.25	0.872	3.19	3.144	2.781	1.945	3.144	2.063
	Cost	2	0.25	2	64	128	128	0.25	64	0.5
	Accuracy (Training)	0.754	0.54	0.556	0.542	0.553	0.555	0.526	0.57	0.531
	Kappa (Training)	0.508	0.08	0.111	0.083	0.104	0.108	0.051	0.139	0.058
	AccuracySD (Training)	0.014	0.055	0.046	0.025	0.047	0.043	0.063	0.048	0.035
	KappaSD (Training)	0.027	0.11	0.093	0.051	0.095	0.085	0.124	0.096	0.07
	Accuracy (Test)	0.714	0.489	0.586	0.464	0.493	0.557	0.554	0.489	0.511
	Kappa (Test)	0.43	-0.02	0.121	-0.072	-0.018	0.065	0.049	-0.015	-0.033
GS	Lag	0	1	2	3	4	5	6	7	8
	InputSet	5	5	8	2	5	5	8	5	11
	Sigma	0.992	0.992	3.236	0.187	0.992	0.992	3.236	0.992	0.381
	Cost	128	128	0.5	0.25	1	64	32	16	0.25
	Accuracy (Training)	0.734	0.563	0.572	0.57	0.578	0.576	0.553	0.572	0.562
	Kappa (Training)	0.462	0.109	0.109	0.104	0.118	0.132	0.094	0.126	0.061
	AccuracySD (Training)	0.023	0.071	0.034	0.032	0.031	0.05	0.02	0.063	0.014
	KappaSD (Training)	0.045	0.143	0.071	0.066	0.064	0.101	0.03	0.119	0.029
	Accuracy (Test)	0.686	0.482	0.593	0.571	0.521	0.468	0.55	0.493	0.529
	Kappa (Test)	0.333	-0.001	0.06	0.102	0.006	-0.03	0.01	0.007	-0.02
INTC	Lag	0	1	2	3	4	5	6	7	8
	InputSet	5	9	5	2	1	6	7	7	4
	Sigma	0.509	5.75	0.477	0.188	0.25	1.727	1.017	1.011	4.235
	Cost	64	4	64	0.25	0.5	8	0.5	0.25	2
	Accuracy (Training)	0.743	0.555	0.558	0.553	0.539	0.537	0.519	0.56	0.533
	Kappa (Training)	0.485	0.108	0.112	0.118	0.079	0.07	0.035	0.112	0.066
	AccuracySD (Training)	0.02	0.065	0.05	0.028	0.058	0.061	0.039	0.048	0.051
	KappaSD (Training)	0.04	0.129	0.101	0.057	0.116	0.121	0.075	0.093	0.101
	Accuracy (Test)	0.529	0.532	0.496	0.457	0.468	0.507	0.536	0.571	0.596
	Kappa (Test)	0.074	0.007	0.007	-0.073	-0.064	-0.046	0.005	0.071	0.187
IBM	Lag	0	1	2	3	4	5	6	7	8
	InputSet	5	6	4	5	4	4	8	4	4

	Sigma	0.537	1.481	3.961	0.537	3.84	3.84	8.621	3.84	3.84
	Cost	32	4	0.5	32	1	2	0.25	2	0.25
	Accuracy (Training)	0.777	0.535	0.547	0.578	0.579	0.524	0.579	0.554	0.556
	Kappa (Training)	0.554	0.069	0.095	0.155	0.159	0.049	0.157	0.109	0.11
	AccuracySD (Training)	0.03	0.017	0.035	0.044	0.068	0.031	0.038	0.049	0.051
	KappaSD (Training)	0.061	0.035	0.07	0.088	0.135	0.062	0.076	0.098	0.1
	Accuracy (Test)	0.671	0.554	0.518	0.496	0.482	0.514	0.5	0.511	0.493
	Kappa (Test)	0.339	0.048	0.029	0.003	-0.03	0.039	-0.062	0.028	-0.018
JPM	Lag	0	1	2	3	4	5	6	7	8
	InputSet	5	3	5	9	6	3	3	4	2
	Sigma	0.634	16.456	0.634	5.743	2.92	16.456	16.456	3.188	0.187
	Cost	64	0.25	16	128	16	128	0.25	128	0.25
	Accuracy (Training)	0.729	0.533	0.545	0.524	0.547	0.535	0.542	0.554	0.56
	Kappa (Training)	0.455	0.05	0.087	0.046	0.091	0.063	0.063	0.101	0.094
	AccuracySD (Training)	0.031	0.024	0.038	0.055	0.03	0.022	0.04	0.054	0.035
	KappaSD (Training)	0.064	0.047	0.07	0.113	0.057	0.041	0.079	0.107	0.076
	Accuracy (Test)	0.629	0.521	0.546	0.532	0.496	0.468	0.546	0.536	0.521
	Kappa (Test)	0.225	0.015	0.102	-0.003	-0.081	-0.118	0.043	0.048	-0.033
GMCR	Lag	0	1	2	3	4	5	6	7	8
	InputSet	4	1	2	9	11	7	9	5	7
	Sigma	4.661	0.25	0.188	18.339	0.691	1.183	19.176	0.733	1.183
	Cost	64	0.25	0.25	0.25	0.25	128	8	8	8
	Accuracy (Training)	0.747	0.519	0.547	0.563	0.522	0.571	0.545	0.544	0.549
	Kappa (Training)	0.491	0	0.093	0.11	0.024	0.14	0.081	0.078	0.091
	AccuracySD (Training)	0.026	0.002	0.019	0.043	0.031	0.06	0.02	0.035	0.023
	KappaSD (Training)	0.053	0	0.037	0.085	0.062	0.118	0.041	0.07	0.046
	Accuracy (Test)	0.771	0.479	0.511	0.536	0.493	0.532	0.557	0.546	0.525
	Kappa (Test)	0.543	0	0.019	0.038	0.008	0.014	0.078	0.076	0.001
MU	Lag	0	1	2	3	4	5	6	7	8
	InputSet	4	3	7	2	2	3	3	3	10
	Sigma	3.008	39.036	0.91	0.187	0.187	43.103	18.087	48.579	0.995
	Cost	16	8	1	0.25	0.25	1	32	2	8
	Accuracy (Training)	0.753	0.604	0.582	0.565	0.553	0.556	0.552	0.566	0.572
	Kappa (Training)	0.502	0.198	0.163	0.097	0.073	0.095	0.095	0.123	0.137
	AccuracySD (Training)	0.021	0.073	0.056	0.043	0.032	0.043	0.034	0.037	0.045
	KappaSD (Training)	0.043	0.145	0.116	0.092	0.069	0.09	0.066	0.069	0.094
	Accuracy (Test)	0.711	0.525	0.546	0.518	0.511	0.532	0.529	0.529	0.489
	Kappa (Test)	0.414	-0.02	0.036	0.004	-0.011	0.016	0.008	0.001	-0.024
MSFT	Lag	0	1	2	3	4	5	6	7	8
	InputSet	4	8	7	2	1	4	11	9	10
	Sigma	4.293	6.552	1.059	0.188	0.25	4.293	0.496	4.43	0.552
	Cost	8	64	1	0.25	0.25	32	16	4	32
	Accuracy (Training)	0.738	0.533	0.569	0.549	0.549	0.537	0.565	0.538	0.545
	Kappa (Training)	0.476	0.066	0.137	0.098	0.098	0.074	0.129	0.075	0.091
	AccuracySD (Training)	0.057	0.035	0.038	0.042	0.05	0.014	0.035	0.043	0.022
	KappaSD (Training)	0.113	0.069	0.076	0.084	0.099	0.028	0.071	0.086	0.044
	Accuracy (Test)	0.721	0.507	0.532	0.511	0.507	0.457	0.543	0.604	0.511
	Kappa (Test)	0.442	-0.041	0.016	0.024	0.017	-0.076	0.086	0.148	0.021
NFLX	Lag	0	1	2	3	4	5	6	7	8
	InputSet	4	7	8	10	4	9	6	3	7
	Sigma	4.138	1.201	15.03	0.805	4.138	12.704	3.368	25.068	1.201
	Cost	4	1	2	4	64	16	0.5	8	32
	Accuracy (Training)	0.724	0.54	0.547	0.57	0.57	0.563	0.579	0.56	0.551
	Kappa (Training)	0.448	0.079	0.093	0.143	0.14	0.126	0.16	0.119	0.1
	AccuracySD (Training)	0.025	0.041	0.083	0.042	0.064	0.023	0.024	0.057	0.033
	KappaSD (Training)	0.051	0.08	0.165	0.084	0.129	0.046	0.048	0.115	0.068
	Accuracy (Test)	0.718	0.55	0.504	0.511	0.529	0.539	0.532	0.518	0.568
	Kappa (Test)	0.43	0.038	-0.035	-0.005	0.075	0.002	-0.01	0.059	0.093
PCLN	Lag	0	1	2	3	4	5	6	7	8
	InputSet	5	8	3	11	3	3	3	8	4
	Sigma	0.578	3.209	15.3	0.427	15.3	15.3	15.3	3.209	3.939
	Cost	32	8	128	64	0.5	2	0.5	64	4
	Accuracy (Training)	0.743	0.57	0.556	0.565	0.554	0.545	0.556	0.556	0.558
	Kappa (Training)	0.484	0.112	0.094	0.123	0.072	0.057	0.076	0.1	0.101
	AccuracySD (Training)	0.056	0.023	0.037	0.034	0.015	0.022	0.018	0.057	0.024
	KappaSD (Training)	0.111	0.046	0.076	0.073	0.037	0.039	0.037	0.112	0.051
	Accuracy (Test)	0.704	0.5	0.514	0.5	0.5	0.489	0.518	0.514	0.493
	Kappa (Test)	0.408	-0.031	0.027	-0.006	0.014	-0.005	0.049	-0.029	-0.005
CRM	Lag	0	1	2	3	4	5	6	7	8
	InputSet	4	4	2	8	11	6	9	8	4
	Sigma	4.611	3.291	0.187	11.188	0.691	1.583	10.623	11.185	3.281
	Cost	2	0.25	0.5	8	4	1	1	2	0.25
	Accuracy (Training)	0.746	0.578	0.561	0.575	0.565	0.555	0.556	0.56	0.544
	Kappa (Training)	0.488	0.141	0.094	0.13	0.112	0.07	0.096	0.091	0.034
	AccuracySD (Training)	0.023	0.043	0.043	0.016	0.031	0.029	0.033	0.036	0.038
	KappaSD (Training)	0.048	0.088	0.09	0.036	0.061	0.066	0.066	0.076	0.075
	Accuracy (Test)	0.725	0.486	0.529	0.539	0.482	0.568	0.554	0.568	0.514
	Kappa (Test)	0.448	-0.034	0.044	0.018	-0.038	0.072	0.037	0.07	-0.002

SBUX	Lag	0	1	2	3	4	5	6	7	8
	InputSet	4	11	6	4	8	4	5	7	5
	Sigma	4.182	0.517	1.537	4.862	5.195	4.848	0.68	0.951	0.677
	Cost	0.5	64	0.25	128	16	32	64	4	32
	Accuracy (Training)	0.734	0.517	0.553	0.551	0.549	0.555	0.546	0.528	0.547
	Kappa (Training)	0.467	0.034	0.105	0.102	0.098	0.109	0.091	0.056	0.094
	AccuracySD (Training)	0.056	0.033	0.025	0.026	0.04	0.041	0.049	0.061	0.032
	KappaSD (Training)	0.112	0.065	0.05	0.054	0.08	0.083	0.097	0.122	0.064
	Accuracy (Test)	0.75	0.496	0.536	0.496	0.514	0.543	0.489	0.568	0.525
	Kappa (Test)	0.49	-0.014	0.061	-0.021	-0.027	0.067	-0.048	0.095	0.047
RIG	Lag	0	1	2	3	4	5	6	7	8
	InputSet	4	8	6	3	7	3	7	7	8
	Sigma	3.845	2.269	1.78	3.846	1.179	4.812	1.092	0.616	2.252
	Cost	16	0.25	0.5	0.5	16	0.25	0.5	0.5	0.25
	Accuracy (Training)	0.752	0.552	0.572	0.572	0.559	0.542	0.568	0.555	0.556
	Kappa (Training)	0.504	0.09	0.143	0.145	0.114	0.083	0.138	0.105	0.116
	AccuracySD (Training)	0.041	0.018	0.03	0.037	0.022	0.077	0.04	0.045	0.038
	KappaSD (Training)	0.082	0.037	0.061	0.075	0.043	0.152	0.082	0.085	0.074
	Accuracy (Test)	0.736	0.568	0.529	0.525	0.561	0.521	0.614	0.618	0.493
	Kappa (Test)	0.463	-0.001	-0.055	-0.029	0.106	-0.039	0.111	0.084	0.022
YHOO	Lag	0	1	2	3	4	5	6	7	8
	InputSet	4	9	2	8	9	2	7	7	4
	Sigma	7.391	12.435	0.188	22.849	13.098	0.188	1.399	1.398	3.9
	Cost	0.25	0.25	0.5	64	0.5	0.25	32	0.5	64
	Accuracy (Training)	0.739	0.551	0.557	0.553	0.576	0.55	0.549	0.535	0.557
	Kappa (Training)	0.473	0.071	0.09	0.106	0.138	0.081	0.089	0.042	0.111
	AccuracySD (Training)	0.043	0.023	0.058	0.07	0.065	0.073	0.043	0.021	0.024
	KappaSD (Training)	0.088	0.048	0.116	0.135	0.132	0.146	0.085	0.039	0.049
	Accuracy (Test)	0.754	0.6	0.514	0.568	0.582	0.518	0.6	0.536	0.493
	Kappa (Test)	0.501	0.106	0.003	0.055	0.088	0.013	0.137	-0.014	-0.016
BBY	Lag	0	1	2	3	4	5	6	7	8
	InputSet	5	5	4	5	6	11	5	4	11
	Sigma	0.535	0.603	2.665	0.642	3.3	0.923	0.626	3.916	1.117
	Cost	64	0.5	32	4	0.25	64	8	16	2
	Accuracy (Training)	0.741	0.568	0.566	0.547	0.558	0.531	0.54	0.561	0.562
	Kappa (Training)	0.48	0.13	0.132	0.091	0.094	0.059	0.076	0.119	0.117
	AccuracySD (Training)	0.058	0.016	0.037	0.037	0.041	0.039	0.049	0.052	0.007
	KappaSD (Training)	0.117	0.031	0.072	0.073	0.082	0.078	0.098	0.107	0.012
	Accuracy (Test)	0.67	0.527	0.505	0.502	0.527	0.43	0.502	0.523	0.477
	Kappa (Test)	0.319	0.017	0.02	-0.021	0.029	-0.123	-0.035	0.065	-0.025
GILD	Lag	0	1	2	3	4	5	6	7	8
	InputSet	5	10	7	5	2	7	8	11	5
	Sigma	0.694	0.842	1.33	0.535	0.186	1.516	34.424	1.412	0.551
	Cost	8	1	0.5	0.25	1	1	0.5	8	128
	Accuracy (Training)	0.736	0.575	0.573	0.561	0.563	0.572	0.595	0.583	0.564
	Kappa (Training)	0.463	0.102	0.082	0	0.074	0.075	0.131	0.132	0.11
	AccuracySD (Training)	0.041	0.046	0.042	0.005	0.028	0.029	0.023	0.037	0.073
	KappaSD (Training)	0.077	0.095	0.088	0	0.061	0.059	0.049	0.079	0.145
	Accuracy (Test)	0.521	0.486	0.536	0.557	0.532	0.6	0.596	0.543	0.45
	Kappa (Test)	0.111	-0.078	-0.056	0	-0.007	0.116	0.017	0.04	-0.002
KORS	Lag	0	1	2	3	4	5	6	7	8
	InputSet	5	5	8	2	4	4	11	2	8
	Sigma	0.758	0.567	8.195	0.187	4.05	4.636	0.615	0.187	8.192
	Cost	128	4	0.5	0.25	2	0.5	0.5	0.25	0.5
	Accuracy (Training)	0.737	0.544	0.533	0.557	0.566	0.546	0.556	0.558	0.556
	Kappa (Training)	0.47	0.072	0.02	0.088	0.105	0.074	0.074	0.088	0.09
	AccuracySD (Training)	0.045	0.05	0.02	0.061	0.044	0.02	0.023	0.029	0.017
	KappaSD (Training)	0.09	0.1	0.04	0.127	0.088	0.036	0.045	0.059	0.036
	Accuracy (Test)	0.671	0.493	0.518	0.482	0.464	0.479	0.504	0.496	0.518
	Kappa (Test)	0.351	0.022	0.042	0.009	-0.015	0.006	0.06	0.034	0.02
FB	Lag	0	1	2	3	4	5	6	7	8
	InputSet	4	3	4	3	3	8	10	5	11
	Sigma	3.748	25.603	3.748	25.603	25.603	26.231	0.58	0.957	0.474
	Cost	2	1	8	0.5	2	4	128	2	4
	Accuracy (Training)	0.707	0.542	0.585	0.547	0.543	0.551	0.526	0.562	0.587
	Kappa (Training)	0.415	0.084	0.17	0.093	0.085	0.101	0.052	0.122	0.174
	AccuracySD (Training)	0.041	0.054	0.02	0.055	0.067	0.073	0.077	0.036	0.046
	KappaSD (Training)	0.083	0.106	0.041	0.11	0.134	0.146	0.154	0.072	0.091
	Accuracy (Test)	0.743	0.536	0.496	0.539	0.525	0.518	0.464	0.482	0.479
	Kappa (Test)	0.488	0.011	0.003	0.021	-0.005	0.01	-0.085	-0.015	-0.053

Table 5: Prediction Accuracies - SVM Model

5 Strategy Specification

R packages quantstrat [21] and blotter [20] are the "workhorses" when it comes to backtesting signal-based quantitative strategies. quantstrat supports strategies which include indicators, signals, and rules and allows strategies to be applied to multi-asset portfolios. It supports market, limit, stoplimit, and stoptrailing order types, order sizing and parameter optimization. blotter does low-level trading system accounting that supports P&L calculation and roll-up across instruments and portfolios.

Indicators are quantitative values derived from market data. Interaction between indicators and market data are used to generate signals (e.g. crossovers, thresholds). Signal processes describe the desire for an action, but the strategy may choose not to act or may not be actionable at the time. Rules make path-dependent actionable decisions and use market data, indicators, signals, and current account/portfolio characteristics to generate orders.

5.1 Indicators

By designing the indicator to have some *stickiness* and to reduce the number of transactions which may occur due to *noise*, the SVM predictions of the n-day-ahead stock price directions are aggregated to create a single indicator.

Specifically, the predicted classes for each n-day-ahead model at time t are added together.

Model_2	Model_1	Indicator
-1.00	-1.00	-2.00
-1.00	1.00	0.00
1.00	-1.00	0.00
1.00	1.00	2.00

Table 6: Possible Predictions and Indicator Value - 2 models

Model_3	Model_2	Model_1	Indicator
-1.00	-1.00	-1.00	-3.00
-1.00	-1.00	1.00	-1.00
-1.00	1.00	-1.00	-1.00
-1.00	1.00	1.00	1.00
1.00	-1.00	-1.00	-1.00
1.00	-1.00	1.00	1.00
1.00	1.00	-1.00	1.00
1.00	1.00	1.00	3.00

Table 7: Possible Predictions and Indicator Value - 3 models

Model_4	Model_3	Model_2	Model_1	Indicator
-1.00	-1.00	-1.00	-1.00	-4.00
-1.00	-1.00	-1.00	1.00	-2.00
-1.00	-1.00	1.00	-1.00	-2.00
-1.00	-1.00	1.00	1.00	0.00
-1.00	1.00	-1.00	-1.00	-2.00
-1.00	1.00	-1.00	1.00	0.00
-1.00	1.00	1.00	-1.00	0.00
-1.00	1.00	1.00	1.00	2.00
1.00	-1.00	-1.00	-1.00	-2.00
1.00	-1.00	-1.00	1.00	0.00
1.00	-1.00	1.00	-1.00	0.00
1.00	-1.00	1.00	1.00	2.00
1.00	1.00	-1.00	-1.00	0.00
1.00	1.00	-1.00	1.00	2.00
1.00	1.00	1.00	-1.00	2.00
1.00	1.00	1.00	1.00	4.00

Table 8: Possible Predictions and Indicator Value - 4 models

5.2 Signals

By using a specified *threshold* and *relationship* with the indicator, a signal can be generated. If for example I use two models and the threshold level is 0 and the relationship is "greater than", then a signal could be generated 1 out of 4 ways (when both the predictions are positive).

Another example would be if I use four models and the threshold level is 2 and the relationship is "greater than or equal to", then a signal could be generated 5 out of 16 ways.

5.3 Rules

Entry, Exits, Risk, Profit Taking, and Portfolio Rebalancing are all rule processes.
Rules are path dependent, meaning that they are aware of the current market state, the current portfolio, all working orders, and any other data available at the time the rule is being evaluated. No action is instantaneous, so rules also have a cost in time.

With a limited amount of capital, each transaction value is limited to approximately $4000 by using a Fixed-dollar order sizing. Stop rules are used to help minimise drawdowns and therefore to help preserve capital.

Entry Rule:

- Buy long when the indicator $>$ upper threshold
- Sell short when the indicator $<$ lower threshold

Exit rule:

- Exit any long or short position when either the high or low cross a threshold

Stop rule:

- Stop loss set at X% below entry price
- Stop loss set at X% below position high (trailing stop)

Fixed−dollar order sizing:

- Adjust the share quantity such that the transaction value is approximately equal to a pre-defined trade size (pyramiding allowed)

$$orderqty = \frac{tradeSize}{ClosePrice} \tag{4}$$

Fees and Transaction Costs:

- A cost of $X will be applied to entries and $X to exits

5.4 Indicator and Signal Process Evaluation

The number of potential entry and exit signals can be measured.
Here, I tabulate the counts when the signal is defined as being "equal to the indicator". In each scenario, there are a total of 634 signals. Only the training data was used.

One could extend this analysis by formulating the distribution of the period between entries and exits, degree of overlap between entry and exit signals, and so on.

Indicator	Signal_Count
2.00	256
0.00	289
-2.00	89

Table 9: Signals Generated at Indicator - 2 models

Indicator	Signal_Count
3.00	128
1.00	281
-1.00	175
-3.00	50

Table 10: Signals Generated at Indicator - 3 models

Indicator	Signal_Count
4.00	73
2.00	202
0.00	220
-2.00	111
-4.00	28

Table 11: Signals Generated at Indicator - 4 models

I choose to use 4 models in all future optimisations and backtests because 4 models has a good spread of trades at most indicator levels. I define the upper threshold for the Long Entry as 1, the lower threshold for the Short Sell as -1 and the thresholds for both the long and short exits as -1 and 1 respectively.

5.5 Evaluating Rules

5.5.1 Fixed-Dollar Order Sizing

As at 2011-02-01, the median closing price of the 27 stocks is $31.50.

With only $100K of capital, I set the trade size to be $4,000. This allows a total of 25 trades to be open at any point in time. Therefore, on average, the order quantity is expected to be;

$$orderqty = \frac{4000}{31.50} = 127 \tag{5}$$

5.5.2 MAE Analysis - Finding Exits

In order to determine where to place the stop losses, I run a backtest using the parameter settings I have just calculated;

Entry Rule:

- Buy long when the indicator $>$ 1
- Sell short when the indicator $<$ -1

Exit rule:

- Exit any long position when the indicator crosses -1
- Exit any short position when the indicator crosses 1

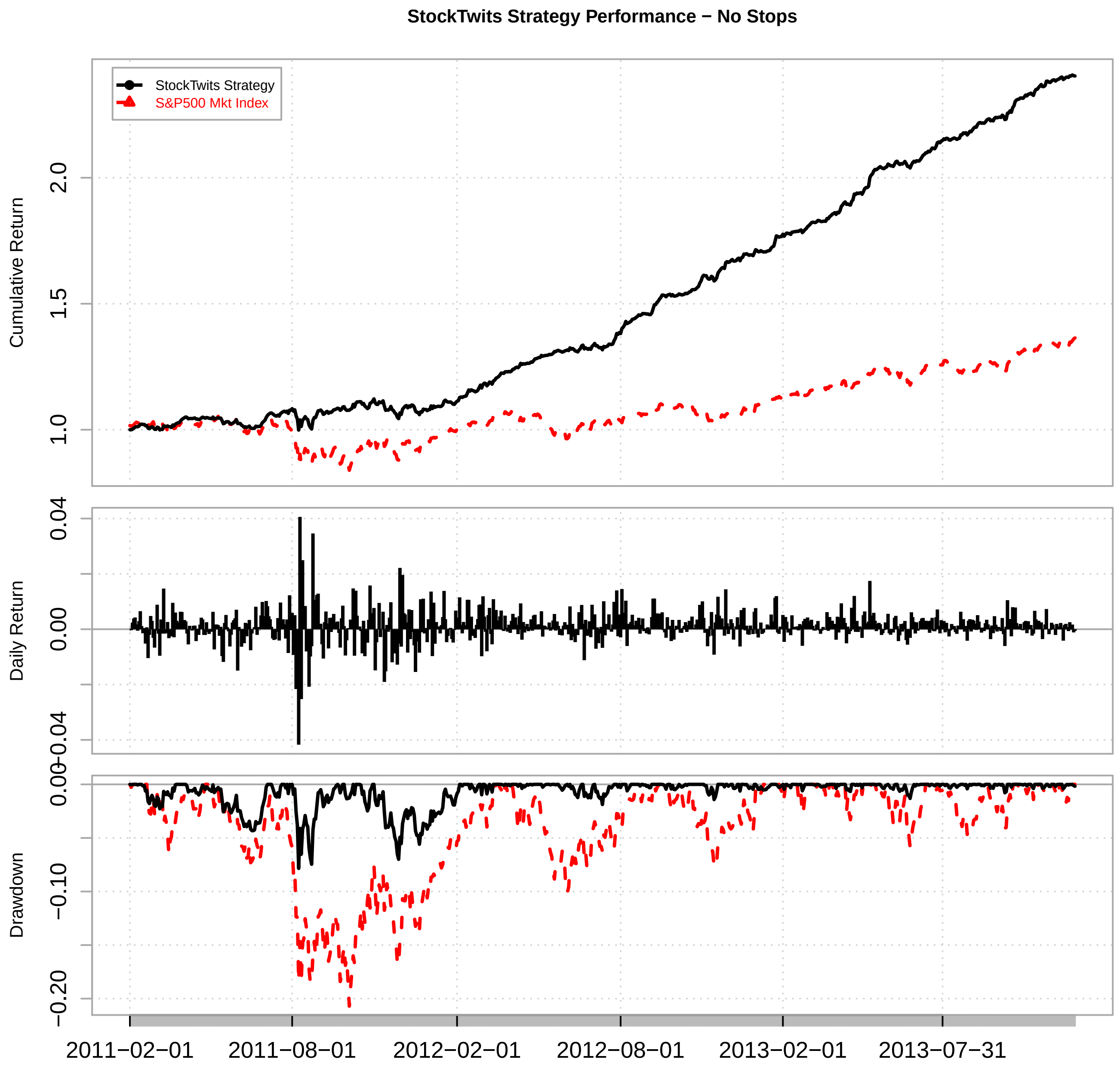
Fixed$-$dollar order sizing:

- Fixed dollar amount of $4000 (approximately 127 orders)

Fees and Transaction Costs:

- A cost of $2.5 will be applied to entries and $3.2 to exits

Pyramiding is allowed.

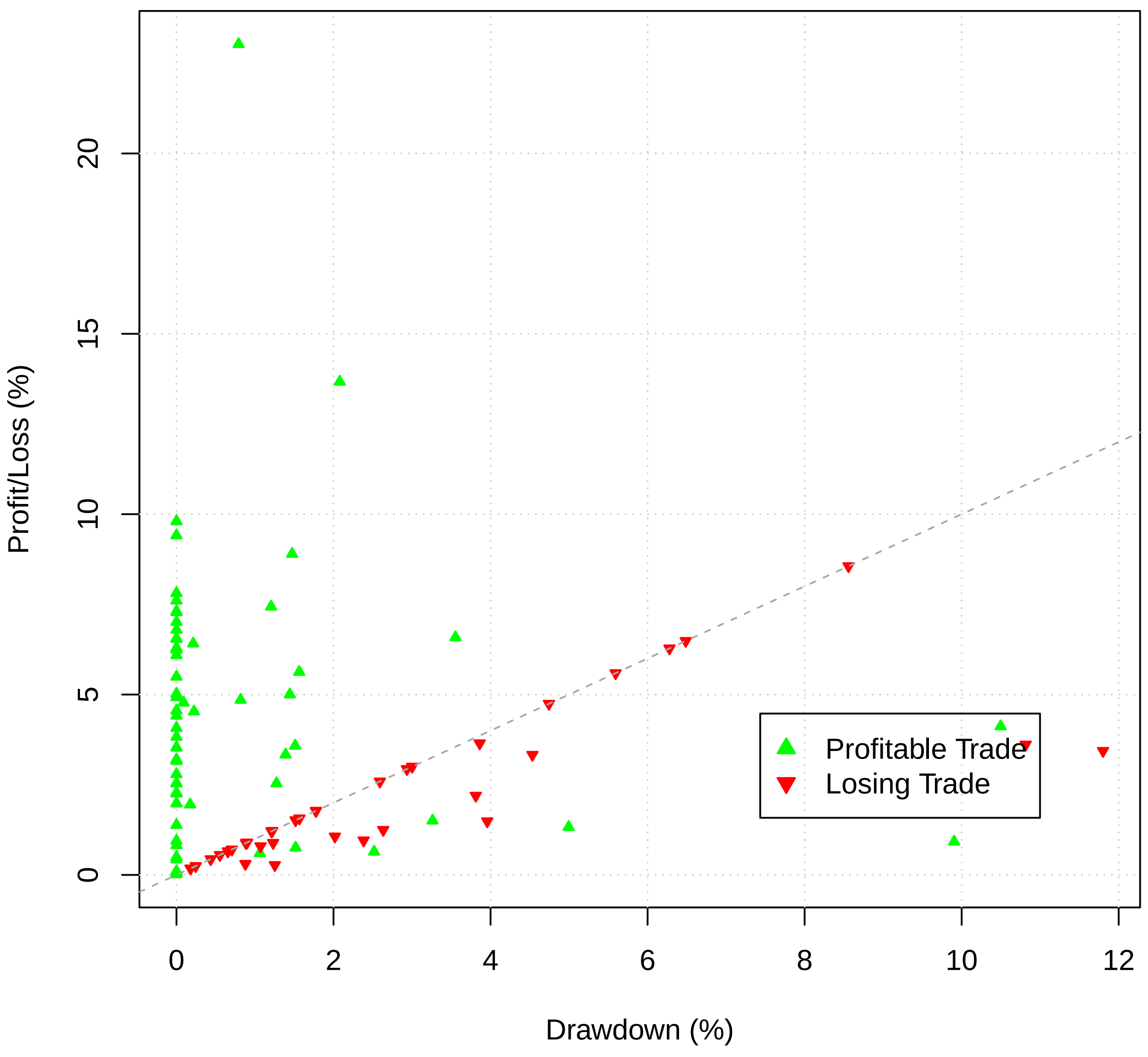


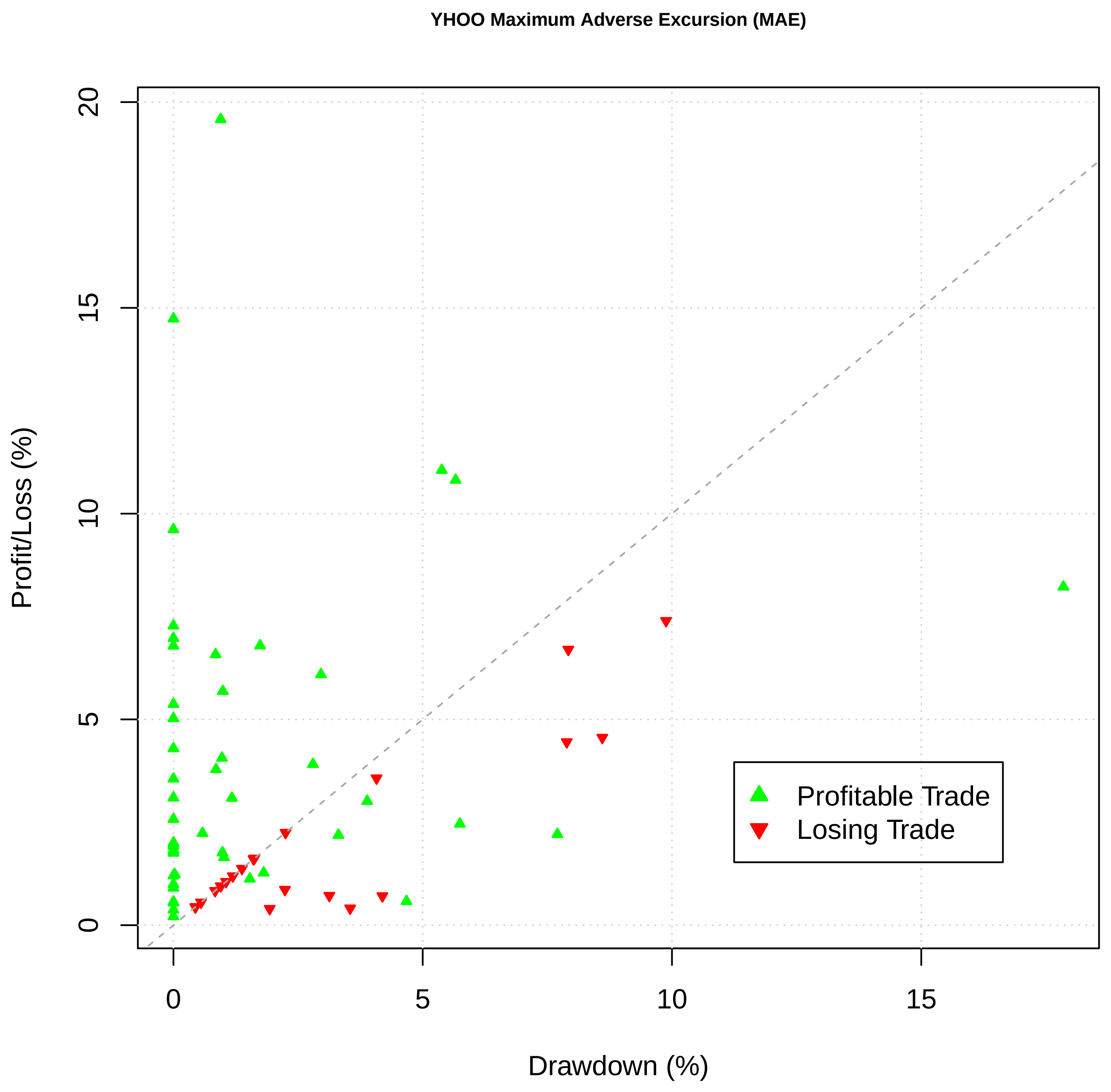
As detailed by Tomasini and Jaekle [36], "The Maximum Adverse Excursion (MAE) is defined as the most intraday price movement against your position. In other words, it's the lowest open equity during the lifespan of a trade. The MAE concept allows you to evaluate your systems' individual trades to determine at what dollar or percentage amount to place your protective stop. Winning and losing trades are clustered on the same graph which makes it easier to figure out how much unrealised loss must be incurred by a trade before it typically does not recover. In this way, the MAE graphic tells you when to cut your loss because the risks associated with the trade are no longer justified".

As an example, the MAE is shown for both AMZN and YHOO. We try to place a stop in an area that captures the majority of winning trades while simultaneously limiting the strategy's exposure to profit erosion. From the graphic, for AMZN this looks to be between 1% and 4% and for YHOO between 2% and 7%.

Trades on the diagonal axis closed on their lows.

AMZN Maximum Adverse Excursion (MAE)





5.5.3 Stop Loss Optimisation

Utilizing all the MAE graphics (not shown), I choose ranges for the stop losses and optimise over multiple backtests.

Optimization range for the stop loss below the entry price:

- 1% to 8% with increments of 1%

Optimization range for the stop loss below the position high (trailing stop):

- 1% to 8% with increments of 1%

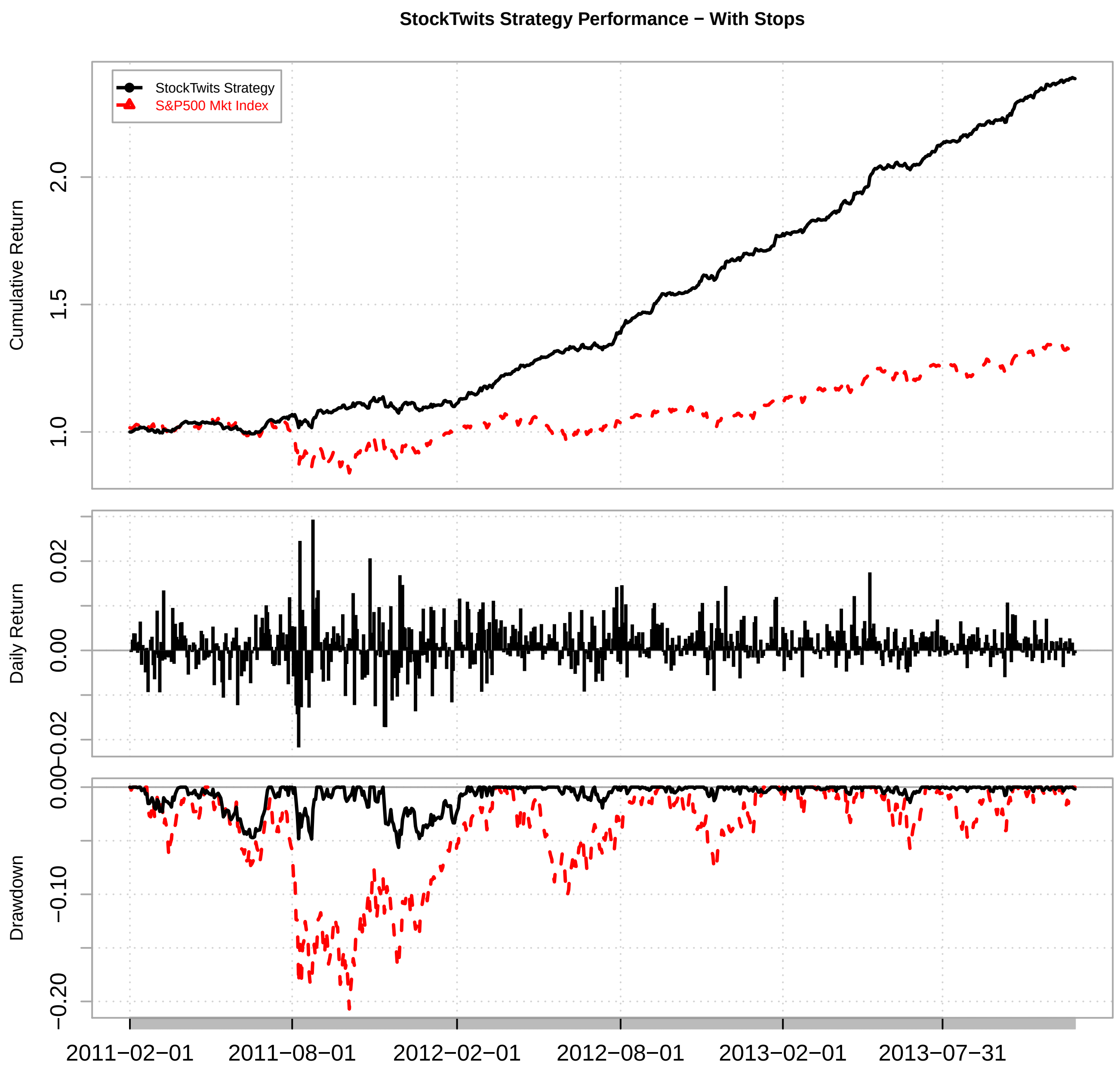
The optimisation constraint is that the stop loss below the entry price must be less than the trailing stop.

After running the optimisation and using a measure looking at the Return to Maximum Drawdown versus the Stop Loss and Trailing Stop, I select the optimal stop losses as;

Stop rule:

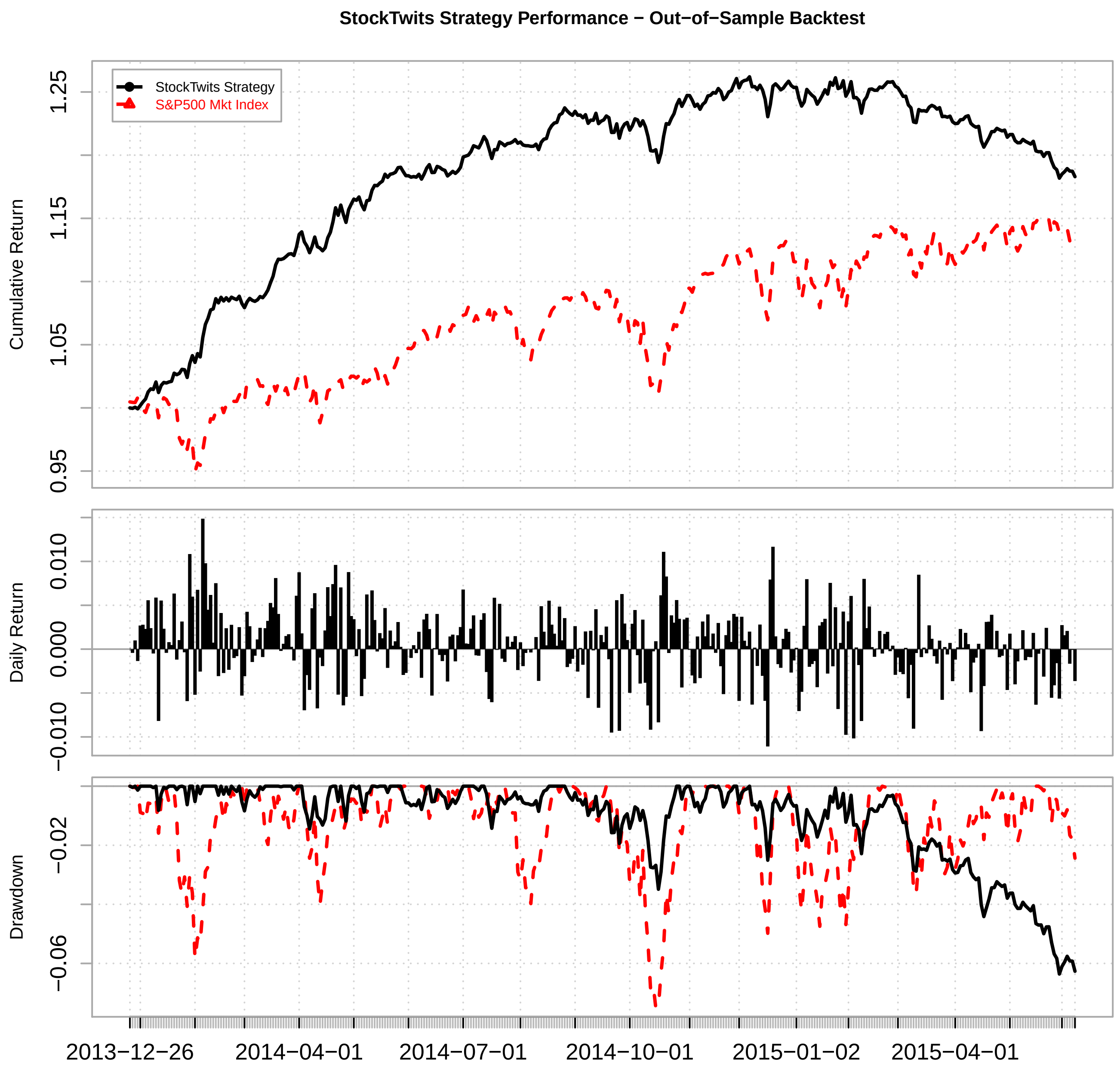
- Stop loss set at 4% below entry price
- Stop loss set at 5% below position high (trailing stop)

Running another backtest on the training data but this time with the stops reveals a slightly better performance.



6 Evaluating the Strategy on an Out-of-Sample Backtest

Now that the parameters of the strategy have been optimised, I run a backtest using the out-of-sample test dataset.



7 Results

The overall accuracy of the model is quite good, especially given that market data is generally very noisy. But it could improve. It's possible that the features by themselves do not characterize the stocks well enough or perhaps the hyper-parameters are not *tuned* appropriately.

The out-of-sample tests degraded a little but not too much. Some explanations for out-of-sample deterioration could be that the market dynamics within the training data range is different from the one in the test data range. Or the system has been adapted too much to market noise within the training period, i.e. curve over-fitting.

However, the returns of the backtest are very good and "beat" the market. I do have some concern regarding the more recent returns in 2015 though. The model may need to be *re-trained* on more recent data to allow the model to adjust to the most recent market conditions. Looking at the summary trade statistics, the strategy met most of my objectives; the Sharpe Ratios on average are excellent and the drawdowns are shallow and thus risk is managed accordingly.

	AA	AAPL	AMZN	BAC	BBY	C	CAT
Portfolio	stockTwits	stockTwits	stockTwits	stockTwits	stockTwits	stockTwits	stockTwits
Symbol	AA	AAPL	AMZN	BAC	BBY	C	CAT
Num.Txns	95	33	121	112	81	178	123
Num.Trades	95	33	121	112	81	178	123
Net.Trading.PL	-32.57	2372.49	999.28	2385.76	3009.24	-1551.49	221.51
Avg.Trade.PL	2.097	74.057	7.604	21.301	38.554	-8.716	1.769
Med.Trade.PL	-2.5	-2.5	-2.5	-2.5	-2.5	-2.5	-2.5
Largest.Winner	553.0	1287.9	562.7	587.2	606.0	223.8	332.7
Largest.Loser	-547.7	-102.7	-721.6	-141.4	-245.7	-262.2	-216.6
Gross.Profits	3352	2681	3624	3803	4538	1904	2609
Gross.Losses	-3152.5	-237.0	-2703.7	-1417.1	-1415.5	-3455.2	-2391.5
Std.Dev.Trade.PL	132.79	239.61	116.62	93.56	129.50	56.38	73.99
Percent.Positive	22.11	33.33	27.27	33.04	37.04	20.79	28.46
Percent.Negative	77.89	66.67	72.73	66.96	62.96	79.21	71.54
Profit.Factor	1.0632	11.3133	1.3403	2.6835	3.2062	0.5510	1.0910
Avg.Win.Trade	159.61	243.71	109.81	102.78	151.28	51.45	74.54
Med.Win.Trade	118.48	114.04	73.40	70.85	111.00	33.65	49.32
Avg.Losing.Trade	-42.60	-10.77	-30.72	-18.90	-27.76	-24.51	-27.18
Med.Losing.Trade	-2.5	-2.5	-2.5	-2.5	-2.5	-2.5	-2.5
Avg.Daily.PL	2.097	74.057	7.604	21.301	38.554	-8.716	1.769
Med.Daily.PL	-2.5	-2.5	-2.5	-2.5	-2.5	-2.5	-2.5
Std.Dev.Daily.PL	132.79	239.61	116.62	93.56	129.50	56.38	73.99
Ann.Sharpe	0.2507	4.9064	1.0351	3.6143	4.7261	-2.4540	0.3795
Max.Drawdown	-1579.5	-513.9	-1360.3	-603.8	-636.6	-1815.2	-638.1
Profit.To.Max.Draw	-0.02062	4.61660	0.73461	3.95100	4.72672	-0.85470	0.34714
Avg.WinLoss.Ratio	3.746	22.627	3.574	5.440	5.450	2.100	2.743
Med.WinLoss.Ratio	47.39	45.62	29.36	28.34	44.40	13.46	19.73
Max.Equity	1546.9	2662.6	2228.7	2416.1	3387.0	181.1	670.0
Min.Equity	-143.753	-8.925	-134.200	-35.853	-51.704	-1634.111	-77.042
End.Equity	-32.57	2372.49	999.28	2385.76	3009.24	-1551.49	221.51

	CMG	CRM	EBAY	FB	FF	FSLR	GILD
Portfolio	stockTwits	stockTwits	stockTwits	stockTwits	stockTwits	stockTwits	stockTwits
Symbol	CMG	CRM	EBAY	FB	FF	FSLR	GILD
Num.Txns	133	107	103	29	125	141	21
Num.Trades	133	107	103	29	125	141	21
Net.Trading.PL	670.30	469.59	163.32	2614.57	1089.49	-648.76	1945.88
Avg.Trade.PL	6.095	4.562	2.981	76.640	8.716	-3.778	64.706
Med.Trade.PL	-2.5	-2.5	-2.5	-2.5	-2.5	-2.5	-2.5
Largest.Winner	704.8	508.0	462.3	629.6	574.8	1373.7	1001.7
Largest.Loser	-296.4	-356.6	-199.8	-378.6	-189.3	-458.9	-165.1
Gross.Profits	4264	3542	2061	2742	3127	5419	2135
Gross.Losses	-3453.1	-3053.6	-1753.5	-519.2	-2037.6	-5952.0	-776.2
Std.Dev.Trade.PL	118.13	117.43	76.08	195.42	88.87	178.41	269.32
Percent.Positive	25.56	24.30	29.13	34.48	26.40	21.28	19.05
Percent.Negative	74.44	75.70	70.87	65.52	73.60	78.72	80.95
Profit.Factor	1.2348	1.1598	1.1751	5.2806	1.5347	0.9105	2.7505
Avg.Win.Trade	125.41	136.22	68.68	274.18	94.76	180.64	533.76
Med.Win.Trade	67.95	74.85	44.82	227.33	50.76	102.35	390.16
Avg.Losing.Trade	-34.88	-37.70	-24.02	-27.33	-22.15	-53.62	-45.66
Med.Losing.Trade	-2.5	-2.5	-2.5	-2.5	-2.5	-2.5	-2.5
Avg.Daily.PL	6.095	4.562	2.981	76.640	8.716	-3.778	64.706
Med.Daily.PL	-2.5	-2.5	-2.5	-2.5	-2.5	-2.5	-2.5
Std.Dev.Daily.PL	118.13	117.43	76.08	195.42	88.87	178.41	269.32
Ann.Sharpe	0.8191	0.6166	0.6221	6.2258	1.5569	-0.3362	3.8139
Max.Drawdown	-1228.4	-1094.3	-632.9	-482.3	-478.2	-1853.6	-987.2
Profit.To.Max.Draw	0.54564	0.42913	0.25803	5.42096	2.27848	-0.35000	1.97112
Avg.WinLoss.Ratio	3.595	3.613	2.859	10.033	4.279	3.369	11.690
Med.WinLoss.Ratio	27.18	29.94	17.93	90.93	20.30	40.94	156.06
Max.Equity	1858.5	1291.9	627.7	2846.6	1559.0	838.4	1986.9
Min.Equity	-362.100	-68.300	-425.994	-2.500	-81.497	-1015.239	-523.098
End.Equity	670.30	469.59	163.32	2614.57	1089.49	-648.76	1945.88

	GMCR	GS	IBM	INTC	JPM	KORS	MSFT
Portfolio	stockTwits	stockTwits	stockTwits	stockTwits	stockTwits	stockTwits	stockTwits
Symbol	GMCR	GS	IBM	INTC	JPM	KORS	MSFT
Num.Txns	41	77	108	95	79	47	125
Num.Trades	41	77	108	95	79	47	125
Net.Trading.PL	1818.53	467.19	1850.55	-604.06	2414.10	-1817.50	436.34
Avg.Trade.PL	44.355	3.500	17.135	-10.318	27.472	-8.894	4.342
Med.Trade.PL	-2.5	-2.5	-2.5	-2.5	-2.5	-2.5	-2.5
Largest.Winner	2383.8	387.7	356.4	281.5	362.5	472.0	604.3
Largest.Loser	-1143.9	-207.9	-117.7	-679.1	-172.7	-549.2	-198.8
Gross.Profits	4121	1575	2972	1919	3163	1880	2446
Gross.Losses	-2302.1	-1305.4	-1121.9	-2898.9	-992.7	-2297.6	-1903.0
Std.Dev.Trade.PL	436.21	70.75	70.16	111.23	92.21	156.55	75.00
Percent.Positive	19.51	27.27	32.41	24.21	32.91	23.40	25.60
Percent.Negative	80.49	72.73	67.59	75.79	67.09	76.60	74.40
Profit.Factor	1.7899	1.2065	2.6495	0.6619	3.1863	0.8181	1.2852
Avg.Win.Trade	515.08	75.00	84.93	83.42	121.65	170.87	76.43
Med.Win.Trade	256.95	68.06	49.95	68.36	92.14	188.80	49.37
Avg.Losing.Trade	-69.76	-23.31	-15.37	-40.26	-18.73	-63.82	-20.46
Med.Losing.Trade	-2.5	-2.5	-2.5	-2.5	-2.5	-2.5	-2.5
Avg.Daily.PL	44.355	3.500	17.135	-10.318	27.472	-8.894	4.342
Med.Daily.PL	-2.5	-2.5	-2.5	-2.5	-2.5	-2.5	-2.5
Std.Dev.Daily.PL	436.21	70.75	70.16	111.23	92.21	156.55	75.00
Ann.Sharpe	1.6141	0.7854	3.8771	-1.4725	4.7292	-0.9018	0.9190
Max.Drawdown	-2129.3	-408.9	-421.0	-1771.0	-358.5	-2354.8	-837.6
Profit.To.Max.Draw	0.85406	1.14265	4.39544	-0.34109	6.73381	-0.77181	0.52096
Avg.WinLoss.Ratio	7.383	3.217	5.526	2.072	6.495	2.677	3.735
Med.WinLoss.Ratio	102.78	27.22	19.98	27.34	36.86	75.52	19.75
Max.Equity	3947.8	480.4	1850.5	568.9	2445.7	418.8	655.7
Min.Equity	-47.203	-408.865	-65.483	-1202.083	-48.416	-1935.999	-541.153
End.Equity	1818.53	467.19	1850.55	-604.06	2414.10	-1817.50	436.34

	MU	NFLX	RIG	SBUX	YHOO
Portfolio	stockTwits	stockTwits	stockTwits	stockTwits	stockTwits
Symbol	MU	NFLX	RIG	SBUX	YHOO
Num.Txns	117	143	53	125	57
Num.Trades	117	143	53	125	57
Net.Trading.PL	1539.09	-2716.22	1209.57	-1096.13	1087.88
Avg.Trade.PL	14.232	-18.995	-12.022	-8.769	20.886
Med.Trade.PL	-2.5	-2.5	-2.5	-2.5	-2.5
Largest.Winner	703.3	529.7	186.4	230.1	1709.5
Largest.Loser	-168.9	-1397.1	-315.7	-249.1	-276.8
Gross.Profits	4879	4469	1031	1666	3204
Gross.Losses	-3213.9	-7185.1	-1668.1	-2761.9	-2013.3
Std.Dev.Trade.PL	138.57	198.90	87.04	64.64	254.43
Percent.Positive	26.50	22.38	16.98	20.00	19.30
Percent.Negative	73.50	77.62	83.02	80.00	80.70
Profit.Factor	1.5181	0.6220	0.6180	0.6031	1.5913
Avg.Win.Trade	157.39	139.65	114.55	66.63	291.26
Med.Win.Trade	91.60	117.20	102.56	45.46	102.80
Avg.Losing.Trade	-37.37	-64.73	-37.91	-27.62	-43.77
Med.Losing.Trade	-2.5	-2.5	-2.5	-2.5	-2.5
Avg.Daily.PL	14.232	-18.995	-12.022	-8.769	20.886
Med.Daily.PL	-2.5	-2.5	-2.5	-2.5	-2.5
Std.Dev.Daily.PL	138.57	198.90	87.04	64.64	254.43
Ann.Sharpe	1.6304	-1.5160	-2.1924	-2.1537	1.3031
Max.Drawdown	-1000.9	-4823.1	-927.6	-1820.1	-1184.4
Profit.To.Max.Draw	1.53764	-0.56317	1.30399	-0.60224	0.91853
Avg.WinLoss.Ratio	4.211	2.157	3.022	2.412	6.655
Med.WinLoss.Ratio	36.64	46.88	41.02	18.19	41.12
Max.Equity	2540.0	2106.9	1884.6	711.3	2272.3
Min.Equity	-42.100	-2716.219	-691.074	-1108.805	-693.982
End.Equity	1539.09	-2716.22	1209.57	-1096.13	1087.88

Table 12: Trade Statistics

.

7.0.4 Extending the Analysis

There are multiple topics I plan to explore by extending the ideas presented in this strategy paper. These include:

- Look at *intra-day* strategies using PsychSignal's sentiment data
- Compare SVM with other *machine learning techniques*
- Research in to how often the classifier should be trained by considering how the model adjusts to the most recent market conditions and *regime shifts*
- Considering a broader range of candidate signals or factors, and applying a *feature selection algorithm* such as the QPFS method [27]
- Run the backtest over a longer time series. A backtest on daily data needs to be run for long enough to have statistical significance in its results

8 Acknowledgements

I would like to thank Brian Peterson from DV Trading (Chicago) and Guy Yollin from the Department of Applied Mathematics at the University of Washington for teaching a great Advanced Systematic Trading Course.

References

[1] Revolution Analytics. domc: Foreach parallel adaptor for the multicore package. https://cran.r-project.org/web/packages/doMC/index.html, 2015.

[2] Social Market Analytics. Sma uses patent-pending technology to estimate sentiment for specific stocks, industries, sectors and indices from social media using twitter's data stream. https://socialmarketanalytics.com/, 2015.

[3] S. Boyd and L. Vandenberghe. *Convex Optimization.* Cambridge University Press, New York, NY, USA, 2004.

[4] B. Caputo, K. Sim, F. Furesjo, and A. Smola. Appearance-based object recognition using svms: Which kernel should i use? *Proceedings of NIPS Workshop on Statistical Methods for Computational Experiments in Visual Processing and Computer Vision*, 2002.

[5] Ernest P Chan. *Quantitative Trading: How to Build Your Own Algorithmic Trading Business.* Wiley Trading, 2009.

[6] Dataminr. Dataminr transforms the twitter stream and other public datasets into actionable signals. https://www.dataminr.com/, 2015.

[7] Brian D Egger. *Social Media Strategies for Investing: How Twitter and Crowdsourcing Tools Can Make You a Smarter Investor.* Adams Media, 2014.

[8] Estimize. Crowdsourced estimates. https://www.estimize.com/, 2015.

[9] Gnip. Offering complete access to realtime streams of public data from the top social networks. https://gnip.com/, 2015.

[10] Harvest. Connects investors through shared expertise across public and private networks from any device. https://www.hvst.com/, 2015.

[11] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction, Second Edition.* Springer-Verlag, 2009.

[12] Ramon. Huerta, Fernando. Corbacho, and Charles. Elkan. Nonlinear support vector machines can systematically identify stocks with high and low future returns. *Algorithmic Finance*, 2(1):45–58.

[13] Alexandros Karatzoglou, Alex Smola, and Kurt Hornik. kernlab: Kernel-based machine learning lab. http://CRAN.R-project.org/package=kernlab, 2015.

[14] Knowsis. Social intelligence for the capital markets. http://www.knowsis.com/, 2015.

[15] Max Kuhn. caret: Classification and regression training. http://CRAN.R-project.org/package=caret, 2015.

[16] Max. Kuhn and Kjell. Johnson. *Applied predictive modeling.* Springer, 2013.

[17] Klaus-Robert. Muller, Sebastian. Mika, Gunnar. Ratsch, Koji. Tsuda, and Bernhard. Scholkopf. An introduction to kernel-based learning algorithms. *IEEE Transactions on Neural Networks*, 12(2):181–202, March 2001.

[18] Brian G. Peterson. Developing & backtesting systematic trading strategies. Technical report, DV Trading, 2015.

[19] Brian G. Peterson and Peter Carl. Performanceanalytics: Econometric tools for performance and risk analysis. http://CRAN.R-project.org/package=PerformanceAnalytics, 2015.

[20] Brian G. Peterson, Joshua Ulrich, Jan Humme, and Peter Carl. blotter: Tools for transaction-oriented trading systems development. http://r-forge.r-project.org/projects/blotter/, 2015.

[21] Brian G. Peterson, Joshua Ulrich, Jan Humme, and Peter Carl. quantstrat: Quantitative strategy model framework. http://r-forge.r-project.org/projects/blotter/, 2015.

[22] Vasilios. Plakandaras, Theophilos. Papadimitriou, Periklis. Gogas, and Konstantinos. Diamantaras. Market sentiment and exchange rate directional forecasting. *Algorithmic Finance*, 4(1-2):69–79, 2015.

[23] Market Prophit. Financial big data analytics for every investor. http://marketprophit.com/, 2015.

[24] PsychSignal. Real-time financial sentiment indices, measuring "bullishness" and "bearishness" by aggregating and filtering mentions on twitter, stocktwits and chat rooms. https://psychsignal.com/, 2015.

[25] Quandl. Find and use data. easily. https://www.quandl.com, 2015.

[26] QuoteMedia. Leading provider of stock market data and research solutions. http://www.quotemedia.com/, 2015.

[27] I. Rodriguez-Lujan, R. Huerta, C. Elkan, and C. Cruz. Quadratic programming feature selection. *Journal of Machine Learning Research*, 11:1491–1516, 2010.

[28] Jeffrey Ryan and Joshua Ulrich. xts: extensible time series. https://cran.r-project.org/web/packages/xts/, 2015.

[29] Jeffrey Ryan, Joshua Ulrich, and Wouter Thielen. quantmod: Quantitative financial modelling framework. https://cran.r-project.org/web/packages/quantmod/, 2015.

[30] John. Shawe-Taylor and Nello. Cristianini. *Kernel Methods for Pattern Analysis.* Cambridge University Press, 2004.

[31] StockTwits. Tap into the pulse of the markets. http://stocktwits.com/, 2015.

[32] SumZero. An investment website for professional investors. https://sumzero.com/, 2105.

[33] James Surowiecki. *The Wisdom of Crowds: Why the Many Are Smarter Than the Few.* Abacus, 2005.

[34] R. Sutton and A. Barto. *Reinforcement learning: An introduction*, volume 1. Cambridge Univ Press, 1998.

[35] R Development Core Team. R: Software environment for statistical computing and graphics. https://www.r-project.org/, 2011.

[36] Emilio. Tomasini and Jaekle. Urban. *Trading Systems: A New Approach to System Development and Portfolio Optimisation.* 2009.

[37] Joshua Ulrich. Ttr: Technical trading rules. http://CRAN.R-project.org/package=TTR, 2015.

[38] Steve Weston. doparallel: Foreach parallel adaptor for the parallel package. https://cran.r-project.org/web/packages/doParallel/index.html, 2015.