Developing a Volume Forecasting Model

Bogdan Batrinca [a],*, Christian W. Hesse [a], Philip C. Treleaven [a]
[a] Department of Computer Science, University College London, Gower Street, London WC1E 6BT, UK

Abstract

This study builds a series of models to predict trading volume in European markets using different statistical methods. The analysis considers a number of aspects, such as special events (e.g. MSCI rebalances, futures expiries, or cross-market holidays), day-of-the-week effects, and the volume-price relation asymmetry, in order to perform contextual one-step ahead prediction. We investigate the prediction error for each calendar circumstance to infer a cross-stock event-oriented switching model for volume prediction. The study concludes by proposing a stock-specific out-of-sample metamodel that is fit by selecting an initial stock-specific model yielding the best performance for the most recent observations.

Keywords:

Trading volume, expiry day effect, holiday effect, behavioral finance, European stock market, feature selection

JEL classification:

C32, C52, C53, G14, G15

Acknowledgements

The majority of the research reported in this paper was a collaboration between UCL and Deutsche Bank, during the time Dr. Christian Hesse worked at Deutsche Bank. We acknowledge the contribution of Deutsche Bank, which drove the real-world topic of this study.

* Corresponding author.
E-mail addresses: bogdan.batrinca.09@ucl.ac.uk (B. Batrinca), c.hesse@ucl.ac.uk (C.W. Hesse), p.treleaven@ucl.ac.uk (P.C. Treleaven).

1. Introduction

Measuring trading performance is a challenging research area, but there are certain factors that have a clear influence on the overall trading performance, such as the market impact, which is the effect caused by a market participant who buys or sells shares, consisting in the extent to which the price goes upward for a buy order or downward for a sell order. The market impact cost is defined as the difference between the actual price and the hypothetical price provided that the order was not created (Johnson, 2010). Market impact can move the prices adversely, leading to decreased profits or turning profitable strategies into losing strategies.

The execution style of an order drives the extent of an order's market impact. An example of a trading strategy to decrease the market impact is when an investor needs to break down a large sell order into smaller orders over a longer period in order to trade slowly with a low market impact. Therefore, predicting the trading volume as a measure of liquidity is of vital importance to forecast the expected market impact.

The aim of this study is to propose a switching volume prediction model by fitting a variety of models that employ different machine learning methods and considering endogenous and exogenous variables that may potentially impact the trading volume. This is motivated by the importance of optimally sizing an order for minimising the market impact and ultimately improving the trading performance. Market participants who size their orders incorrectly can either over-participate by producing excessive market impact or under-participate by creating opportunity cost and price uncertainty. Therefore, predicting the trading volume helps better determine the degree of participation in the market.

The primary focus of this study is to fine-tune the models and identify the optimal model given the market context at a certain point in time, in order to achieve optimal prediction accuracy and model stability. We are investigating the error breakdown by different model types and days that matter (e.g. holidays, expiries, days-of-the-week etc.).

Each stock exhibits different levels of trends, volatility, and magnitude in their market data. Consequently, we perform stock-specific predictive modelling throughout this study by independently training a variety of window-based predictive models for seven machine learning techniques: ordinary least squares, stepwise regression (i.e. ordinary least squares with sequential feature selection), ridge regression, lasso regression, k-nearest neighbours with arithmetic average, k-nearest neighbours with inverse distance weighting, and support vector regression. For each statistical method, we iterate every stock in our pan-European stock universe consisting of 2,353 stocks, every training window type (i.e. moving/sliding vs. growing) and every window size (i.e. 1-month, 3-month, 6-month, 1-year, 2-year windows). We also train three models for special events (i.e. cross-market holidays, MSCI rebalances and futures expiries) using the entire stock universe, although they are ultimately used to make stock-specific predictions. We fit these models in isolation and aim to determine a performance metric for each method and window type.

Eventually, we shift from a static process to an adaptive process and construct a switching dynamic model, which switches between these models based on the current context (e.g. regular trading day, cross-market holiday, futures expiry, MSCI rebalance, certain day-of-the-week etc.). The proposed model is a virtually switching model as it does not switch per se. We are post-processing the model performance and investigate the performance metrics by breaking down the errors by: day-of-the-week, cross-market holidays, futures expiries, MSCI rebalances etc. This leads to the metamodel, which is a stock-specific out-of-sample model that selects the best initial stock-specific model on a 1-month and a 3-month rolling window basis, depending on the recent performance of the initial stock-specific models that are trained independently of each other.

The rest of the study is structured as follows: section 2 reviews the key findings that led to our model choice in this study (e.g. the volume-price relation asymmetry, the day-of-the-week effect, the expiry day effect, and the cross-market holidays effect) and outlines the methods employed in this analysis; the market and calendar data sets are introduced in section 3; section 4 provides the analysis approach and briefly describes the high performance computing design of this computationally expensive analysis, followed by a methodological introduction of the cross-stock models and the stock-specific models; this is followed by section 5, which presents the main findings of this study, including a performance breakdown of the models, and introduces the switching model and the out-of-sample stock-specific metamodel; eventually, section 6 provides a conclusion of this analysis and discusses the obtained results.

2. Background

Previous studies provided empirical evidence for the volume-price relation and its asymmetry, and the existence of the day-of-the-week effect, the expiry day effect and the so-called 'cross-market holiday' effect in relation with trading volume. These findings are summarised below and are followed by a review of the statistical methods employed in this analysis.

2.1. Volume-Price Relation and Asymmetry

The price-volume relation is of great importance for this study as most of the behavioural literature focuses on the impact of certain anomalies on price returns, while trading volume is the main focus of this study. Price changes represent the market response to new information, whereas the trading volume indicates the level of information disagreement among investors (Beaver, 1968). Although the literature on a potential relation between price changes and volume is far from homogenous, there is a large proportion confirming a positive correlation between trading volume and price changes (Harris & Raviv, 1993) (Hong & Stein, 2007). Batrinca et al. (2016) provided empirical evidence that trading volume is correlated with historical price indicators (i.e. intraday range and intraday return for the previous day, and overnight return for the previous night, which acts as a proxy for the opening auction volume, i.e. more recent information) and that volume exhibits autoregression, where we employed lagged time series volume data (i.e. raw past observations) and also smoothed lagged time series (i.e. moving average of past observations, which acts as a low-pass filter effect in the data).

The formulae for the intraday return, intraday range and overnight return are outlined below, where t_0 is the day for which we predict the trading volume and t_{-1} is the previous trading day, whose price and volume information is available.

$$Intraday\ return\ log\ ratio = log\frac{close_{t_{-1}}}{open_{t_{-1}}} \tag{2.1}$$

$$Intraday\ range\ log\ ratio = log\frac{high_{t_{-1}}}{low_{t_{-1}}} \tag{2.2}$$

$$Overnight\ return\ log\ ratio = \frac{log\frac{open_{t_0}}{close_{t_{-1}}}}{\#\ intervening\ nights} \tag{2.3}$$

In general, there are two key representations of the volume-price relation, where trading volume is positive correlated either with the magnitude (i.e. absolute value) of the price change (Assogbavi & Osagie, 2006), i.e. $|\Delta p|$, or with the price change per se (i.e. the raw value of the price change), i.e. Δp (Karpoff, 1987) (Ying, 1966). The asymmetric relation in the latter representation exhibits a volume/price change ratio that is different in magnitude for upticks than for downticks. Equation (2.5) shows the levels of volume based on the sign of the price change, compared to the symmetric model in Equation (2.4).

$$Symmetry: (v_t|\Delta p_t^+) = (v_t|\Delta p_t^-) \tag{2.4}$$

$$Asymmetry: (v_t|\Delta p_t^+) > (v_t|\Delta p_t^-)\ or\ (v_t|\Delta p_t^+) < (v_t|\Delta p_t^-) \tag{2.5}$$

Batrinca et al. (2016) provided empirical evidence for the price-volume relation asymmetry, which was exhibited in over 70% of the analysed European stocks; there is a moderate overnight asymmetry, which is almost evenly distributed, and a more salient intraday asymmetry (in approximately 60% of the stocks).

2.2. The Day-of-the-Week Effect

The day-of-the-week effect consists of certain trends associated with a particular day-of-the-week. The most broadly studied day-of-the-week effect is the weekend effect (French, 1980) (Gibbons & Hess, 1981) (Jaffe & Westerfield, 1985) (Pettengill, 2003) (Cross, 1973) (Dubois & Louvet, 1996) (Harris, 1986) (Abraham & Ikenberry, 1994), or Monday effect, where the closing price on Monday is lower than the closing price of the previous Friday. These results are intriguing as they are opposite to the expectation of higher returns on Monday, as its returns reflect three consecutive days. The weekend effect has been widely documented in conjunction with price changes. There are very few studies investigating the relation between the day-of-the-week effect and trading volume. For example, Berument and Kiymaz (2001) found day-of-the-week anomalies in both returns and volatility, with the highest volatility on Friday and the lowest on Wednesday, while Lakonishok and Maberly (1990) found a relative increase in the trading activity of individuals on Mondays.

Batrinca et al. (2016) reported a clear improvement of the trading volume prediction model when adding the day-of-the-week features. The indicator variable for Monday improves the model in more than 75% of the cases, having predominantly negative coefficients, despite the fact that we divide the overnight return by the number of intervening nights, which suggests that the negative coefficient for Monday is not a corrective factor and that there is simply less activity on Mondays. Fridays improve the volume model in 45% of the stocks and their coefficients are surprisingly mostly negative, even if the traditional definition of the weekend effect states that the Friday volume and prices are usually higher than those of the following Monday.

2.3. The Expiry Day Effect

The expiry day effect exhibits higher trading volume and abnormal volatility around the close on expiry days for futures and options (Stoll & Whaley, 1997) (Sukumar & Cimino, 2012) (Chow, et al., 2003) (Sadath & Kamaiah, 2011) (Pope & Yadav, 1992) (Vipul, 2005) (Chiang, 2009), and for MSCI quarterly reviews (Chakrabarti, et al., 2005).

Following these findings, Batrinca et al. (2016) further analysed the effect of periodical events on the trading volume, while investigating the stock index futures expiries and MSCI quarterly index reviews in the pan-European markets. The stock index futures expiries occur on the third Friday of each expiry month or on the previous trading day in case that Friday is a bank holiday. The futures contracts are traded either quarterly (i.e. March, June, September and December) or monthly. The indices of Morgan Stanley Capital International (MSCI) are updated quarterly in order to reflect the up-to-date state of the financial markets. The constituent list of these indices changes close to the last trading day of the four rebalancing months: February, May, August and November. We reported the existence of the futures expiry effect and the MSCI rebalance effect, both leading to a surge in trading volume for their index constituents. The trading volume increases significantly during the four days in the run-up to the expiry, lasts two days after the futures expiry, and then returns to normal levels of trading activity starting on the third trading day after the expiry day. The MSCI rebalances exhibit a similar trend, causing surges in the trading volume on the day before the the review day and on the effective rebalance date. We discriminated between these two instances of the expiry day effect and the Friday and end-of-month effects and concluded that the futures expiry effect is essentially causing the so-called Friday effect. However, we could not find enough evidence that the MSCI quarterly reviews could drive the anecdotal end-of-month effect; the trading volumes on the four months with MSCI quarterly reviews are significantly different from those on the adjacent months, but their magnitude is not sufficiently large in order to explain the end-of-month effect throughout the entire year.

2.4. The Cross-Market Holiday Effect

In a previous study, Batrinca et al. (2016) coined the term ‘cross-market holiday effect’, which refers to the anecdotal evidence of lower volumes in a particular country when one or more external markets are not trading. There are only a couple of studies investigating this effect although they focus mainly on the subduing effect of the US holidays on other markets, such as Canada (Cheung & Kwan, 1992) and Europe (Casado, et al., 2013). We documented a salient cross-market holiday effect when a dominant market is on holiday or when most of the European markets are shut. Since the UK is Europe's largest market, we examined whether it is actually the Monday effect that drives down the volumes, as most of the bank holidays fall on a Monday in the UK. However, we reported strong evidence that the Mondays with at least one cross-market holiday have significantly lower volumes that the other Mondays.

Throughout the previous in-sample analyses on the day-of-week, expiry day and cross-market holiday effects, strong evidence of volume autoregression is observed (Batrinca, et al., 2016). Given the results of these previous independent studies, we aim to integrate their findings in an out-of-sample study. Here, we aim to build a virtually adaptive model, which fits a number of models in parallel and switches from one underlying model to another, by taking into account the event dates (e.g. futures expiry, MSCI rebalance, certain day-of-the-week etc.) when we expect the markets to behave significantly different. We also raise additional questions on the optimal training window and the appropriate methodology. We are empirically testing a number of statistical methods in order to understand how the performance of each method is affected and to explore the relationship between trading volume and event dates in a predictive framework.

2.5. Methodology Review

In this section, we review the basic principles for the supervised learning models that are employed in this study. There are seven different statistical methods that are fit simultaneously and independently in order to predict the one-step ahead trading volume. We start with the ordinary least squares (OLS); it is the most basic model and estimates the variable coefficients of a linear regression model by minimising the sum of the squared distances between the predicted values and the observed values.

Feature selection can be applied after a model is fit using OLS, by performing stepwise regression. This can be achieved through forward selection, backward elimination, or bidirectional elimination. We chose forward selection for the second method of this study (i.e. stepwise regression), which adds new variables having p-values that are less than a given improvement measure. We start from a reduced model consisting of the intercept, the lagged volumes, and the smoothed lagged volumes, allowing the model to pick the most informative price log-ratio and day-of-the-week features. The rationale for using forward selection is driven by the design of the day-of-the-week categorical variable as five dummy variables. Generally, a categorical variable having n values is encoded as $n-1$ dummy variables, although in this study, the day-of-the-week dummy variables are mutually exclusive since the aim is to perform feature selection and extract the variables with the highest statistical significance for volume prediction and this is conducted in a feature selection framework. We preferred forward selection to backward elimination because of the potential collinearity problems; adding a collinear variable could make matrix inversion impossible when determining the optimal beta.

The next two techniques employ regression shrinkage methods, namely ridge regression and lasso regression. Linear regression relies on the independence of the model variables and therefore the matrix $(X^T X)^{-1}$ becomes close to singular when the design matrix X has columns that exhibit an approximate linear dependence. As a result, the least squares estimate shown in Equation (2.6) produces a high variance because of its sensitivity to random errors in the observed response variable y.

$$\hat{\beta} = (X^T X)^{-1} X^T y \tag{2.6}$$

Ridge regression, or L$_2$ regularisation, addresses the problem of multicollinearity by estimating the regression coefficients using Equation (2.7), where λ is the ridge parameter and I is the identity matrix. This method introduces bias, but reduces the variance of the coefficient estimates, producing a lower mean squared error (MSE) compared to the least squares estimates. We start by identifying the optimal value for λ (i.e. the ridge parameter) that minimises the cross-validation error, by using a two-section search consisting of grid search and followed by the bisection method (also known as binary search). The grid search traverses 21 consecutive values of λ in logarithmic space, from -10 to 10 and cross-validates the data set for each λ. The value with the minimum average MSE across the grid search is then passed to the bisection method, whose initial left and right points are calculated as $\lambda - 1$ and $\lambda + 1$, respectively, which are also expressed in logarithmic space. The bisection method runs until at least one of the following three tolerance criteria is not met anymore: minimum delta (i.e. minimum change in λ) = 0.1%, minimum error change = $10^{-11}\%$, and maximum number of iterations = 20. The ridge coefficient estimates are restored to the original scale of the data. This transformation also computes the parameter for the constant term (or intercept) and provides a model that is more useful for making predictions, unlike a model with standardised coefficients.

$$\hat{\beta} = (X^T X + \lambda I)^{-1} X^T y \tag{2.7}$$

Lasso (Tibshirani, 1996), or L$_1$ regularisation, is another regularisation method that is similar to ridge regression. The main difference is that when the penalty term λ increases, more coefficients are set to zero, whereas ridge regression sets the coefficients close to zero, but not exactly zero. The lasso estimator produces a smaller model with fewer predictors. Based on the resulting model, lasso can be regarded as an alternative to the second methodology described above, i.e. stepwise regression, and other dimensionality reduction techniques. For a nonnegative regularisation parameter λ, lasso solves the regularisation problem in Equation (2.8), where N is the number of observations, y_i represents the response variable for observation i, x_i is the observed data for observation i consisting of a vector of p values that correspond to each predictor, β_0 is a scalar for the intercept coefficient, and β is a p-vector for the other model terms' coefficients.

$$\min_{\beta_0,\beta}\left(\frac{1}{2N}\sum_{i=1}^{N}(y_i-\beta_0-x_i^T\beta)^2+\lambda\sum_{j=1}^{p}|\beta_j|\right) \tag{2.8}$$

We implemented lasso regression in a similar manner to ridge regression. The optimal λ is determined through 10-fold cross-validation using a two-section search (i.e. grid search in logarithmic space between -10 and 10, followed by binary search for the same set of tolerance criteria that we defined for ridge regression). MATLAB's implementation of lasso regression fits the regularised regression without a constant term, although its coefficient is returned in the 'FitInfo.Intercept' variable, and is eventually appended to the coefficient vector corresponding to the model's predictors.

The k-nearest neighbours (kNN) technique is a non-parametric method belonging to the instance-based learning family, which can be used for both classification and regression problems, where the function is only approximated locally. It is memory-based and requires no model to be fit, i.e. it memorises all of the observations and predicts the target variable based on the chosen similarity measure, which is typically a distance function. The most common distance metric for continuous variables is the Euclidean distance shown in Equation (2.9), whereas the Hamming distance, represented in Equation (2.10), is typically used for binary/categorical variables and is calculated as the number of instances where two observations are different.

$$d_{Euclidean}=\sqrt{\sum_{i=1}^{k}(x_i-y_i)^2} \tag{2.9}$$

$$d_{Hamming}=\sum_{i=1}^{k}|x_i-y_i| \tag{2.10}$$

The algorithm retrieves the k memorised examples that are the most similar to the one that is used for the current prediction using an appropriate distance function. The kNN method does not have any costs associated with the learning process as there is no model inferred and, because of this, it is also known as 'lazy learning', as the entire cost of this technique consists of the prediction computation; there are no assumptions about the characteristics of the data, although the lack of any learning costs makes kNN impossible to be interpreted as there is no description of the learnt concepts. Moreover, the accuracy of kNN can be significantly impacted by the presence of noisy or irrelevant features. The basic version of kNN is the 1-nearest neighbour estimate, whose bias if often low, but the variance is high. An interesting property of the nearest-neighbour is that its error rate is never more than twice the minimum achievable error rate of an optimal classifier (Hastie, et al., 2011) (Bishop, 2007).

In order to identify the optimal value of k, we perform 10-fold cross-validation and we pick the value of k that minimises the cross-validation average error. A small value of k means that noise will have a higher impact on the results, whereas a large value of k is computationally expensive and signals a highly non-linear and noisy structure. The number of neighbours can be regarded as a measure of noisiness; for example, 1NN is an indication of clear data. In general, a larger value for *k* is more precise, although the boundaries within the feature space become blurred. A few authors (Duda, et al., 2000) (Hassanat, et al., 2014) suggest an empirical rule-of-thumb, and setting k equal to the square root of the number of instances, $k = n^{1/2}$, as a starting point. We also attempted to apply PCA on the standardised variables in order to remove the correlations before running kNN, but it mainly dealt with the intercept only and did not improve the resulting model.

We begin by standardising each feature of the data set to have mean zero and variance 1, because the variables have different measurement scales and there is also a mixture of continuous and categorical/binary variables (Hastie, et al., 2011). This allows us to use the Euclidean distance as the nearest neighbours' similarity measure.

The following two methods represent slightly different implementations of kNN, which vary in their approach of aggregating the contribution of the identified neighbours. The first approach is kNN with arithmetic mean, which treats all of a point's neighbours equally and computes the prediction as the average of the target variable of the k nearest neighbours, as shown in Equation (2.11). The second approach is kNN with inverse distance weighting, where the neighbours are assigned weights based on their distance from the prediction point, such that the nearer neighbours contribute more to the average compared to the further neighbours. This method assigns a weight to each neighbour, which is equal to the inverse of its distance to the prediction point; this weighted average is illustrated in Equation (2.12). The algorithm finds the *k* nearest observations using the Euclidean distance metric, then calculates the inverse distance weight of each neighbour and normalises the inverse distances such that their sum is equal to one. Finally, the method computes the weighted average of the *k* neighbours using their inverse distance weights. We implemented both methods in order to better understand the data structure. Using the inverse distance weighting could potentially lead to a large number of neighbours being identified, where most of them could have extremely small weights that would not influence the prediction significantly and would simply introduce more noise to the model. If this is the case, a parsimonious model could be identified by using the arithmetic mean and implicitly assigning equal weights to the neighbours.

$$\hat{y} = \frac{1}{k}\sum_{i=1}^{k} y_i \qquad (2.11)$$

$$\hat{y} = \sum_{i=1}^{k} {d'}_i^{-1} \cdot y_i \qquad (2.12)$$

Both techniques begin by identifying the optimal value for k. This is accomplished by performing 10-fold cross-validation grid search for *k* ranging between 1 and 49, with a step size of 1, and between 50 and 100, with a step size of 5. Once the optimal value of *k* is found, the algorithm retrieves the closest *k* neighbours by standardising the training data set (or computing the z-score, such that each variable has unit variance and zero mean), and then standardising the test set using the mean and variance obtained from the training set. Then the predicted value is computed either by the mean of the target variable of the *k* neighbours for the kNN with arithmetic mean method, or by weighing the neighbours taking into account their normalised inverse distances.

The last method is based on support vector machine (SVM) analysis (Cortes & Vapnik, 1995) (Vapnik, 1999), which is a popular method that was traditionally employed for classification; a version of SVM for regression was introduced later (Drucker, et al., 1997) and is called support vector regression (SVR). SVM is non-parametric as it relies on kernel functions. SVM produces non-linear boundaries by creating a linear boundary in a transformed representation of the feature space (Hastie, et al., 2011). SVM maximises the margin around the separating hyperplane and defines the solution in terms of a small subset of training samples, which are called the support vectors, i.e. the training data points that are closest to the decision hyperplane and that are most difficult to classify. SVR produces a model that depends only on a subset of the training data, since the model's cost function ignores the training data points that are close to the model prediction. We implemented the sequential minimal optimisation (SMO) algorithm (Platt, 1998), which does not require a numerical optimisation algorithm or matrix computation and storage, because it divides a very large quadratic programming (QP) optimisation problem into a series of smallest possible QP problems that are solved analytically; this eliminates the need for a time-consuming numerical optimisation as an inner loop. The SMO algorithm is fast, easy to implement, and provides better scaling properties. The algorithm also flags the day-of-the-week features as categorical predictors. The SVR models in this study use the Gaussian kernel function. Keerthi and Lin (2003) proved that the linear kernel is a degenerate version of the Gaussian kernel, also called radial basis function (RBF), and therefore the linear kernel would never have a better accuracy than the Gaussian kernel.

Our SVR method implements the linear epsilon-insensitive SVM (ε-SVM) regression, which is also called the L_1 loss. By using the predictor variables and the observed response variables, the goal of ε-SVM is to identify a function $f(x)$ such that its deviation from y_n is no greater than ε for each training point and is as flat as possible (MathWorks, 2016). There are two main formulations for the optimisation problem: the primal formula and the dual formula. The primal formula consists of a convex optimisation problem, where it is possible that there is no function that satisfies the constraints for all points. This issue is overcome by introducing slack variables, which help deal with infeasible constraints and lead to the objective function, also known as the primal formula. The primal formula includes the box constant, which acts as a regularisation method in order to prevent overfitting; this imposes a penalty on all of the observations lying outside the ε margin and determines the trade-off between the flatness of the function $f(x)$ and its tolerance. The SVR loss is calculated based on the distance between the observed target variable y and the ε boundary. The dual formula provides a computationally simpler solution to the primal formula; it employs Lagrange multipliers in order to transform the optimisation problem into a form that can be solved analytically. The optimal values of these two problem formulations are not necessarily equal and their difference is known as the duality gap. The solution of the dual problem is used exclusively when the problem is convex and meets a constraint qualification condition.

3. Data Set

We compile one of the most comprehensive pan-European data sets, ranging from 1st January 2000 to 10th May 2015. It consists of over 7 million observations of daily market data for 2,353 stocks, 3,039 bank holidays for 22 countries, 1,042 stock index futures expiries for 7 indices, and 49 MSCI quarterly review dates, along with a historical log of 1,420 leavers and joiners for the investigated futures and MSCI indices. A great effort has been put into collecting, cleansing and processing the calendar data set due to the lack of a comprehensive database of bank holidays for financial markets.

3.1. Market Data

The market data contains daily observations consisting of the opening, closing, low and high prices and the trading volume for the constituents of the 31 most important European indices. The data set was retrieved from Thomson Reuters and was further processed. We compute the consolidated trading volume for each stock by retrieving the corresponding trading volume across the main European multilateral trading facilities (MTFs), i.e. BATS, CHI-X and Turquoise, and adding the MTF volume to the trading volume of the primary exchange. The resulting consolidated volume is used across this study in order to better reflect the true liquidity of a stock. The analysis discards the stocks with less than 100 trading days. South Africa was included in the analysis due to its close ties with the European financial markets. The processed market data covers 21 European countries and Table 3.1 outlines the number of stocks and their daily observations for each country.

Table 3.1
Stock universe – Breakdown by country.

Country Name	Country Code	Number of Stocks	Number of Observations
Austria	AT	32	98,179
Belgium	BE	62	205,414
Czech Republic	CZ	5	14,491
Denmark	DK	43	144,352
Finland	FI	130	390,209
France	FR	346	1,117,220
Germany	DE	176	539,142
Greece	GR	61	209,103
Hungary	HU	4	15,311
(Republic of) Ireland	IE	43	100,910
Italy	IT	111	330,609
Netherlands	NL	46	157,156
Norway	NO	69	172,562
Poland	PL	65	162,509
Portugal	PT	18	53,449
South Africa	ZA	42	139,568
Spain	ES	61	179,410
Sweden	SE	158	462,935
Switzerland	CH	104	339,998
Turkey	TR	130	412,273
United Kingdom	GB	647	1,952,265

3.2. Calendar Data

The market data is augmented by a comprehensive list of event dates, which can be classified as bank holidays and expiry days (i.e. stock index futures expiries and MSCI rebalances). These special events are expected to impact on the normal state of the financial markets and cause non-stationarity in trading volume.

Bank Holidays

The data set for bank holidays is customised specifically for the financial markets and can be different in certain instances from the official national public holidays for a given country: when an exchange venue is owned by a company which is based in another country (e.g. Euronext) and enforces a different trading calendar, when a trading venue is located in a region with additional holidays, or when unforeseeable events occur (e.g. Hurricane Sandy, 11th September Terrorist Attacks etc.). This calendar is an accurate reflection of the trading state of the US and the pan-European exchanges, consisting of 22 countries. The United States of America was included in the data set since it is a dominant financial market, whose magnitude could potentially influence the European liquidity. The non-trading calendar was meticulously compiled from scratch and multiple sources (e.g. the trading calendar on the exchanges' websites and public holidays from www.timeanddate.com) were used to make decisions on the final outcome. These were double-checked with the empirical trading calendar resulting from the market data, which truly proved whether an exchange has been trading on a particular day. The accuracy of this calendar was vital to perform a cross-market holiday model and had to be manually constructed because there was no such trading calendar available; there are very few such calendars, although their information is either incomplete or they contain conflicting information.

There are also country-specific characteristics for generating the public holidays calendar. For example, if a public holiday falls on a weekend, different countries substitute it with the previous trading day (e.g. New Year's Eve in Austria and Belgium), with the following day, or do not substitute it at all. Additional 'bridge' holidays can be observed in particular countries (e.g. Hungary and Poland), when a holiday falls on a Tuesday or on a Thursday, resulting in four-day weekends.

An illustrative example of the difference between the official public holidays and the non-trading calendar is on 1st May in the Netherlands, where the financial markets are shut despite the fact that 1st May is not a bank holiday. This is observed after the Amsterdam stock exchanged merged with the Brussels and Paris stock exchanges, in order to form the Euronext group. Similarly, the Belgian, Portuguese and French trading calendars changed after their main trading exchanges joined Euronext and therefore the public holidays between 1st May and Christmas Eve became regular trading days.

Expiry Days

The expiry day calendar incorporates periodic trading events which could be positively correlated with the trading volume, and consists of the futures expiries for seven liquid indices and the MSCI quarterly review for the MSCI International Pan Euro Price Index. By using the most liquid indices in Europe, this expiry calendar is an accurate representation of the main expiry dates in the European markets.

We retrieved the up-to-date constituents for these indices as of 11th May 2015, which represent the 'current constituents'. In order to create an accurate representation of the expiring indices at a given point in our analysis timeframe, we constructed a historical list of additions and eliminations for each index, which allowed the generation of a snapshot of a stock's constituent stocks. Table 3.2 outlines the number of constituents for each index, for both futures expiries and MSCI rebalances, where the 'historical constituents' column represents the number of previous stocks that were part of the constituent list of a given index before 11th May 2015, but which were subsequently eliminated, such that they are not a constituent anymore on 11th May 2015.

Table 3.2
Market data European indices for the futures expiry analysis and MSCI rebalance analysis.

Analysis Type	Index Name	Current Constituents	Historical Constituents	Location
Futures expiry	Amsterdam Exchanges Index	25	37	Netherlands
	CAC 40 Index	40	54	France
	FTSE MIB Index	40	51	Italy
	FTSE 100 Index	100	149	United Kingdom
	Deutsche Boerse DAX Index	30	37	Germany
	IBEX 35 Index	35	44	Spain
	OMX Stockholm 30 Index	30	33	Sweden
MSCI rebalance	MSCI International Pan Euro Price Index EUR Real Time	204	338	Europe

Stock Index Futures Expiries

There are 32,408 observations of stock index futures expiries for seven indices, whose expiries occur either monthly or quarterly (i.e. December, March, June and September) as follows:

- Monthly: CAC 40 Index Futures, FTSE MIB Index Futures, IBEX 35 Index Futures, Amsterdam Exchanges (AEX) Index Futures, and OMX Stockholm 30 (OMXS30);
- Quarterly: FTSE 100 Index Futures, and DAX 30 Index Futures.

The expiry occurs on the third Friday of the expiry month, or on the previous trading day when the third Friday is a non-trading day. The futures contract specifications were retrieved from Euronext (AEX and CAC 40), Eurex Exchange (DAX 30), London Stock Exchange (FTSE 100), Borsa Italiana (FTSE MIB), Bolsas y Mercados Españoles (IBEX 35) and NASDAQ OMX (OMXS30), in order to verify the expiry specifications for each index.

MSCI Quarterly Reviews

The MSCI rebalances have 10,298 observations across 16 countries: Austria, Belgium, Switzerland, Germany, Denmark, Spain, Finland, France, United Kingdom, Greece, (Republic of) Ireland, Italy, Netherlands, Norway, Portugal, and Sweden. Each stock's country represents the country where that stock is trading, e.g. the United Kingdom is defined as a Spanish stock's country if this stock is trading on the London Stock Exchange.

In general, the MSCI quarterly reviews are implemented on the last trading day of the February, May, August, and November quarterly cycle, although there are a few exceptions when the MSCI rebalance falls a few days before the end of the month. The MSCI quarterly review dates were double-checked with the quarterly index review documents from www.msci.com.

4. Predictive Modelling

We build a 1-step ahead out-of-sample model for predicting the trading volume, while fitting different supervised learning methods and examining event dates.

4.1. Analysis Approach

The methodological approach for constructing the predictive model is described in this section. For a given stock, these models predict the volume of the next day (i.e. the target date) based on past observations, employing a variety of machine learning methods and training window types. All of the models are fit with a constant term.

There are three cross-stock models for event dates (i.e. cross-market holidays, stock index futures expiries, and MSCI quarterly reviews), which are fit using normalised data from all of the relevant stocks. In the case of special events, very few training observations would be available for an individual stock, hence the necessity of aggregating the training points for multiple stocks. However, after learning the model on the normalised data set, the volume for each stock is predicted individually, by using the stock-specific benchmark volume that was used for normalising the stock's past volumes. The feature set of these cross-stock models includes 20 lagged volumes for each stock's observation, which are normalised by dividing them by their median, in order to remove any differences in magnitude across our stock universe.

Besides these three cross-stock models, there are seven stock-specific models, which are fit using different types of supervised learning methods: OLS, stepwise regression, ridge regression, lasso regression, kNN with arithmetic mean, kNN with inverse distance weighting, and SVR. We define an iteration as a fit model for every combination of stock, target date, learning method, and window type. For each iteration, these stock-specific models follow a similar training routine, starting by defining the 10-fold stratified cross-validation (CV) partitions from the beginning of the analysis, in order to conduct the entire iteration analysis on the same data partitions (e.g. when cross-validating potential values for method-specific parameters such as λ or k). The CV splits the data into 10 equally-sized partitions, while ensuring these are stratified by the binary indicator variables (i.e. the day-of-the-week binary features), such that these features are evenly distributed across the folds; its aim is to minimise the average mean squared error (MSE) throughout the 10 folds. The models can potentially contain 15 raw lagged volumes (i.e. autoregressive past observations) and 14 smoothed lagged volumes (i.e. moving average past observations), in order to explain the trading volume using recent time series. The iteration analysis identifies the optimal orders for the raw lagged volumes (or 'volume lags') and the smoothed lagged volumes (or 'volume windows'). It starts by fitting a linear regression for the lowest orders (i.e. volume lag 1, or volume window 2), then it increments the order by one, fits the second model, and compares the CV average MSE for these two models. If the higher order model performs better, the process is incrementally repeated for the next pair of orders (up to order 15), until the optimal order has been found, either when the higher order model has a larger error (and therefore the current model pair's lower order becomes the optimal one), or when the order reaches the maximum limit of 15. This incremental comparison of nested models is conducted independently for the volume lags and the volume windows.

When kNN, ridge regression or lasso regression are employed, the model proceeds to parameter calibration and runs grid search for k and performs a two-section search (i.e. grid search and binary search) for λ. These searches perform 10-fold cross-validation for each value. Then, all of the models proceed to feature construction and model training. The iteration analysis ends by testing the learnt model, i.e. computing the 1-step ahead prediction for the target date.

Training Windows

Each model is trained using two approaches: moving window and growing window. This helps understand whether a model relies only on recent data or whether it improves when more and more data points are used for training the model. These two approaches differ in the size of past observations when learning a model. When iterating the target dates of a stock for which predictions are made, the moving window approach trains the model using a fixed number n of past observations, starting from the most recent data point (i.e. the observation occurring right before the prediction 'unseen' data point) and going backward until n points are accumulated. Throughout the next iterations, the moving window gradually adds a newer observation and drops the oldest observation, whereas the growing window approach adds a newer observation without discarding any other observations. Therefore, the number of observations on the kth iteration of a model is n for the moving window and $n + k - 1$ for the growing window.

There is a discrete number of sliding window sizes, whose representations are marked in brackets and are used when outlining the model results for this study: 1 month ('MW_1M'), 3 months ('MW_3M'), 6 months ('MW_6M'), 1 year ('MW_1Y'), and 2 years ('MW_2Y').

The growing window starts with a training size that is equal to the largest moving window size, i.e. 2 years, and is represented by 'GW'.

Each stock-specific learning method is trained using the five moving window types and the growing window, whereas the cross-stock models are fit using only the two largest moving window sizes (i.e. 'MW_1Y' and 'MW_2Y') and the growing window. The rationale of using only the 1-year and 2-year moving windows is driven by the significantly lower number of observations in the case of event dates (i.e. cross-market holidays, futures expiries, and MSCI rebalances).

For each training window iteration, the models are re-trained based only on the data available in that particular training window. The window sizes have been translated into a certain number of trading days, such that a constant number of observations are used to train the models throughout the different window iterations and stocks. There are 2,937 holidays for 22 countries whose market data is investigated, throughout 15 full years, between 1st January 2000 and 31st December 2014. This period covers exactly 15 years, or 5,479 days including weekends, or 3,913 days excluding weekends. On average, there are 252 trading days per year for each country, which are derived from the difference between the total number of business days and the average number of holidays per country, which is then divided by the number of years: $(3913 - 2937/22)/15 = 251.97$. Therefore, the fixed-length moving windows are defined in trading days as follows: 21 days for 1 month, 63 days for 3 months, 126 days for 6 months, 252 days for 1 year, and 504 days for 2 years. The year 2015 was excluded from this averaging because our data set includes observations until 10th May 2015 and therefore this year has incomplete data.

Out of the 2,353 pan-European stocks, there are 163 stocks (or 6.93%) whose number of observations is less than 504 (corresponding to the 2-year window). As for the remaining 2,190 stocks with available data spreading on over 2 years, there are 26 stocks with less than 100 days outside the 2-year period, 150 stocks with more than 100 days and less than 1,000 days, and 2,014 stocks with over 1,000 days of observations outside the 2-year period.

Cluster Job Management

Given the tremendous number of iterations and runtime required by the stock-specific models, we ran these models on two distinct computer clusters for high performance computing, which operate on the Sun Grid Engine grid computing system.

The stock-specific total runtime was 11,878 days (or 33 years), excluding the queuing times associated with each job, which tended to reach even several days during peak times. The stock-specific models have been split into jobs of maximum 1,000 iterations (i.e. 1,000 consecutive target dates for a given stock). For example, a stock with 3,683 observations running a 2-year moving (or growing) window, needs $3683 - 504 = 3179$ iterations to traverse all of the target dates for 1-step ahead volume prediction; therefore, there are 4 jobs for this stock (broken down into 3 jobs of 1,000 iterations and another job of 179 iterations), for a particular learning method. Table 4.1 outlines the total runtime for each method (across all of the stocks and window types) and for each window type (across all of the stocks and learning methods), along with the corresponding number of jobs and target dates.

Table 4.1
The distribution of runtime and number of iterations/target dates by method and window type.

Breakdown Item	Item Name	Runtime (Days)	Jobs	Target Dates
Method	OLS	554.64	45,990	39,604,267
	Stepwise regression	819.76	45,990	39,604,267
	Ridge regression	767.18	45,990	39,604,267
	Lasso regression	6,014.76	45,990	39,604,267
	kNN (arithmetic mean)	1,791.22	45,990	39,604,267
	kNN (inverse distance)	1,361.45	45,990	39,604,267
	SVR	569.41	45,990	39,604,267
Window type	Moving window, 1 month	938.53	55,622	49,802,970
	Moving window, 3 months	993.51	55,391	49,111,188
	Moving window, 6 months	1,142.35	54,978	48,074,677
	Moving window, 1 year	1,196.28	53,571	46,029,606
	Moving window, 2 years	1,494.24	51,184	42,105,714
	Growing window	6,113.52	51,184	42,105,714

Lasso and kNN are computationally expensive, mainly because lasso performs a two-section search (i.e. grid search and bisection method) and deals with a large number of features (up to 40 variables) when regularising their coefficients and sets some to zero, while kNN is memory-based and requires heavy resources when finding the *k* nearest neighbours for a test point. The runtime for the various window sizes is larger when the window grows in size and is significantly larger for the growing window approach.

4.2. Cross-Stock Models

We investigated the effect of event dates on trading volume, focusing on cross-market holidays, stock index futures expiries, and MSCI quarterly reviews. The sparsity of these observations determined the models to be trained on cross-stock data. Since stocks exhibit different volume and price magnitudes, we normalised the past observations of trading volume and aggregated the data for the entire stock universe. Even after aggregating, the number of observations was significantly less than in the case of stock-specific models; there are 2,904 target dates and predicting their volume had a runtime of 15 days. Each model corresponds to only one learning method. The cross-market holiday model employs ridge regression, whereas the futures expiries and the MSCI rebalances are fit using OLS.

Unlike the stock-specific models where the target variable consists of the logarithmic consolidated volume, the cross-stock models employ the 'relative volume' as the target variable. Equation (4.1) shows the formula for the relative volume, which is determined by the log-ratio between the consolidated volume on the target date (also called 'event date' or 'special date') and the stock-specific benchmark volume. This benchmark is computed as the median of the trading volumes of the 20 trading days prior to the target date (i.e. the futures expiry, MSCI rebalance, or cross-market holiday). The median was selected among other measures of central tendency (e.g. geometric mean or arithmetic mean) because it was the most robust to the outliers in our data set. By dividing a stock's target date volume by the benchmark volume, we get a normalised value for the trading volume, which works well across our stock universe. This normalisation, consisting in the identification of observations from multiple stocks that have a common target date, was necessary as these event dates are periodic, but sparse.

$$V_{rel} = \hat{y} = log\frac{V_{t0}}{V_{benchmark}} = log\frac{V_{t0}}{median(V_{t-lag-1}, V_{t-lag-2}, \dots, V_{t-lag-20})} \tag{4.1}$$

When performing 1-step ahead prediction, these models estimate the relative volume. In order to be able to make stock-specific predictions, the relative volume needs to be converted to a particular stock's logarithmic volume. Essentially, we train the model on the entire stock universe sharing a common event date, but we make stock-specific predictions by transforming the target variable from being stock-agnostic to being stock-specific. Equation (4.2) shows how to calculate the stock-specific volume estimate $\widehat{y'}$ based on the relative volume. We add the benchmark volume to the relative volume, as this is the stock-specific term that customises the volume prediction for a given stock.

$$V_{stock} = \widehat{y'} = V_{rel} + log\, V_{benchmark} \tag{4.2}$$

Cross-Market Holidays

The cross-market holiday model implements ridge regression, which performs a two-section search for each iteration. Ridge regression was appropriate for the cross-market holidays as it addresses the problem of multicollinearity and reduces the coefficient variance. Its predictors consist of the constant term, 20 lagged normalised volumes (i.e. a stock's most recent 20 volumes divided by their median), 21 indicator variables for the trading country, and 22 indicator variables for the holiday country, adding the US on top of the 21 trading countries. The regression line is outlined in Equation (4.3), where β_0 is the constant term, T_i is the indicator variable signalling whether the *i*th country is trading, and H_i indicates whether it is on holiday.

$$\hat{y} = \beta_0 + \sum_{i=1}^{20} \beta_{lag_i} \frac{V_{t-i}}{V_{benchmark}} + \sum_{i=1}^{21} \beta_{country_i} T_i + \sum_{i=1}^{22} \beta_{country_i} H_i \tag{4.3}$$

Stock Index Futures Expiries

The futures expiry model is fit using OLS. Stepwise regression, or more generally feature selection, was not performed based on the previous findings (Batrinca, et al., 2016), where the OLS provided a more stable model across the analysis. The feature set consists of the constant term, 20 lagged normalised volumes and 7 indicator variables corresponding to the futures indices included in this pan-European analysis (i.e. Amsterdam Exchange, CAC 40, FTSE MIB, FTSE, Deutsche Boerse DAX, IBEX 35, and OMX Stockholm 30), showing which expiring index a particular observation is a member of. The model is summarised in Equation (4.4), where E_i indicates whether a particular stock is the constituent of the *i*th index.

$$\hat{y} = \beta_0 + \sum_{i=1}^{20} \beta_{lag_i} \frac{V_{t-i}}{V_{benchmark}} + \sum_{i=1}^{7} \beta_{index_i} E_i \tag{4.4}$$

MSCI Quarterly Reviews

The MSCI rebalance model is similar to the futures expiry model and is fit using OLS due to the same considerations. It is modelled for the MSCI International Pan Euro Price Index, which covers 204 stocks from 16 countries: Austria, Belgium, Switzerland, Germany, Denmark, Spain, Finland, France, United Kingdom, Greece, (Republic of) Ireland, Italy, Netherlands, Norway, Portugal, and Sweden. The model terms include the intercept, 20 lagged normalised volumes and 16 indicator variables for the trading country of each stock, i.e. the exchange country where the stock is trading; these are outlined in (4.5), where C_i represents the indicator variable for the *i*th country included on the MSCI pan-European index.

$$\hat{y} = \beta_0 + \sum_{i=1}^{20} \beta_{lag_i} \frac{V_{t-i}}{V_{benchmark}} + \sum_{i=1}^{16} \beta_{country_i} C_i \qquad (4.5)$$

4.3. Stock-Specific Models

There are seven stock-specific models employing different supervised learning techniques and they all begin from the model function in Equation (4.6). The initial feature set includes the constant term, 15 volume (autoregressive) lags, 14 volume (moving average) windows, 5 price metrics P_i that are trained using the opening, closing, low and high prices of the previous trading day (i.e. $t-1$) and the opening price of the target day (i.e. t_0) in the case of the overnight return, and finally five indicator variables corresponding to each business day, denoted by DOW_i, where *i* ranges from 1 to 5 (i.e. Monday to Friday). The target variable of these models is a particular stock's estimated logarithmic volume for the next trading day (i.e. t_0).

$$\hat{y} = \beta_0 + \sum_{i=1}^{15} \beta_{lag_i} volLag_{t-i} + \sum_{i=2}^{15} \beta_{lag_i} volWin_{t-i} + \sum_{i=1}^{5} \beta_{pr_i} P_{i,t-1} + \sum_{i=1}^{5} \beta_{dow_i} DOW_i \qquad (4.6)$$

We use three main price log-ratios: intraday range, asymmetric intraday return, and asymmetric overnight return; their formulae are shown in Equation (4.7), Equation (4.8), and Equation (4.9), respectively. The overnight return is divided by the number of intervening nights in order to correct for the additional non-trading day observed throughout weekends and bank holidays. The previous empirical evidence of Batrinca et al. (2016) found that better performance is achieved when splitting the intraday return and overnight return log-ratios at zero, into positive absolute values (denoted by 'absPos', representing the absolute value of positive returns only), and negative absolute values (denoted by 'absNeg', corresponding to the absolute value of negative returns). Consequently, we include the following 5 price metrics in the initial model: intraday range, 'absPos' intraday return, 'absNeg' intraday return, 'absPos' overnight return, and 'absNeg' overnight return.

$$Intraday\ return\ log\ ratio = log \frac{close_{t_{-1}}}{open_{t_{-1}}} \qquad (4.7)$$

$$Intraday\ range\ log\ ratio = log \frac{high_{t_{-1}}}{low_{t_{-1}}} \qquad (4.8)$$

$$Overnight\ return\ log\ ratio = \frac{log \frac{open_{t_0}}{close_{t_{-1}}}}{\#\ intervening\ nights} \qquad (4.9)$$

Given this initial model, the analysis follows the framework described in the Analysis Approach section for each individual iteration (i.e. for each target date, given a particular learning method, a particular stock and a particular window type): partitioning the data for the subsequent stratified 10-fold cross-validation applications and determining the optimal orders for the volume lags and the volume windows, producing a model with potentially less features than the initial model, where 15 volume lags and 14 volume windows were included. Then, if the method is a shrinkage method (i.e.

ridge regression or lasso regression) or kNN, the method optimal parameter is identified using cross-validation on that iteration's training set. Eventually, each of the following methods is applied to this model definition, using the methodology described in the Methodology Review section: OLS, stepwise regression, ridge regression, lasso regression, kNN with arithmetic mean, kNN with inverse distance weighting, and SVR. A particular constraint is applied to stepwise regression, where we force the constant term, the volume lags and volume windows to be kept into the reduced model when performing sequential feature selection.

5. Results

The results of this study are outlined in this section, along with an interpretation of their meaning and implication. We start by investigating the distribution of the volume lags and windows, and then explore the method-specific results, such as the model parameter distribution and feature selection. Next, we provide a performance benchmark for the various models employed in this model, leading to an interpretation of the optimal learning method and training window, and to a breakdown of the model performance by event dates. Based on this performance breakdown by special events, we propose a switching model that virtually adapts from one underlying model to another, based on the current state of the market, which is driven by event dates (e.g. futures expiries, MSCI rebalances, and cross-market holidays) or the current day-of-the-week.

5.1. Contribution of Recent Data: Volume Lags and Windows

For each unique combination of stock, learning method and window type, a certain optimal order for the volume lag and volume window is determined for every target date. Since a stock has different cross-validation partitions across its various models consisting of different learning methods and window types, we report very minor fluctuations in the order distribution of volume lags and volume windows for the same window type across the seven training methods. Therefore, we aggregated the order values across the seven models, grouped by window type.

Table 5.1 outlines the descriptive statistics for each window for volume lags and volume windows. We observe a correlation between the size of the training window and the mean and median of the volume lag/window orders. This suggests that the larger the training set, the more relevant past volumes tend to become in fitting an accurate prediction model. This confirms that trading volume is autoregressive and that past observations are meaningful if a substantial training set is available to learn the model.

Table 5.1
Descriptive statistics for the orders of the volume lag and the volume window, grouped by window type.

Past Volume Type	Window Type	Min	Max	Mean	Median	Standard Deviation
Volume lag	MW_1M	1	15	1.22	1	0.51
	MW_3M	1	13	1.33	1	0.63
	MW_6M	1	11	1.65	1	0.89
	MW_1Y	1	13	2.35	2	1.23
	MW_2Y	1	15	3.56	3	1.57
	GW	1	15	7.67	7	3.22
Volume window	MW_1M	2	15	2.24	2	0.52
	MW_3M	2	10	2.24	2	0.54
	MW_6M	2	12	2.34	2	0.67
	MW_1Y	2	15	2.65	2	0.98
	MW_2Y	2	15	3.50	3	1.50
	GW	2	15	7.70	7	3.28

In Figure 5.1, we can visualise how the highly positive skewness from Panel A, which corresponds to the smallest training window (i.e. 1-month moving window) gradually transforms into a relatively symmetrical distribution in Panel F, where the growing window approach trains the model using a variety of lag orders, including the high orders towards 15.

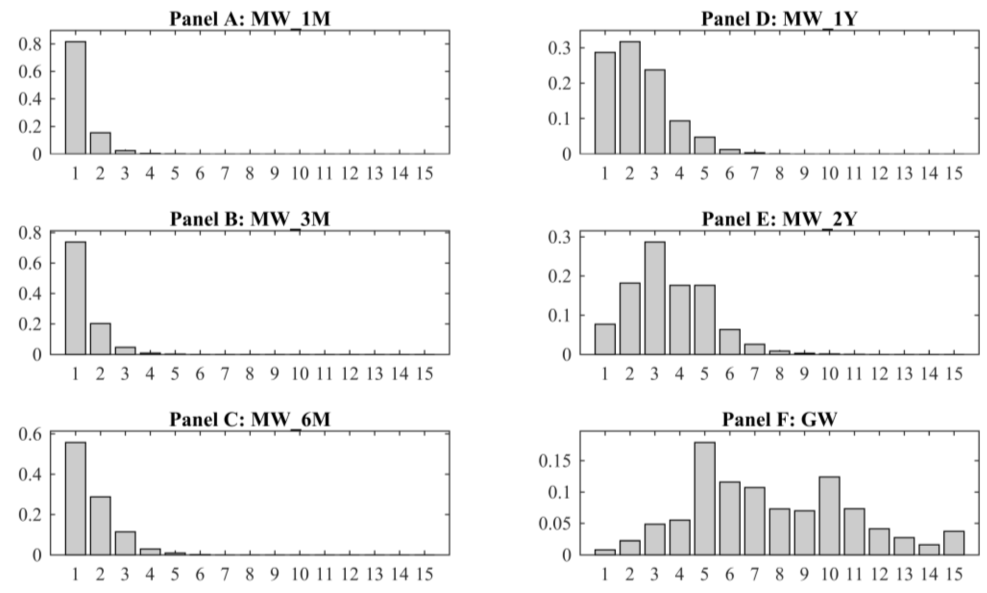

Figure 5.1: Distribution of the volume lag orders across the six different window types.

The moving average-based volume windows in Figure 5.2 exhibit a similar pattern and the volume window orders are positively skewed for the smallest training window (i.e. 1-month moving window in Panel A). The positive skewness decreases once the training window is extended, and becomes rather symmetrical for the largest training window in Panel F (i.e. the growing window). The growing window starts from an initial window size of 2 years, whose distribution is outlined in Panel E. However, the larger the window becomes, the less the order distribution is positively skewed, exhibiting a negatively skewed distribution for the largest window sizes of the growing window, which ultimately yields the relatively symmetrical distribution in Panel F.

The shift in the order distribution from smaller to larger training windows provides evidence that recent volume data contributes to the prediction accuracy and that the amount of meaningful recent data (in the form of lag and window orders) increases with the number of observations in the training window.

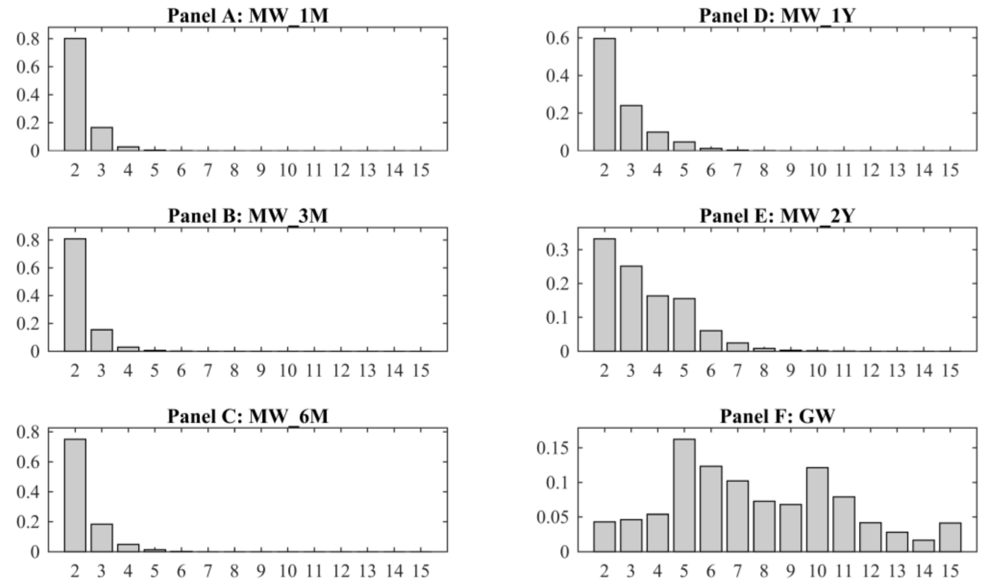

Figure 5.2: Distribution of the volume window orders for different window types.

5.2. Method-Specific Parameters

We performed grid search between 1 and 50, with a unit-sized step, and between 50 and 100, with a step size of 5, in order to identify the optimal value of *k* for the two kNN models, while conducting a two-section search for identifying the optimal value of λ for the two regularisation methods in this analysis. At each step, 10-fold stratified cross-validation was performed to validate the model performance. Below, we outline the distributions and patterns of these two method-specific parameters, i.e. *k* and λ.

k-Nearest Neighbours

There are two models implementing kNN, one that treats neighbours equally and uses the arithmetic mean to determine the target variable, and one that penalises the distance between the test point and its neighbours through inverse distance weighting. Table 5.2 includes the descriptive statistics for the values of *k* for each window type of the two kNN models, i.e. kNN with arithmetic mean and kNN with inverse distance weighting, for the entire stock universe. We observe similar results for the distribution of *k* across these two models, although the inverse distance weighting approach tends to have slightly higher values of central tendency, having the mean and median with almost 3 neighbours more than the arithmetic mean approach. We report that the mean and median increase with the window size, especially when comparing the 2-year moving window with the growing window, as their initial iteration is identical, confirming that the market data has a highly noisy structure. The value of *k* for the 1-month moving window reaches is less than 18 as it contains 21 trading days and similarly the 3-month moving window has less than 56 neighbours as it contains 63 trading days in total.

However, *k* reaches the maximum number of 100 neighbours once the training window is at least 6 months long; we did not allow for more than 100 neighbours in order to avoid over-smoothing and eliminating important properties in the data distribution. Although we expected the inverse distance weighting approach to have a significantly higher number of neighbours on average potentially because it could assign very low weights to a high number of neighbours with a possible blurring effect, the difference is not very conspicuous. We conclude that the kNN with arithmetic average approach produces a model that is slightly more parsimonious that the one yielded by the inverse distance weighting, although their overall parameter distribution is rather similar. Their performance is discussed in a subsequent section.

Table 5.2
Descriptive statistics for the values of k for the 6 different window types of the two kNN models.

kNN Approach	Window Type	Observations	Min	Max	Mean	Median	Standard Deviation
Arithmetic Mean	MW_1M	7,111,213	1	18	10.82	11	5.25
	MW_3M	7,012,429	1	55	23.26	20	13.92
	MW_6M	6,864,419	1	100	32.43	29	22.29
	MW_1Y	6,572,392	1	100	38.17	32	27.68
	MW_2Y	6,012,088	2	100	39.42	27	29.51
	GW	6,012,088	3	100	48.23	41	26.47
Inverse Distance Weighting	MW_1M	7,111,213	1	18	11.74	12	5.23
	MW_3M	7,012,429	1	55	26.20	24	14.70
	MW_6M	6,864,419	1	100	36.44	32	24.23
	MW_1Y	6,572,392	1	100	41.72	37	28.99
	MW_2Y	6,012,088	1	100	42.37	30	30.13
	GW	6,012,088	1	100	51.02	44	26.99

The empirical cumulative distribution function (CDF) plot in Figure 5.3 is for the growing window models; Panel A represents the kNN with arithmetic mean model, while Panel B represents the kNN with inverse distance weighting. The minor difference in central tendency is noticeable, e.g. in Panel A 65% of the values of *k* are less than or equal to 50, whereas the proportion in Panel B is 60%.

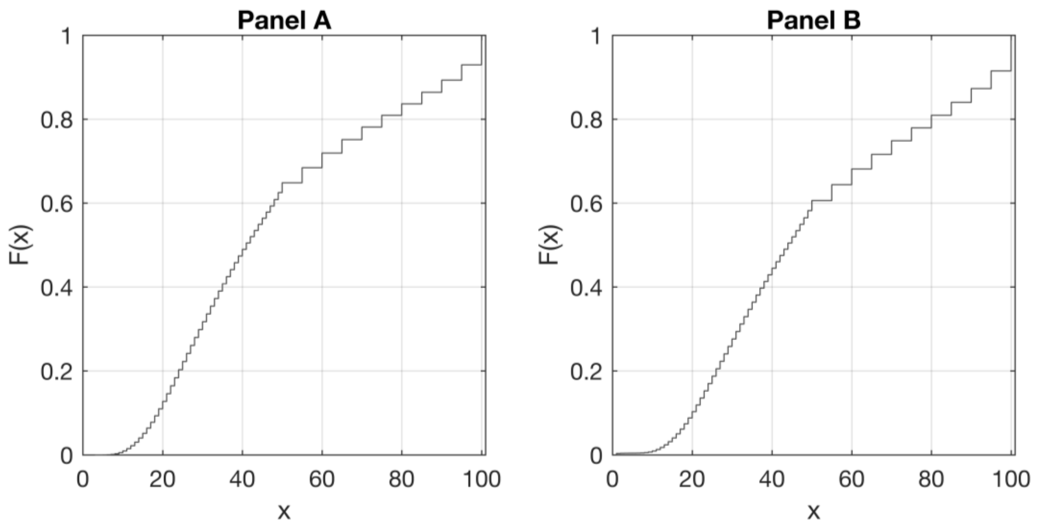

Figure 5.3: Empirical CDF of k for the growing window model for kNN with arithmetic mean in Panel A and kNN with inverse distance weighting in Panel B.

Regularisation Methods

The shrinkage methods in this analysis consist of an initial identification step for λ, employing a two-section search, where we first perform grid search to locate the optimal λ in the (base 10) common logarithm interval $[-10,10]$, using a unit-sized step in the $\log_{10}$ space, and then use the spotted value to perform bisection method for determining a more precise value for λ. The most extreme values that λ can take are -11 and 11; this happens when the optimal value for λ in the grid search section is either -10 or 10 and this value is then used as the initial midpoint of the bisection method, with potential extreme values lying one unit away from this midpoint, allowing for values between the interval $[-11,11]$, expressed in base 10 logarithm space. This leads to 24,596 unique values for λ across the six window types in the ridge regression model, and 18,862 unique values in the lasso regression model. Based on the descriptive statistics for λ in Table 5.3, which are reported for the entire stock universe in the base 10 logarithmic space, we observe significant differences in the distribution of λ between ridge regression and lasso regression. The values of λ are more dispersed throughout the interval $[-11,11]$ in the case of ridge regression, whereas lasso regression exhibits a positively skewed distribution, with mostly negative values, where the maximum is either 0 or 1. While the median for λ is around 2 for the ridge regression model, it is -2 for lasso regression.

Table 5.3
Descriptive statistics for the values of λ.

Shrinkage Model	Window Type	Observation	Min	Max	Mean	Median	Standard Deviation
Ridge Regression	MW_1M	7,111,213	-11	11	5.13	2.15	4.59
	MW_3M	7,012,429	-11	11	3.05	2.00	3.15
	MW_6M	6,864,419	-11	11	2.26	1.96	1.98
	MW_1Y	6,572,392	-11	11	1.94	2.00	1.17
	MW_2Y	6,012,088	-11	11	1.86	2.00	0.87
	GW	6,012,088	-11	11	2.04	2.00	0.80
Lasso Regression	MW_1M	7,111,213	-11	1	-0.36	0.00	0.87
	MW_3M	7,012,429	-11	1	-0.93	-1.00	1.08
	MW_6M	6,864,419	-11	1	-1.42	-1.25	1.22
	MW_1Y	6,572,392	-11	1	-1.82	-2.00	1.28
	MW_2Y	6,012,088	-11	0	-2.12	-2.00	1.27
	GW	6,012,088	-11	0	-2.43	-2.01	1.16

The difference in the distribution of λ stems from the different ways in which the two penalties work when dealing with two variables that are highly correlated: the L_1 regulariser (i.e. lasso regression) picks only one of the two correlated predictors, whereas the L_2 regulariser (i.e. ridge regression) keeps both of them in the model and jointly shrinks their coefficients. Therefore, L_2 penalties can minimise the prediction error better than L_1 penalties, although L_1 penalties can reduce overfitting and produce a more parsimonious model. In this section, we only examine the patterns observed in the distribution of these method-specific parameters. The predictive power of these models is compared in a subsequent section of this study.

5.3. Feature Selection

We analyse the results of feature selection, which was conducted by two methods: stepwise regression and lasso regression. The stepwise regression models were enforced to keep the intercept and the volume features (i.e. volume lags and volume windows), performing feature selection on the price features (i.e. intraday range, intraday return absPos/absNeg, and overnight return absPos/absNeg) and the five day-of-the-week features, whereas the lasso models could eliminate any feature from the full model. Because of this methodology difference, we start by investigating the selection of the volume features in the reduced model produced by lasso regression. Since every model starts by identifying the optimal order of the volume lags and volume windows, Table 5.4 outlines the proportion of each volume order (ranging from 1 to 15 for the volume lags and from 2 to 15 for the volume windows) in the full models throughout all of the window types of lasso regression. We observe that the volume lag and window orders below 7 are initially included in over 10% of the samples, out of a total of 39,584,629 model iterations. Once these full models are fit with the optimal orders, lasso regression performs variable selection.

Table 5.4
The proportion of volume lag and volume window orders in the full models of lasso regression, averaged over the six window types.

	Order														
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Volume lags	100	56.52	36.80	24.50	18.69	12.30	9.34	7.26	6.00	4.89	2.98	1.86	1.23	0.81	0.57
Volume windows	-	100.00	42.65	25.27	18.46	12.42	9.34	7.35	6.09	5.01	3.15	1.94	1.30	0.88	0.63

In order to compute the proportion of each volume feature in the reduced lasso model, we take into account that the volume lag and window orders are mostly less than the maximum value (i.e. 15) and, for each stock, we count the number of occurrences of each feature in the reduced model and also the number of models where a particular volume feature could not be possibly part of the reduced model, because the initially identified optimal order of the full model is lower than this particular volume order. Lasso regression selected the intercept in 100% of the model iterations. Figure 5.4 illustrates the selection proportions of the volume features (i.e. both volume lags and volume windows) in the reduced models across the six window types of lasso regression. The volume lags up to order 11 are selected in more than half of the model iterations, whereas the proportions for the volume window features are significantly lower, ranging from 10% to 35%.

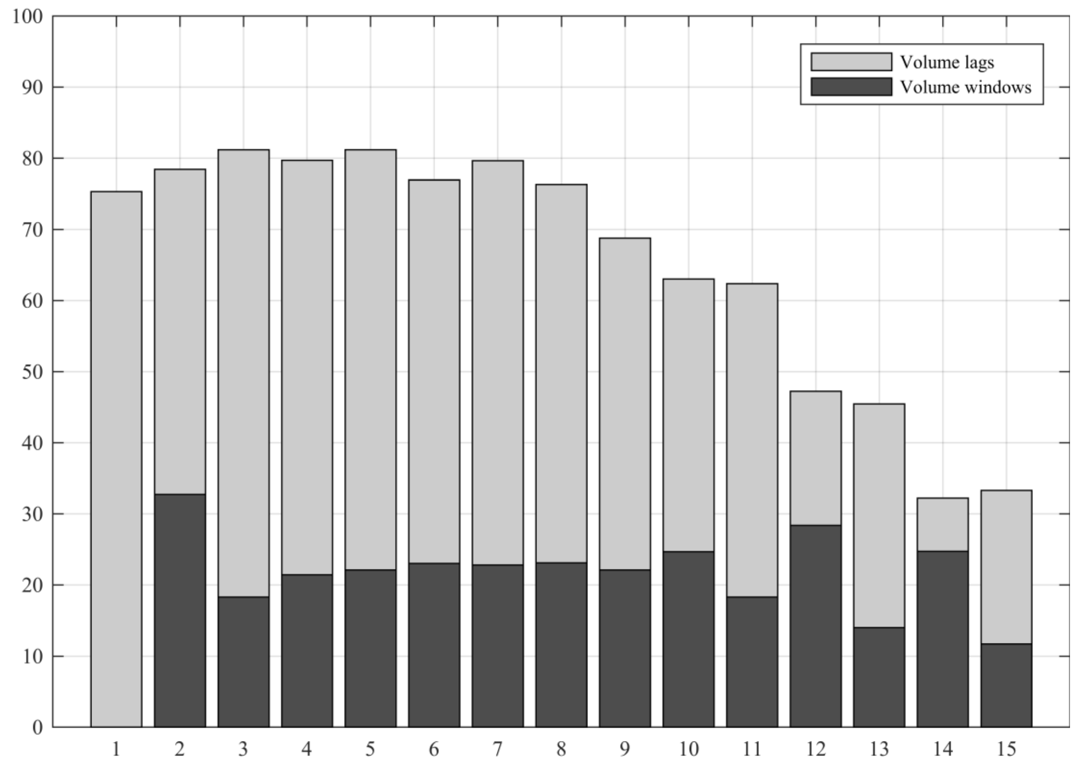

Figure 5.4: The proportion of volume lag and volume window orders in the reduced model produced by lasso regression.

The distribution of the selection proportions of the volume features for each window type is outlined in Table 5.5, for the purpose of spotting potential trends in the volume autoregressive nature. We observe a very high retention of the volume lag features for the models that are trained on at least 6 months of data (i.e. MW_6M, MW_1Y, MW_2Y, and GW). This trend is not followed by the selection of the volume windows, but we can conclude that past observations of the trading volume become more relevant when the learnt model is trained on more than 6 months of observations, and that volume becomes more autoregressive in this context.

Table 5.5
Selection proportion for the volume lag and volume window orders for each window type.

Order	Window type for volume lags						Window type for volume windows					
	MW_1M	MW_3M	MW_6M	MW_1Y	MW_2Y	GW	MW_1M	MW_3M	MW_6M	MW_1Y	MW_2Y	GW
1	17.70	56.56	82.94	95.47	99.28	99.94	0	0	0	0	0	0
2	23.68	65.29	88.02	95.89	98.12	99.52	5.74	18.10	32.64	43.14	48.83	47.80
3	29.75	71.37	91.92	97.07	97.83	99.24	7.60	20.23	23.94	16.51	14.25	27.20
4	30.43	67.71	90.10	96.30	96.47	97.27	11.94	27.23	31.64	19.56	12.71	25.57
5	36.85	71.30	90.46	96.15	96.27	96.05	15.30	28.71	33.85	20.43	12.09	22.30
6	40.33	60.84	79.05	91.04	95.13	95.23	19.04	26.46	33.56	22.16	14.42	22.60
7	41.57	64.21	84.99	94.46	97.57	95.03	19.62	23.95	31.74	24.32	15.86	21.45
8	34.09	60.38	79.70	92.01	97.63	94.07	20.73	24.49	29.78	25.01	17.96	20.65
9	5.56	45.45	77.14	92.23	97.66	94.61	20.45	23.91	26.90	23.27	18.61	19.56
10	0	0	90.32	95.59	97.02	95.08	0	33.33	28.57	39.39	26.99	19.87
11	0	0	90.91	92.65	93.68	96.84	0	0	41.67	21.57	26.95	19.59
12	0	0	0	91.30	95.28	96.77	0	0	100.00	27.27	25.22	17.76
13	0	0	0	75.00	99.29	98.41	0	0	0	33.33	32.28	18.31
14	0	0	0	0	95.45	97.72	0	0	0	100.00	28.95	19.41
15	0	0	0	0	100.00	99.80	0	0	0	0	35.71	34.42

Next, we discuss the feature selection of the price variables and the day-of-the-week indicator variables for both stepwise regression and lasso regression. Table 5.6 shows the selection percentage for each feature after averaging the results across the six window types. There is a notable difference in the selection proportion of the price features with significantly higher values for the stepwise regression implementation, whereas the price features are selected in approximately 3% of the lasso regression models. The day-of-the-week variables are similarly selected in both learning methods. Mondays have the highest percentage (approximately 42%), proving their great significance, either as a weekend effect or as an impact of the Monday bank holidays. Friday is the second most frequent day-of-the-week in the reduced model (being selected in approximately 25% of the models), possibly because of the weekend effect or due to the expiry day effect (e.g. stock index futures expiries or MSCI quarterly reviews).

Table 5.6
The features selected by stepwise regression and lasso regression, averaged across the six window types.

Variable	% in model	
	Stepwise regression	Lasso regression
Intraday range	27.62	3.71
Asymmetric intraday return (absPos)	26.44	3.34
Asymmetric intraday return (absNeg)	26.91	3.03
Asymmetric overnight return (absPos)	42.32	3.31
Asymmetric overnight return (absNeg)	38.60	3.19
Day-of-week: 1	41.81	42.84
Day-of-week: 2	20.60	26.18
Day-of-week: 3	18.88	25.36
Day-of-week: 4	17.63	21.43
Day-of-week: 5	23.52	26.99

5.4. Methodology Performance

For the comparison of the various methods, we need to bear in mind that different stocks have different error magnitudes. Employing the commonly used residual-based evaluation (i.e. including the cross-stock MSE of each method) would not be informative as we are looking to obtain stock-specific model stability. We need to look at some type of error normalisation and a simple way of doing this is to rank the different methods/models for each stock, and then look at the overall (average) ranks when comparing across stocks. Here, we ask the question "What proportion of the time was one method better than the other" and look at the relative performance for each stock. We perform the rankings for each stock and then answer how often each method was the best.

This error-based ranking approach is common in statistics (Rosset, et al., 2007), and can be used overall, as well as on the specific event days. Ranking-based evaluation measures for regression models are interpretable and they are robust against extreme outliers. We used the prediction data, containing the predicted and the observed trading volume for each trading day. Then, we computed the MSE for every stock and then ranked the methods based on the MSE. We used dense ranking (or "1223" ranking), where the models with equal predictions get the same ranking number and the next model receives the following ranking number.

The rank averages for each method and window type in Table 5.7 show that ridge regression trained on a 2-year moving window is the best method for all of the trading dates, including special events. The optimal length of the sliding window approach is 2 years, both for the cross-stock models (i.e. futures expiries, MSCI rebalances, and cross-market holidays) and the stock-specific models (i.e. OLS, stepwise regression, ridge regression, lasso regression, and kNN), with the exception of SVR, whose best error is achieved by the 6-month moving window. The 2-year moving window and the growing window tend to have the best performance across all methods. The 2-year moving window is better in 5 models (i.e. futures expiries, stepwise regression, ridge regression, lasso regression, and SVR), whereas the growing window is better in the other 5 models (i.e. MSCI rebalances, cross-market holidays, OLS, kNN with arithmetic average, and kNN with inverse distance weighting). The average rank for the stock-specific models are: 37.85 for MW_1M, 28.99 for MW_3M, 23.21 for MW_6M, 18.80 (18.61 including the special event models) for MW_1Y, 17.30 (16.55 including the special event models) for MW_2Y, and 17.52 (16.61 including the special event models) for GW. For the moving windows, the rank improves once the window length increases. However, the growing window has a slightly worse rank than MW_2Y, suggesting that recent data might be more relevant as there could be structural breaks across the years. This pattern is not applicable to the OLS method, where the lowest rank across all the models is achieved by OLS GW.

Table 5.7
The mean of the rank of each method and window type for all of the target dates.

Window size	Futures expiries	MSCI rebalances	Cross-market holidays	OLS	Stepwise regression	Ridge regression	Lasso regression	kNN (Arithmetic mean)	kNN (Inverse distance)	SVR
MW_1M	-	-	-	45.82	44.32	38.93	29.93	34.79	33.50	37.65
MW_3M	-	-	-	36.33	29.81	19.56	21.01	30.07	28.37	37.74
MW_6M	-	-	-	22.54	18.73	13.71	15.63	28.43	26.57	36.88
MW_1Y	13.93	16.85	23.74	13.01	11.12	8.78	11.16	26.10	24.52	36.92
MW_2Y	12.11	11.76	20.55	8.27	7.78	6.96	9.85	25.22	23.87	39.14
GW	13.93	10.57	19.03	7.94	8.20	8.16	10.09	23.62	22.27	42.33

The performance of the two kNN methods improves when the window size is larger, as more similar data points are found among the past observations. Throughout the stock-specific learning methods, ridge regression is the best one for 4 window types (MW_3M, MW_6M, MW_1Y, and MW_2Y). OLS has the best average rank for the growing window approach, although the rank of ridge regression growing window is the second best. When using fewer points to train the model, lasso regression achieves the best error. SVR with Gaussian kernel has the poorest performance; this could be further improved by implementing SVR with feature selection.

The standard deviations in Table 5.8 show the performance volatility of the three cross-stock models compared to the stock-specific models.

Table 5.8
The standard deviation of each method and window type for all of the target dates.

Window size	Futures expiries	MSCI rebalances	Cross-market holidays	OLS	Stepwise regression	Ridge regression	Lasso regression	kNN (Arithmetic mean)	kNN (Inverse distance)	SVR
MW_1M	-	-	-	2.06	2.55	5.37	8.03	5.68	5.48	5.51
MW_3M	-	-	-	6.93	7.12	5.38	6.58	6.43	6.33	3.79
MW_6M	-	-	-	7.32	5.88	4.84	4.97	6.61	6.59	3.95
MW_1Y	15.39	17.75	14.49	6.34	5.09	4.55	4.67	5.94	6.21	4.77
MW_2Y	14.99	16.13	13.88	5.72	4.66	4.92	4.88	5.62	6.05	4.74
GW	16.91	15.73	13.91	6.39	6.02	6.68	6.27	8.39	8.37	4.45

5.5. The Switching Model

From the methodology performance ranks, we infer an adaptive switching model. This is a cross-stock in-sample analysis that aims to better understand the performance of the various models on specific dates of interest. The 6,012,088 samples are drilled down to the lowest possible granularity by various temporal characteristics, such as: non-event dates (i.e. dates without any special event such as cross-market holidays, futures expiries, or MSCI rebalances), futures expiry index, MSCI rebalances, cross-market holidays, and day-of-the-week. This breakdown incorporates all combinations of these temporal aspects in order to find the best local model. Table 5.9 provides a dissection of the switching model for all the temporal combinations (i.e. every combination of event dates) and outlines each of the 42 sub-models, along with their best models, window sizes and average ranks. For a given trading day, the switching model chooses between these sub-models and picks the locally optimal model. Non-event dates and (special) event dates are mutually exclusive. Moreover, futures expiries and MSCI rebalances also have no overlapping days. The switching model is fit based on the 42 time intervals and their associated best models.

We make a specific comparison of errors on the various event days. The performance comparison between two methods for a given temporal circumstance is computed by getting the intersection of the trading dates that match the current temporal circumstance (e.g. non-event date, special event, certain day-of-the-week etc.) and comparing their ranks. In the situation of a clash between two special events, we choose the model preference by investigating the performance of these models using the intersection of the trading days for these special events, and then computing the MSE per stock and ranking each method for this reduced data set.

Table 5.9
Switching model drilldown based on granular temporal circumstances.

Event type	Samples	Method	Window	Rank Average	Rank Standard Deviation
Non-event Mondays	813,433	Ridge regression	MW_2Y	7.45	5.80
Non-event Tuesdays	1,063,098	Ridge regression	MW_2Y	7.45	5.63
Non-event Wednesdays	1,075,948	Ridge regression	MW_2Y	7.23	5.33
Non-event Thursdays	986,720	Ridge regression	MW_2Y	7.41	5.49
Non-event Fridays	930,276	Ridge regression	MW_2Y	7.25	5.22
FTSE MIB Futures expiry (Thursdays)	94	OLS	MW_2Y	16.73	10.92
IBEX 35 Futures expiry (Thursdays)	25	Lasso regression	MW_6M	18.32	12.41
OMX Stockholm 30 Futures expiry (Thursdays)	105	OLS	MW_2Y	13.30	9.90
Amsterdam Exchanges Futures expiry (Fridays)	2,410	Futures expiry	GW	3.06	3.90
CAC 40 Futures expiry (Fridays)	4,380	Futures expiry	GW	4.53	5.56
FTSE MIB Futures expiry (Fridays)	3,571	Futures expiry	GW	5.96	7.19
FTSE 100 Futures expiry (Fridays)	3,035	Futures expiry	GW	3.72	6.90
DAX Futures expiry (Fridays)	1,037	Futures expiry	MW_1Y	3.76	8.13
IBEX 35 Futures expiry (Fridays)	3,432	Futures expiry	GW	5.55	8.58
OMX Stockholm 30 Futures expiry (Fridays)	3,220	Ridge regression	MW_1Y	8.22	6.15
MSCI rebalance Mondays	417	Lasso regression	MW_3M	17.96	11.51
MSCI rebalance Tuesdays	2,091	MSCI rebalances	GW	11.95	12.90
MSCI rebalance Wednesdays	1,556	MSCI rebalances	MW_2Y	9.20	9.40
MSCI rebalance Thursdays	1,152	MSCI rebalances	GW	16.61	13.90
MSCI rebalance Fridays	3,268	MSCI rebalances	GW	15.04	14.57
Cross-market holiday Mondays	358,289	Cross-market holidays	GW	7.96	9.37
Cross-market holiday Tuesdays	153,291	Cross-market holidays	GW	10.54	9.82
Cross-market holiday Wednesdays	141,087	Cross-market holidays	GW	8.91	10.15
Cross-market holiday Thursdays	221,034	Cross-market holidays	GW	8.57	9.58
Cross-market holiday Fridays	230,755	Cross-market holidays	GW	8.31	8.63
Cross-market holiday and Amsterdam Exchanges Futures expiry (Thursdays)	59	OLS	GW	17.31	12.42
Cross-market holiday and Amsterdam Exchanges Futures expiry (Fridays)	733	Futures expiry	GW	8.94	11.58
Cross-market holiday and CAC 40 Futures expiry (Thursdays)	108	OLS	MW_3M	20.23	16.55
Cross-market holiday and CAC 40 Futures expiry (Fridays)	1,342	Futures expiry	MW_2Y	6.47	9.86
Cross-market holiday and FTSE MIB Futures expiry (Thursdays)	70	Stepwise regression	MW_1Y	12.14	9.63
Cross-market holiday and FTSE MIB Futures expiry (Fridays)	955	Futures expiry	MW_2Y	9.69	11.07
Cross-market holiday and FTSE 100 Futures expiry (Thursdays)	86	Futures expiry	GW	13.67	12.41
Cross-market holiday and FTSE 100 Futures expiry (Fridays)	1,326	Futures expiry	GW	7.19	10.91
Cross-market holiday and DAX Futures expiry (Thursdays)	29	Futures expiry	MW_2Y	14.03	17.95
Cross-market holiday and DAX Futures expiry (Fridays)	457	Futures expiry	MW_1Y	2.62	3.18
Cross-market holiday and IBEX 35 Futures expiry (Thursdays)	84	Stepwise regression	MW_2Y	15.12	10.86
Cross-market holiday and IBEX 35 Futures expiry (Fridays)	1,026	Futures expiry	MW_2Y	7.35	10.78
Cross-market holiday and OMX Stockholm 30 Futures expiry (Thursdays)	133	Cross-market holidays	GW	6.80	7.79
Cross-market holiday and OMX Stockholm 30 Futures expiry (Fridays)	831	Futures expiry	GW	11.91	7.10
Cross-market holiday and MSCI rebalance Mondays	342	Cross-market holidays	GW	15.51	12.88
Cross-market holiday and MSCI rebalance Wednesdays	199	Cross-market holidays	GW	14.53	11.80
Cross-market holiday and MSCI rebalance Fridays	584	MSCI rebalances	GW	14.67	14.42

Next, we compare the switching and the non-switching models, using the average ranks of these methods. The switching model does not have a certain window size enforced as it adapts to the right window size depending on the temporal circumstance. The average ranks in Table 5.10, along with the standard deviations in Table 5.11, show the impressive performance of the switching model, which strongly suggests that markets switch to different states on special events. The switching model has the lowest average rank (5.64); the next best rank is achieved by ridge regression MW_2Y (7.73) and the worst by OLS MW_1M (46.82). The switching model was ranked first in 26.32% of the 2,181 stocks, whereas ridge regression MW_2Y is the best in only 1.65% of the cases. Throughout 76.98% of the stocks, the switching model outperforms the second best model, i.e. ridge regression MW_2Y. Moreover, the switching model is better than the least performing model for every stock in our universe.

Table 5.10
The average rank for every method and window type, along with the switching model, for all of the target dates.

Window size	Futures expiries	MSCI rebalances	Cross-market holidays	OLS	Stepwise regression	Ridge regression	Lasso regression	kNN (Arithmetic mean)	kNN (Inverse distance)	SVR	Switching model
MW_1M	-	-	-	46.82	45.32	39.92	30.91	35.78	34.49	38.64	-
MW_3M	-	-	-	37.33	30.81	20.52	21.96	31.05	29.34	38.74	-
MW_6M	-	-	-	23.52	19.70	14.62	16.55	29.40	27.54	37.87	-
MW_1Y	14.41	17.36	24.53	13.90	11.98	9.59	12.01	27.07	25.50	37.92	-
MW_2Y	12.56	12.12	21.31	9.07	8.58	7.73	10.71	26.21	24.86	40.14	-
GW	14.39	10.90	19.77	8.57	8.84	8.79	10.80	24.58	23.23	43.33	-
-	-	-	-	-	-	-	-	-	-	-	5.64

Table 5.11
The standard deviation of the rank of each method and window type, along with the switching model, for all of the target dates.

Window size	Futures expiries	MSCI rebalances	Cross-market holidays	OLS	Stepwise regression	Ridge regression	Lasso regression	kNN (Arithmetic mean)	kNN (Inverse distance)	SVR	Switching model
MW_1M	-	-	-	2.06	2.56	5.40	8.08	5.71	5.50	5.54	-
MW_3M	-	-	-	6.95	7.14	5.48	6.69	6.48	6.38	3.82	-
MW_6M	-	-	-	7.35	5.93	4.97	5.11	6.68	6.66	3.97	-
MW_1Y	15.77	18.14	14.79	6.44	5.21	4.65	4.82	5.99	6.26	4.79	-
MW_2Y	15.34	16.52	14.20	5.81	4.73	4.98	4.96	5.65	6.07	4.75	-
GW	17.29	16.11	14.22	6.62	6.27	6.93	6.50	8.45	8.43	4.46	-
-	-	-	-	-	-	-	-	-	-	-	4.82

Figure 5.5 illustrates one of the best switching models (i.e. the lowest MSE for a particular stock) for Telefonica SA (TEF.MC), whose MSE for the entire period is 0.078. The plot shows the observed volume and the predicted volume of the switching model for a cropped period of 1 year, due to clarity considerations, between 02/01/2009 and 30/12/2009, where the 1-year MSE is 0.063 for 254 observations.

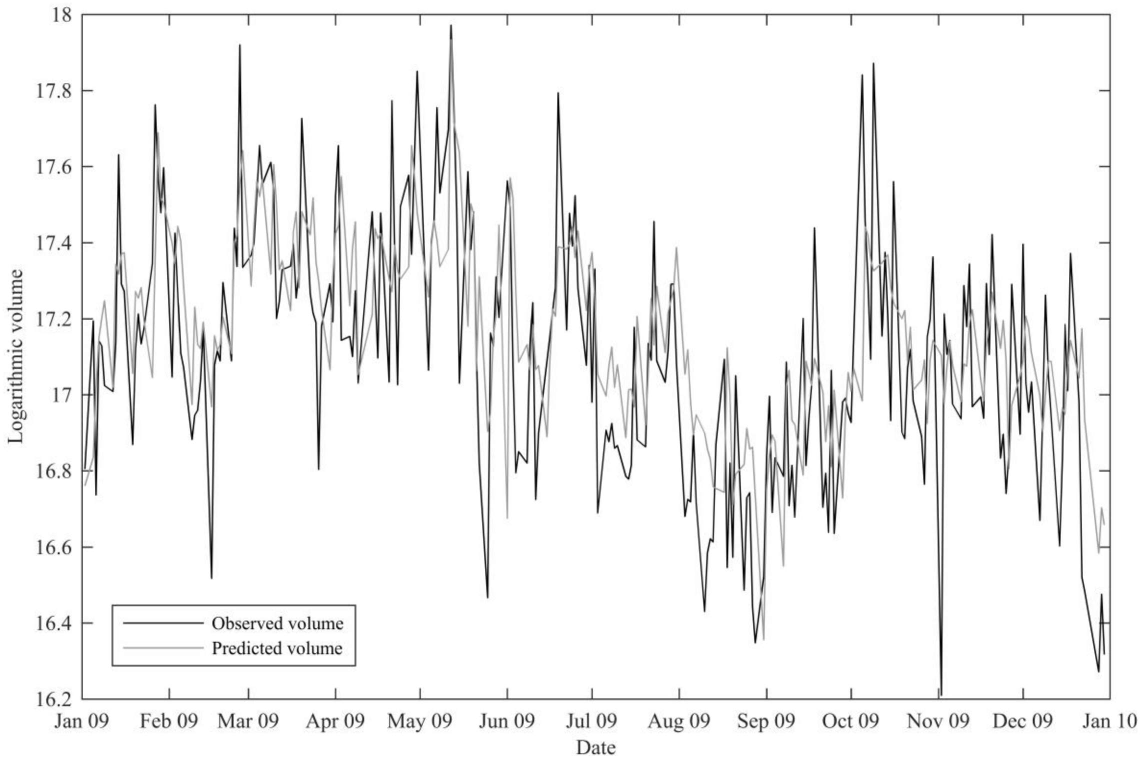

Figure 5.5: Volume prediction using the switching model over one year for Telefonica SA.

The best performance improvement achieved by the switching model, compared to the best initial stock-specific models, is 17.41% for Total SA (TOTF.PA). This is computed using the relative change in MSE from the best initial stock-specific model to the switching model. For clarity purposes, we cropped its timeline in Figure 5.6 to 1 year, between 02/01/2013 and 31/12/2013. For these 255 observations, the improvement percentage is 33.40%, the MSE of the best initial model (i.e. ridge regression MW_2Y) is 0.095884, while the MSE of the switching model is 0.063856.

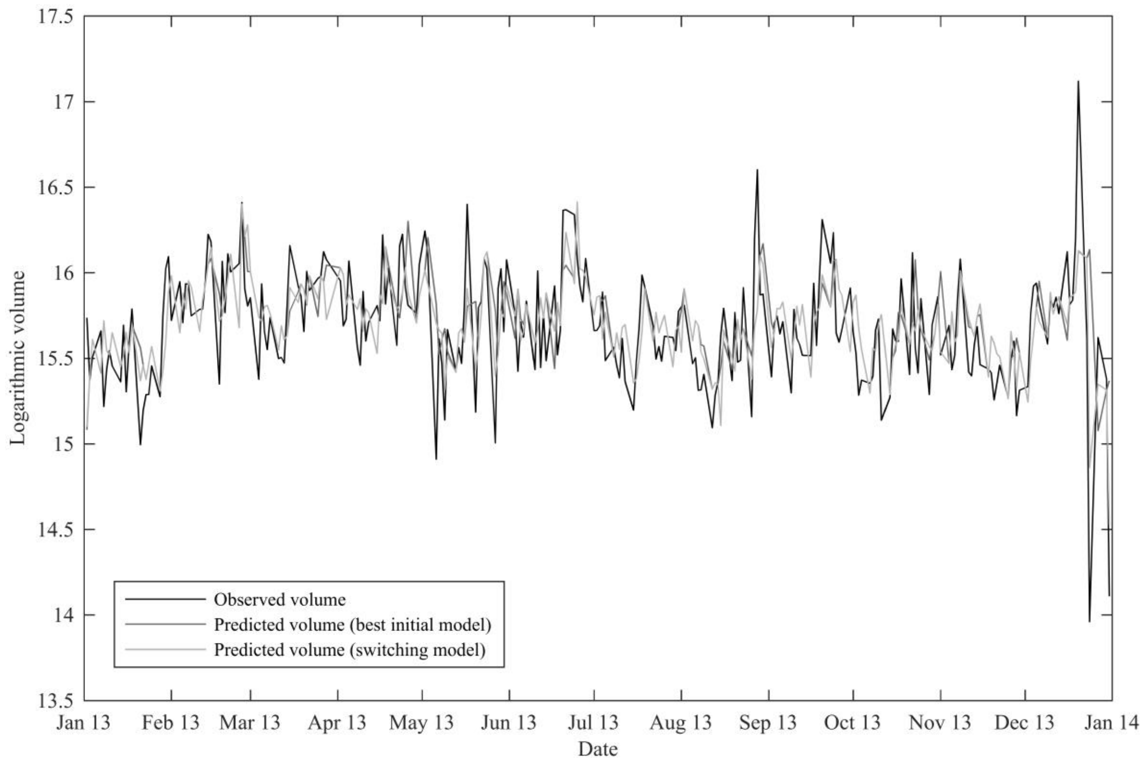

Figure 5.6: The performance improvement of Total SA from the best initial stock-specific model to the switching model.

The largest performance improvement of the switching model when compared to the worst performing initial model is 99.99% improvement and is achieved for 5 stocks. As an illustrative example, Avenir Finance SA (AVEF.PA) has 3,220 observations and the MSE of the worst initial model (i.e. OLS MW_1M) is 4640865.802, whereas the switching model MSE is 4.448. Across all stocks, the performance of the worst initial models is improved by the switching model by 74.595% on average.

5.6. Stock-Specific Metamodel

Since the switching model provides an in-sample analysis suggesting the various states markets shift between, we further pose the question whether we can improve the switching model better and provide an out-of-sample model for a given stock. Therefore, we use the ranking-based evaluation measures in order to build an out-of-sample stock-specific metamodel (or surrogate model). For a given stock, we employ a fixed size window of past observations, where the various methods are ranked and the best method is picked to make the next one-step ahead volume forecast. We train two metamodels, using a 1-month and a 3-month moving window. At each step, we evaluate the previous month (corresponding to 21 trading days) or 3 months (corresponding to 63 trading days) and we pick the current best performing method at a given time to make the next day volume prediction. We must note that these metamodel moving windows are different from the concept of moving windows applied to the stock-specific initial models. Here, we still train the initial models using the various training windows (ranging from the one-month moving window to the growing window), and then we investigate the prediction error over the past month or 3 months in order to select the best model throughout the recent time series.

We compute the squared errors for all of the stocks. Then, for each stock, we perform a moving average over one month (21 days) and three months (63 days). We discard 20 stocks having less than 100 test dates as these would not provide enough data for this out-of-sample analysis. This results in 6,011,125 samples, which are further processed by discarding the initial *n* days for each stock, where *n* is the lag number (i.e. 21 trading days or 63 trading days), yielding 5,965,744 samples for the 1-month metamodel and 5,874,982 samples for the 3-month metamodel.

One-Month Metamodel

The one-month metamodel is the 27th best model based on the average rank (23.42) in Table 5.12, having a standard deviation of 6.40. Throughout the initial models, there are 8 cross-stock models and 19 stock-specific models that outperform the one-month metamodel. The best rank is achieved by ridge regression MW_2Y and the worst one by OLS MW_1M. The one-month metamodel was the best model for 2 stocks (0.09%), whereas ridge regression MW_2Y was the best in 8.33% of the stocks. The metamodel is better than ridge regression MW_2Y for 43 stocks (1.99%) and it is better than the least performing model, i.e. OLS MW_1M, for all of the 2,161 analysed stocks.

Table 5.12
The mean of each method and window type, along with the one-month metamodel, for all of the target dates.

Window size	Futures expiries	MSCI rebalances	Cross-market holidays	OLS	Stepwise regression	Ridge regression	Lasso regression	kNN (Arithmetic mean)	kNN (Inverse distance)	SVR	1-month meta model
MW_1M	-	-	-	46.84	45.37	40.06	30.79	35.80	34.48	38.68	-
MW_3M	-	-	-	37.34	30.77	19.93	21.55	30.83	29.06	38.78	-
MW_6M	-	-	-	22.94	18.92	13.76	15.79	29.23	27.32	37.85	-
MW_1Y	14.28	17.31	24.40	12.98	11.10	8.73	11.15	26.82	25.19	37.90	-
MW_2Y	12.36	12.05	21.09	8.22	7.71	6.91	9.82	25.85	24.47	40.13	-
GW	14.27	10.86	19.58	7.87	8.18	8.13	10.05	24.11	22.72	43.37	-
-	-	-	-	-	-	-	-	-	-	-	23.42

The largest improvement from the best initial stock-specific model to the one-month metamodel is 3.88% and it is achieved for DBV Technologies SA (DBV.PA). Figure 5.7 illustrates the predictions of the one-month metamodel compared to the best initial model (i.e. stepwise regression) for 247 observations of DBV Technologies SA, between 14/05/2014 and 30/04/2015, where the metamodel performance improvement is 4.5458%. The best initial model MSE is 0.51099, whereas the metamodel MSE is 0.48776.

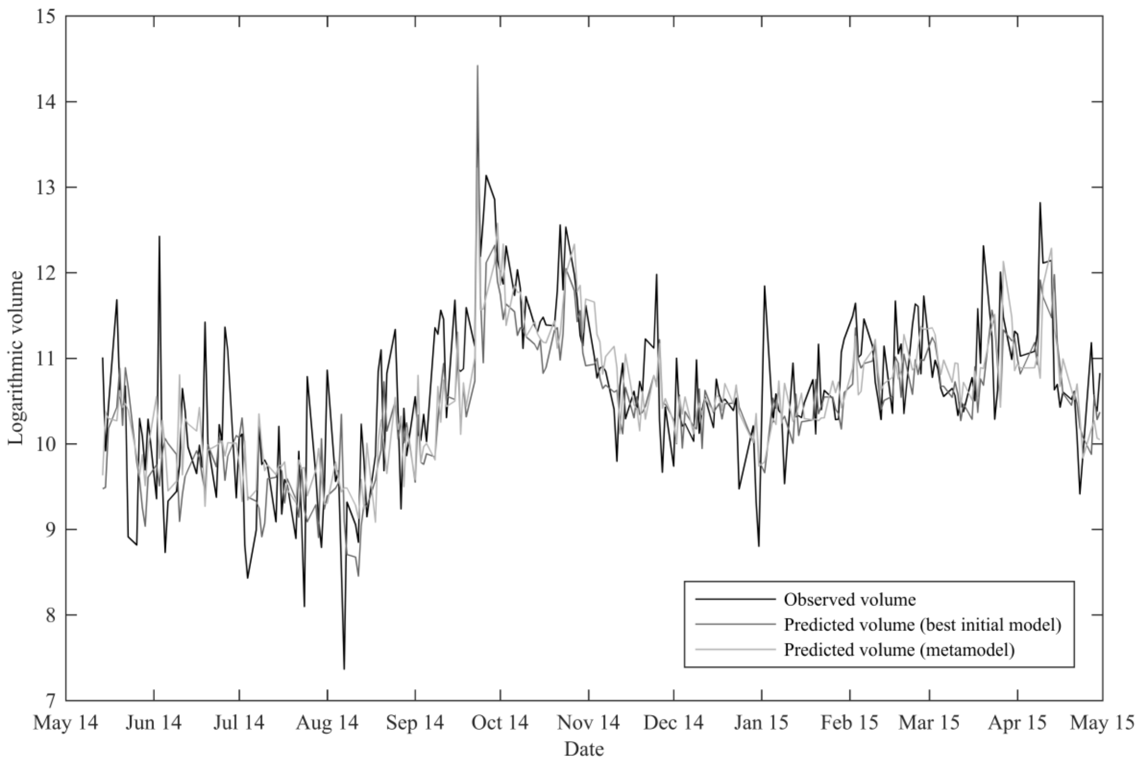

Figure 5.7: The volume prediction of the best initial model and the one-month metamodel for DBV Technologies SA.

Three-Month Metamodel

The three-month metamodel model is the 19th best model based on the average rank (14.93) outlined in Table 5.13, having a standard deviation of 5.31. There are 13 cross-stock models and 5 initial stock-specific models that are better than the three-month metamodel. Again, the best rank is achieved by ridge regression MW_2Y and the worst by OLS MW_1M. The three-month metamodel was the best model in only 0.42% of the stocks, i.e. 9 out of 2,161 stocks, whereas ridge regression MW_2Y is the top ranked model in 8.28% of the stocks. The metamodel is better than the ridge regression MW 2Y model in 9.58% of the stock universe (i.e. 207 stocks) and it outperforms the least performing initial model across all of the stocks.

Table 5.13
The mean of each method and window type, including the three-month metamodel, for all of the target dates.

Window size	Futures expiries	MSCI rebalances	Cross-market holidays	OLS	Stepwise regression	Ridge regression	Lasso regression	kNN (Arithmetic mean)	kNN (Inverse distance)	SVR	3-month meta model
MW_1M	-	-	-	46.83	45.35	40.06	30.97	35.83	34.52	38.75	-
MW_3M	-	-	-	37.25	30.77	20.39	21.89	31.07	29.27	38.79	-
MW_6M	-	-	-	23.41	19.55	14.18	16.39	29.42	27.53	37.83	-
MW_1Y	14.44	17.57	24.61	13.28	11.31	8.87	11.44	27.09	25.40	37.82	-
MW_2Y	12.53	12.18	21.42	8.32	7.78	6.96	10.01	26.11	24.75	40.03	-
GW	14.36	11.11	19.87	7.99	8.34	8.28	10.31	24.43	23.04	43.36	-
-	-	-	-	-	-	-	-	-	-	-	14.93

The largest improvement from the best initial model is achieved by the three-month metamodel in the case of Sponda Oyj (SDA1V.HE), with a performance improvement of 3.24%. Figure 5.8 illustrates the volume predictions made by the best initial stock-specific model and the three-month metamodel for Sponda Oyj. There are 253 observations in the one-year cropped timeline, between 03/01/2005 and 30/12/2005. The metamodel, whose MSE is 0.8728, improves the performance of the best initial model (i.e. lasso MW_1Y), whose MSE is 1.0219, by 14.5936%.

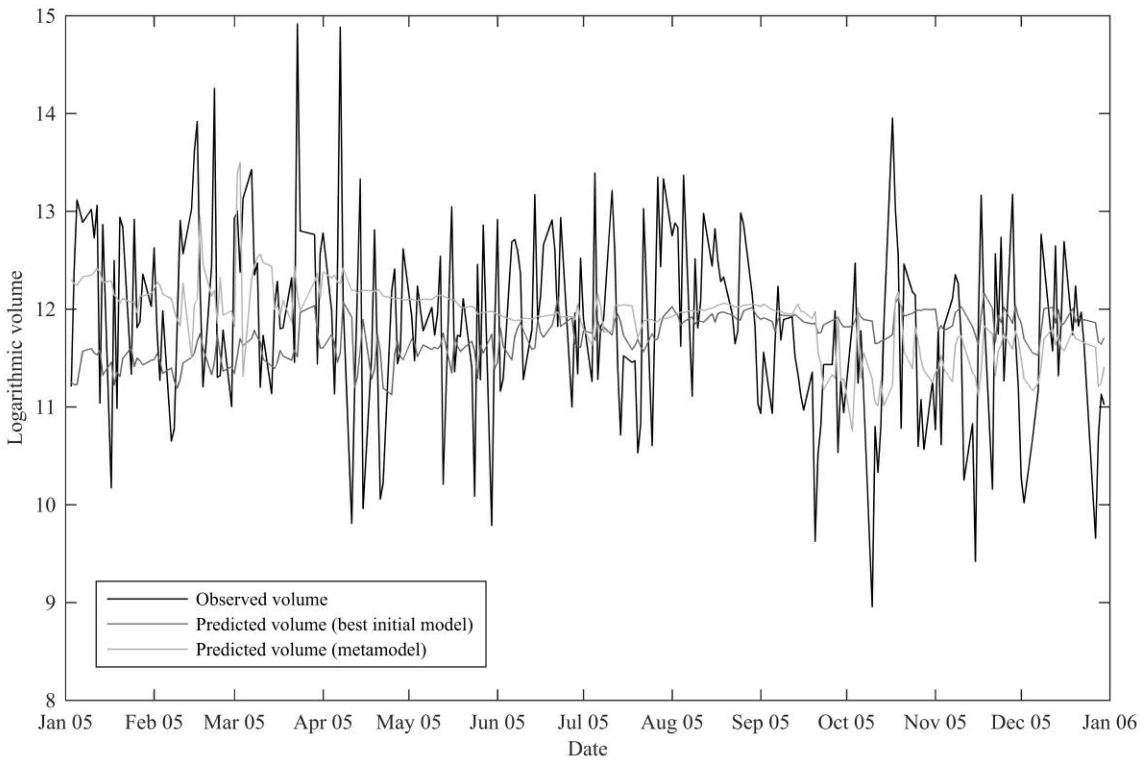

Figure 5.8: The best initial model vs. the 3-month metamodel volume prediction for Sponda Oyj.

The three-month metamodel has a significantly better performance than the one-month metamodel and provides improved model stability, by exhibiting a lower standard deviation.

6. Discussion

Volume prediction is critically important for optimal order allocation in order to minimise the market impact. Traders and portfolio managers aim to model the market liquidity by predicting the trading volume such that they do not over-participate, by incurring excessive market impact, or under-participate, by incurring opportunity cost. The study employs an enormous data set, comprising the daily market data for 2,353 European stocks from 21 countries, along with a precisely constructed trading calendar covering more than 15 years for these 21 European countries and the United States.

The aim of this study is to train a variety of learning methods and window types in order to better understand how they perform in certain circumstances, by specifically investigating event dates, such as cross-market holidays, futures expiries, or MSCI quarterly reviews, along with other aspects, e.g. day-of-the-week effect, price-volume relation asymmetry etc. Considering the difference in the volume and price magnitudes among our European stock universe, we independently train 42 stock-specific models, by fitting seven learning methods (i.e. OLS, stepwise regression, ridge regression, lasso regression, kNN with arithmetic mean, kNN with inverse distance weighting, and SVR) for each window type (i.e. the 1-month, 3-month, 6-month, 1-year, and 2-year moving windows, and the growing window). These independently fit stock-specific models had a remarkable runtime of 33 years on the high performance computing clusters. Three additional models are trained using cross-stock normalised observations for the special events (i.e. cross-market holidays, futures expiries, and MSCI rebalances), which are eventually used to make stock-specific predictions. These cross-stock models are learnt using the 1-year and 2-year moving windows and the growing window, producing 9 cross-stock models in total.

Our results corroborate previous findings and provide empirical evidence that the trading volume is autoregressive and this property becomes stronger (i.e. the autoregressive order increases) once the size of the training set is large enough (i.e. in excess of 6 months of training data). For example, the median order of the volume lags and volume windows for the growing window approach is 7. The volume observations from the previous one and a half weeks provide relevant trends, given that the model is trained on a substantial number of data points. The number of neighbours selected by kNN increases gradually once the window length becomes larger. Both kNN with arithmetic mean and kNN with inverse distance weighting reach the maximum number of 100 neighbours that we imposed in our analysis only when the size of the training window is at least 6 months long.

While investigating the effects on volume of the days of the week, we provide consistent results with previous findings (Batrinca, et al., 2016). Mondays are retained by the feature selection methods in 42% of the models, followed by Fridays, whose indicator variable is kept in almost 25% of the models.

Using a ranking-based evaluation, we report that the best model is trained using ridge regression on a two-year moving window. The results indicate that OLS, i.e. the study's most rudimentary method, trained on a growing window has a marginally worse performance than ridge regression, which deals with the multicollinearity problems. The rank of the moving windows improves once the window length increases and the optimal size of the moving window approach is 2 years, whose performance is similar to that of the growing window, although the 2-year moving window has a better rank average across all of the seven learning methods. This could be explained by possible structural breaks across the 15 years analysed by this study, potentially worsening the performance of the growing window when the window reaches a very large size.

Based on a thorough dissection of the temporal circumstances for all of the stocks, we infer a cross-stock switching model that employs the best initial stock-specific model for a given date characteristic. There are 42 disjoint temporal circumstances that are described by different models, which best apply to a particular state of the financial markets. This cross-stock in-sample analysis drills down the 6 million samples into high granularity circumstances identified based on a variety of temporal factors, such as non-event dates, futures expiries, MSCI rebalances, cross-market holidays, day-of-the-week etc. The excellent performance achieved by the switching model confirms our hypothesis that markets are event-driven and shift to different states based on special events.

Ultimately, the goal of this research is to improve model stability and we propose an out-of-sample stock-specific metamodel that evaluates the initial independent stock-specific models on a time window of one month or three months, and picks the model whose performance rank is the best throughout the chosen time window, in order to predict the following day's trading volume. The average performance rank of the one-month metamodel is 23rd, whereas the three-month metamodel performs significantly better and its rank decreases to the 15th position. These metamodels provide an out-of-sample dynamic framework, which aims to improve error stability and forecasts the expected volume to mitigate market impact.

Funding

This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

Bibliography

Abraham, A. & Ikenberry, D. L., 1994. The Individual Investor and the Weekend Effect. *The Journal of Financial and Quantitative Analysis,* 29(2), pp. 263-277.
Assogbavi, T. & Osagie, J. E., 2006. Equity valuation process and price-volume relationship on emerging markets. *International Business & Economics Research Journal,* 5(9), pp. 7-18.
Batrinca, B., Hesse, C. & Treleaven, P., 2016. *European Trading Volumes on Cross-Market Holidays,* London: Manuscript submitted for publication.
Batrinca, B., Hesse, C. & Treleaven, P., 2016. *Examining Drivers of Trading Volume in European Markets,* London: Manuscript submitted for publication.
Batrinca, B., Hesse, C. & Treleaven, P., 2016. *Expiry Day Effects on European Trading Volumes,* London: Manuscript submitted for publication.
Beaver, W. H., 1968. The Information Content of Annual Earnings Announcements. *Journal of Accounting Research,* Volume 6, pp. 67-92.
Berument, H. & Kiymaz, H., 2001. The day of the week effect on stock market volatility. *Journal of Economics and Finance,* 25(2), pp. 181-193.
Bishop, C. M., 2007. *Pattern Recognition and Machine Learning.* 1st ed. New York: Springer-Verlag New York.
Casado, J., Muga, L. & Santamaria, R., 2013. The effect of US holidays on the European markets: when the cat's away.... *Accounting and Finance,* Volume 53, pp. 111-136.
Chakrabarti, R., Huang, W., Jayaraman, N. & Lee, J., 2005. Price and volume effects of changes in MSCI indices–nature and causes. *Journal of Banking & Finance,* 29(5), pp. 1237-1264.
Cheung, C. S. & Kwan, C. C. Y., 1992. A note on the transmission of public information across international stock markets. *Journal of Banking & Finance,* 16(4), pp. 831-837.
Chiang, C.-H., 2009. *Trading Volume, Returns and Option Expiration Date,* New York: Columbia University.
Chow, Y., Yung, H. H. & Zhang, H., 2003. Expiration day effects: The case of Hong Kong. *Journal of Futures Markets,* 23(1), pp. 67-86.
Cortes, C. & Vapnik, V., 1995. Support-Vector Networks. *Machine Learning,* 20(3), pp. 273-297.
Cross, F., 1973. The Behavior of Stock Prices on Fridays and Mondays. *Financial Analysts Journal,* 29(6), pp. 67-69.
Drucker, H. et al., 1997. Support Vector Regression Machines. *Advances in neural information processing systems,* Volume 9, pp. 155-161.
Dubois, M. & Louvet, P., 1996. The day-of-the-week effect: The international evidence. *Journal of Banking & Finance,* 20(9), pp. 1463-1484.
Duda, R. O., Hart, P. E. & Stork, D. G., 2000. *Pattern Classification.* 2nd Edition ed. New York: John Wiley & Sons, Inc..
French, K. R., 1980. Stock returns and the weekend effect. *Journal of Financial Economics,* 8(1), pp. 55-69.
Gibbons, M. R. & Hess, P., 1981. Day of the Week Effects and Asset Returns. *The Journal of Business,* 54(4), pp. 579-596.
Harris, L., 1986. A transaction data study of weekly and intradaily patterns in stock returns. *Journal of Financial Economics,* 16(1), pp. 99-117.
Harris, M. & Raviv, A., 1993. Differences of Opinion Make a Horse Race. *The Review of Financial Studies,* 6(3), pp. 473-506.
Hassanat, A. B., Abbadi, M. A. & Altarawneh, G. A., 2014. Solving the Problem of the K Parameter in the KNN Classifier Using an Ensemble Learning Approach. *(IJCSIS) International Journal of Computer Science and Information Security,* 12(8), pp. 33-39.

Hastie, T., Tibshirani, R. & Friedman, J., 2011. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction.* 2nd ed. New York: Springer.
Hong, H. & Stein, J. C., 2007. Disagreement and the Stock Market. *Journal of Economic Perspectives,* 21(2), pp. 109-128.
Jaffe, J. & Westerfield, R., 1985. The Week-End Effect in Common Stock Returns: The International Evidence. *The Journal of Finance,* 40(2), pp. 433-454.
Karpoff, J. M., 1987. The Relation Between Price Changes and Trading Volume: A Survey. *The Journal of Financial and Quantitative Analysis,* 22(1), pp. 109-126.
Keerthi, S. S. & Lin, C.-J., 2003. Asymptotic behaviors of support vector machines with Gaussian kernel. *Neural Computation,* 15(7), pp. 1667-1689.
Lakonishok, J. & Maberly, E., 1990. The Weekend Effect: Trading Patterns of Individual and Institutional Investors. *The Journal of Finance,* 45(1), pp. 231-243.
MathWorks, 2016. *Understanding Support Vector Machine Regression.* [Online]
Available at: http://uk.mathworks.com/help/stats/understanding-support-vector-machine-regression.html?refresh=true
[Accessed 10 03 2016].
Pettengill, G. N., 2003. A Survey of the Monday Effect Literature. *Quarterly Journal of Business and Economics,* 42(3), pp. 3-27.
Platt, J. C., 1998. *Sequential Minimal Optimization: A Fast Algorithm for Training Support Vector Machines,* Redmond: Microsoft Research.
Pope, P. F. & Yadav, P. K., 1992. The Impact of Option Expiration on Underlying Stocks: The UK Evidence. *Journal of Business Finance & Accounting,* 19(3), pp. 329-344.
Rosset, S., Perlich, C. & Zadrozny, B., 2007. Ranking-based evaluation of regression models. *Knowledge and Information Systems,* 12(3), pp. 331-353.
Sadath, A. & Kamaiah, B., 2011. Expiration Effects of Stock Futures on the Price and Volume of Underlying Stocks: Evidence from India. *The IUP Journal of Applied Economics,* 10(3), pp. 25-38.
Stoll, H. R. & Whaley, R. E., 1997. Expiration-Day Effects of the All Ordinaries Share Price Index Futures: Empirical Evidence and Alternative Settlement Procedures. *Australian Journal of Management,* 22(2), pp. 139-174.
Sukumar, N. & Cimino, A., 2012. *European Trading Volumes Rise Before Last Options Expiry.* [Online]
Available at: http://www.bloomberg.com/news/2012-12-18/european-trading-volumes-rise-before-last-options-expiry.html
[Accessed 19 05 2014].
Tibshirani, R., 1996. Regression Shrinkage and Selection via the Lasso. *Journal of the Royal Statistical Society. Series B (Methodological)* , 58(1), pp. 267-288.
Vapnik, V. N., 1999. An overview of statistical learning theory. *IEEE Transactions on Neural Networks,* 10(5), pp. 988-999.
Vipul, 2005. Futures and options expiration-day effects: The Indian evidence. *Journal of Futures Markets,* 25(11), pp. 1045-1065.
Ying, C. C., 1966. Stock Market Prices and Volumes of Sales. *Econometrica,* 34(3), pp. 676-685.