European Exchange Trading Funds Trading with Locally Weighted Support Vector Regression

June 2016

Georgios Sermpinis*, Charalampos Stasinakis, Rafael Rosillo***, David de la Fuente******

In this paper, two different Locally Weighted Support Vector Regression (wSVR) algorithms are generated and applied to the task of forecasting and trading five European Exchange Traded Funds. The trading application covers the recent European Monetary Union debt crisis. The performance of the proposed models is benchmarked against traditional Support Vector Regression (SVR) models. The Radial Basis Function, the Wavelet and the Mahalanobis kernel are explored and tested as SVR kernels. Finally, a novel statistical SVR input selection procedure is introduced based on a principal component analysis and the Hansen *et al.* (2011) model confidence test. The results demonstrate the superiority of the wSVR models over the traditional SVRs and of the v-SVR over the ε-SVR algorithms. We note that the performance of all models varies and considerably deteriorates in the peak of the debt crisis. In terms of the kernels, our results do not confirm the belief that the Radial Basis Function is the optimum choice for financial series.

Keywords: Locally Weighted Support Vector Regression; Support Vector Regression; Kernels; Trading; Exchange Traded Funds;

*University of Glasgow Business School, University of Glasgow, Adam Smith Building, Glasgow, G12 8QQ, UK (georgios.sermpinis@glasgow.ac.uk)
**University of Glasgow Business School, University of Glasgow, Adam Smith Building, Glasgow, G12 8QQ, UK (charalampos.stasinakis@glasgow.ac.uk)
*** Business and Management Department, University of Leon, Campus de Vegazana s/n 24071, Spain (rrosc@unileaon.es)
**** Business and Management Department, University of Oviedo, Campus de Viesques, s/n. 33204, Spain (david@uniovi.es)

1. Introduction

Support Vector Machines (SVMs) is a class of powerful nonlinear algorithms with a variety of applications in almost every aspect of Science. They can generate nonlinear decision boundaries through linear classifiers and still have a simple geometric interpretation. Initially their applicability was limited to classification problems but this changed with the novel work of Vapnik (1995) and the introduction of Support Vector Regression (SVRs). SVRs are capable of modelling high-dimensional, noisy and complex feature problems (Suykens *et al.*, 2002). These features made them perfect tools for Finance practitioners challenged with the task of producing profitable trading algorithms.

Financial trading series are dominated by non-linear, noise and non-stationary nature. They are affected not only by a large universe of relevant time series but also by behavioural elements (Froot *et al.*, 1992) and exogenous factors such as political decisions (Frisman, 2001). It is not possible to model financial trading series completely. Practitioners seek for algorithms capable of generating enough profitable trading signals to beat the market. Contrary to the popular belief, the aim of traders is not to maximize their wealth but to survive in the market (Lo, 2004). Linear trading rules seem unable to help traders on this task. They seem to have low power and volatile behaviour through time (LeBaron (2000) and Qi and Wu (2006)). On the other hand, artificial intelligence models and heuristics have provided promising empirical evidence in trading applications (see amongst others Allen and Karjalainen (1999), Jasic and Wood (2004) and Sermpinis *et al.* (2016)). However their atheoretic nature and the lack of sufficient robustness checks on their performance is generating scepticism among finance professionals and researchers.

In this study, twelve SVRs models are applied to the task of forecasting and trading five equity markets Exchange Traded Funds (ETFs). Compared with other non-linear techniques (such as Neural Networks), SVRs are based on sound theory and are capable of providing global and unique solutions. In SVRs the practitioner can apply kernel functions to data so the vector space is not fixed in terms of dimensions. Although the use of SVRs in financial forecasting is not new, this study aims to provide an insight to a class of promising SVR algorithms, the locally weighted SVR (wSVR). wSVRs are based on the notion that the nearest to the predictor values are its best indicators. They can capture the local trend of the data and handle noise adaptively. This feature seems advantageous in financial series which exhibit structural breaks, where recent observations have more influence than past.

There is a wealth of SVR applications in Finance. Pai *et al.* (2006) combines SVR with genetic algorithms and forecasts accurately several exchange rates. Ince and Trafalis (2006) forecast with SVRs four different exchange rates and demonstrate their superiority over NNs. In a similar task, Huang *et al.* (2010) generates a hybrid chaos-based SVR algorithm which performs better than chaos-based NNs and traditional non-linear models. Hsu *et al.* (2009) integrate SVR in a two-stage architecture for stock price prediction and present empirical evidence that show that its forecasting performance can be significantly enhanced compared to a single SVR model. Lu *et al.* (2009) use independent component analysis to select the inputs of their SVR models and predicts the Nikkei 225 opening index and the TAIEX closing index. On the other hand, Bao *et al.* (2005) and Lin and Pai (2010) apply fuzzy SVR models to the task of forecasting the Shanghai Stock Exchange index and business cycles respectively. Kim and Sohn (2010) apply successfully SVM in default prediction of small and medium enterprises. Yeh *et al.* (2011) develop a two-stage multiple-kernel learning algorithm for SVRs and forecast the TAIEX index. Yao *et al.* (2015) evaluate SVRs over the task of predicting recovery rates of defaulted corporate instruments between 1985 and 2012. The results show the superiority of the SVRs over other commonly used methods, such as linear regression, fractional response regression and the two-stage methodology. Geng *et al.* (2015) compares NNs, SVM, decision trees and majority voting classifiers in a forecasting competition. All algorithms are predicting the financial distress of listed Chinese companies. In this study, NNs outperform the SVM, but it is acknowledged that this is contradicting previous literature. Rosillo *et al.* (2014) simulate the S&P 500 movements with GARCH and Support Vector Machines while Sermpinis *et al.* (2015) forecast three exchange rates with genetically optimized SVRs. Finally, Yao *et al.* (2015) estimate accurately the recovery rates of corporate bonds with SVRs.

In the field of wSVR, there are only a handful of applications in Science. Huang *et al.* (2006) apply wSVR to the task of forecasting three stock indices. The statistical evaluation indicates the superiority of wSVR compared to a simple SVR model. Chuang (2007) combines wSVR with fuzzy logic. The proposed approach proves seems more accurate and computationally efficient than more traditional SVR algorithms. Huang and Shen (2008) apply wSVR at a 1-dimensional data set simulation. Elattar *et al.* (2010) estimates successfully the electric load forecasting with the LSVR. In this approach, the weighted distance of the algorithm is based on the Mahalanobis distance. Suykens *et al.* (2002) presents a sparse approximation procedure for weighted and unweighted for least squared support vector machines (SVM). Jiang and He (2012) forecast financial series with SVR and a Grey relational grade weighting function. Their model seems able to forecast accurately the Shenzhen Stock and the Shanghai Stock Exchange indices.

To the best of our knowledge, this is the first study that applies wSVR to a financial trading application. Additionally, compared to previous studies, we will attempt to answer the kernel question. The vast majority of studies on SVRs in Finance apply the Radial Basis Function (RBF) as kernel. In addition to RBF, this research is extended to two recently introduced promising kernels, the Wavelet and the Mahalanobis kernels. For both kernels there are limited applications in SVRs although they possess several advantages. In wSVR studies, there is none that applies the Wavelet or the Mahalanobis kernel. Additionally, all models are examined under different market conditions. This provides a useful insight on the robustness of the models under consideration and strengthens the conclusions. It also provides empirical evidence in favour of the Adaptive Market Hypothesis (AMH) of Lo (2004). AMH argues that trading models have different strength under different market conditions. If the market is stressed, the performance of trading rules should be worse compared to normal market conditions. Finally, compared with the majority previous studies in the field, this research follows a robust statistical approach on selecting the SVRs inputs. The performance of a SVR model is crucially depending on the informational capacity of the chosen inputs. In this study, a large pool of linear and nonlinear predictors is generated and the SVR inputs are selected through two statistical procedures, a Principal Component Analysis (PCA) and the Hansen *et al.* (2011) Model Confidence Set (MCS) test. This unique approach aims on increasing the forecasting performance of the SVR models and providing a statistically stable process on SVR input selection.

From the results of this study, it emerges that LSVR present a superior forecasting and trading performance compared to the traditional SVR models. Similarly, the ν-SVR algorithms seem superior from their ε-SVR counterparts for the series and time periods under study. The results concerning the kernel function seem mixed and are not supporting the notion that RBF is superior in simulating financial datasets. Nevertheless, all SVR models seem capable of producing profitable forecasts after transaction costs. Finally, we note the performance of all models deteriorates on the peak of the European debt crisis. It seems that technical models lose their power to produce profitable signals during periods of financial crisis.

The rest of the paper is organized as follows. In Section 2, there is a detail description of this study's dataset. A description the SVR algorithms under study are presented in Section 3. The statistical and trading performance of the implemented models is in Section 4 while the concluding remarks are in Section 5. In the appendix, there is the characteristics of the linear and nonlinear predictors that constitute the SVR input pool and a description of the statistical and trading measures.

2. Dataset

In this study, twelve SVR algorithms are applied to the task of forecasting and trading the logarithmic returns of five ETFs. These ETFs are designed to replicate the performance of the Spanish, French, Italian, German and the whole European Monetary Union (EMU) equities markets. ETFs offer investors the opportunity to trade stock market indices with very low transaction costs[1].The summary statistics of the five return series and the tickers of the ETFs under study for the whole dataset are presented in Table 1 below.

Table 1. Summary Statistics and Tickers

Equity Market	Spain	Italy	France	Germany	EMU
Ticker	EWP	EWI	EWQ	EWG	EZU
Mean	-0.0002	-0.0004	-0.0002	0.0000	-0.0005
Standard deviation	0.0218	0.0218	0.0196	0.0194	0.0254
Skewness	-0.1332	-0.1914	-0.1523	0.0504	-11.1060
Kurtosis	7.2671	6.8587	8.3027	11.4435	315.1631

[1] The transaction costs for the three ETFs tracking their respective benchmarks do not exceed 0.5% per annum for medium size investors (see, for instance, www.interactive-brokers.com).

Jarque-Bera (p value)	0	0	0	0	0
ADF (p value)	0.0001	0.0001	0.0001	0.0001	0.0001

As anticipated, the five returns series exhibit small skewness and high kurtosis. The Jarque-Bera statistic confirms that the three return series are non-normal at the 99% confidence level. The Augmented Dickey-Fuller (ADF) reports that the null hypothesis of a unit root is rejected at the 99% statistical level for all five return series.

All models are evaluated through three different forecasting exercises. In the first forecasting exercise the out-of-sample represents the start of the debt EMU crisis, the second is on the peak of the crisis while the third (and more recent) refers to the end of the crisis, when most EMU countries seem to have stabilize their economies. The intuition behind the selection of these periods is to test the robustness and the profitability of the models under different market conditions. A summary of these periods is presented in Table 2 below.

Table 2. Dataset

Forecasting Exercise	Periods	Dates	Trading Days
1	In-Sample	3 January 2007 to 31 December 2008	504
	Out-of-Sample	3 January 2009 to 31 January 2009	252
2	In-Sample	3 January 2010 to 31 December 2011	504
	Out-of-Sample	3 January 2012 to 31 December 2012	252
3	In-Sample	3 January 2012 to 31 December 2013	504
	Out-of-Sample	3 January 2014 to 31 December 2014	252

All models will be optimized in the in-sample and their forecasts will be evaluated in the out-of-sample.

In order to make the application more realistic, a novel training procedure will be followed. A trader is unlikely to keep the same specification for a model that performs badly in the out-of-sample over a given period. In real world trading environments practitioners re-estimate the parameters of their algorithms when there is a structural brake in their series and their losses exceed a level. In this study, all models will be re-estimated when the maximum drawdown of the trades of a model in the out-of-sample reach or exceeds the -10%. Maximum drawdown, the essence of risk for an investor, represents the maximum consecutive loss over a given period. If the maximum drawdown of trades of a model in the out-of-sample reach or exceeds -10%, the in-sample will be roll forward until this point (in order to include the most recent information) and the parametrization procedure presented in Section 3.5 will be reinitiated.

3. Theoretical Framework

Support Vector Machines (SVM) is a class of powerful supervised learning systems introduced by Boser *et al.* (1992). Their formulation embodies the Structural Risk Minimisation principle which is superior to the traditional Empirical Risk Minimization applied by conventional supervised learning systems (Gunn *et al.*, 1997). SVMs machines have provided promising empirical evidence in classification problems in various aspects of Science (see for example Christianini and Shawe-Taylor (2000)). Their initial use in financial forecasting was limited as most problems require a regression based technique. This changed with the introduction of the *ε*-insensitive loss function and the Support Vector Regressions (SVRs) by Vapnik (1995). SVRs main advantage is their ability to generate nonlinear decision boundaries through linear classifiers, while having a simple geometric interpretation. Additionally, their solution is global and unique and do not suffer from multiple local minima, such as the solution of NNs. This allows them to balance between model accuracy and complexity and to present a remarkable forecasting ability (Kwon and Moon (2007) and Suykens *et al.* (2002)).

3.1 The *ε*-SVR

Let's consider the training data {(x_1,y_1), (x_2,y_2)…, (x_n, y_n)}, where $x_i \in X \subseteq R, y_i \in Y \subseteq R, i = 1...n$ and *n* is the total number of training samples. SVRs seek to estimate functions:

$$f(x) = w^T \phi(x) + b \qquad [1]$$

where *φ(x)* is the feature input of *x* (see Figure 2c) and *w* and *b* are coefficients which estimated by minimizing the regularized risk function:

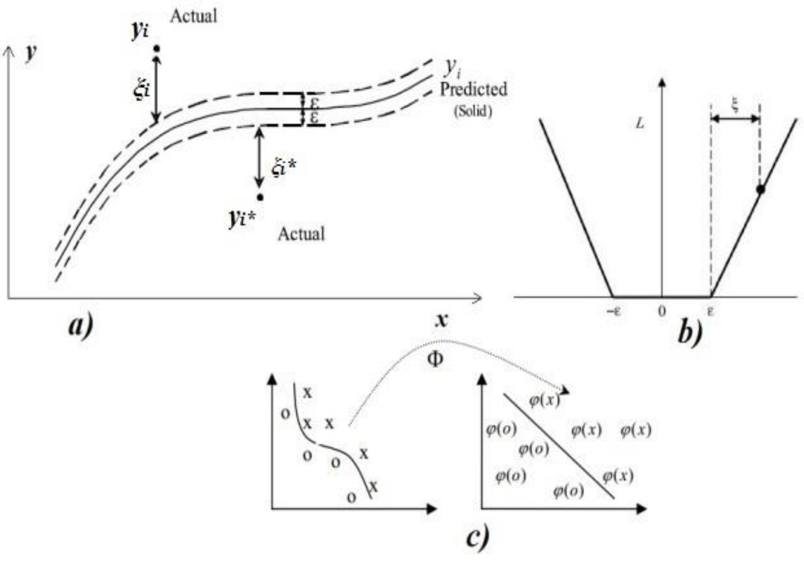
$$R(C) = C\frac{1}{n}\sum_{i=1}^{n} L_e(y_i, f(x_i)) + \frac{1}{2}\|w\|^2$$ [2]

The parameters C and *ε* are user determined parameters, y_i is the actual value at time i while $f(x_i)$is the predicted value at the same period. The function $L_\varepsilon(d_i,y_i)$ is called the *ε*-insensitive loss function and is described below:

$$L_e(y_i, f(x_i)) = \begin{cases} |y_i - f(x_i)| - \varepsilon, if\ |y_i - f(x_i)| \geq \varepsilon \\ 0,\ if\ other \end{cases}$$ [3]

The norm term $\|w\|^2$ characterizes the flatness of the model, C is a constant that determines the trade-off between empirical risk and the model flatness and *ε* is the degree of model noise insensitivity. Equation 3 finds the predicted values that have at most *ε* deviations from the actual obtained values y_i (see Figure 1a and 1b).

Figure 1: a) The f(x) curve of SVR and the ε-tube, b) plot of the ε-sensitive loss function and c) mapping procedure by ϕ*(x)*



If we introduce two positive slack variables ξ_i, ξ_i^* that correspond to the distance of the actual values from the corresponding boundary values of the *ε*-tube (figure 1a), then Equation [2] is transformed to the following argument:

$$\text{Minimize } C\sum_{i=1}^{n}(\xi_i+\xi_i^*)+\frac{1}{2}\|w\|^2 \text{ subject to } \begin{Bmatrix}\xi_i\geq 0\\ \xi_i^*\geq 0\\ C>0\end{Bmatrix} \text{ and } \begin{Bmatrix} y_i-w^T\varphi(x_i)-b\leq +\varepsilon+\xi_i\\ w^T\varphi(x_i)+b-y_i\leq +\varepsilon+\xi_i^*\end{Bmatrix} \qquad [4]$$

The term $\left\{\sum_{i=1}^{n}(\xi_i+\xi_i^*)\right\}$ is the training error, as specified by the slack variables. This optimization problem can be transformed in a dual problem which solution is based on two Langrangian multipliers (a_i and a_i^*) and mapping through a kernel function $K(x_i, x)$:

$$f(x)=\sum_{i=1}^{n}(a_i-a_i^*)K(x_i,x)+b \text{ where } 0\leq a_i, a_i^*\leq C \qquad [5]$$

Factor *b* is estimated following the Karush-Kuhn-Tucker conditions (for a detail mathematical derivation see Vapnik (1995)). Support Vectors (SVs) are called all the x_i that contribute to equation [5], thus they lie outside the ε-tube, whereas non-SVs lie within the ε-tube.[2] Increasing ε leads to more SVs being selected, whereas decreasing it results to more 'flat' estimates'.

3.2. The v-SVR

In the v-SVR, the parameter *ε* is encompassed in the optimization process and controlled with a new parameter $v\in(0,1]$. The motivation for the v-SVR is to simplify the optimization process and create models with more meaningful interpretation (Chang and Lin, 2002). In the v-SVR, the optimization problem is transformed to:

$$\text{Minimize } C\left(v\varepsilon+\frac{1}{n}\sum_{i=1}^{n}(\xi_i+\xi_i^*)\right)+\frac{1}{2}\|w\|^2 \text{ subject to } \begin{Bmatrix}\xi_i\geq 0\\ \xi_i^*\geq 0\\ C>0\end{Bmatrix} \text{ and } \begin{Bmatrix} y_i-w^T\varphi(x_i)-b\leq +\varepsilon+\xi_i\\ w^T\varphi(x_i)+b-y_i\leq +\varepsilon+\xi_i^*\end{Bmatrix} \qquad [6]$$

The methodology is similar as in the *ε*-SVR. Vapnik (1995) showed that with Lagrange multipliers, Equation [6] is a convex optimization problem with a global minimum. The solution takes the form:

$$f(x)=\sum_{i=1}^{n}(a_i-a_i^*)K(x_i,x)+b \text{ where } 0\leq a_i, a_i^*\leq \frac{C}{n} \qquad [7]$$

[2] A SV is either a boundary vector ($(a_i-a_i^*)\in[-C/n, C/n], \xi_i=\xi_i^*=0$) or an error vector ($a_i, a_i^*=C/n \text{ and } \xi_i, \xi_i^*>0$).

Schölkopf *et al.* (1999) demonstrate that an increase of *ε* leads to the proportional increase of the first term of $\left\{ v\varepsilon + \frac{1}{n}\sum_{i=1}^{n}(\xi_i + \xi_i^*) \right\}$, while the second term decreases proportionally to the fraction of points outside the *ε*-tube. Factor *v* can be considered as the upper bound on the fraction of errors. Similarly a decrease in *ε* leads to a proportional change of the first term, but the second term's change is proportional to the fraction of SVs. That means that *ε* will shrink as long as the fraction of SVs is smaller than *v*, therefore *v* is also the lower band in the fraction of SVs.

3.3. The locally weighted SVR

Locally weighted SVR (wSVR) is an extension of the Locally Weighted Regression (LWR) to the SVR methodology. LWR is a simple memory based procedure for fitting a regression surface to the data through multivariate smoothing. It is based on the assumption that the nearest to the predictor values are its best indicators. This is extremely beneficial in problems such as modelling financial trading series where some training points are more important than others and models will have higher accuracy for the training input data that are closer to the new input point for prediction.

LWR can approximate an estimate g(x) of the regression surface for every value x in the dimensional space of the independent variables. Following the suggestions of Cleveland and Devlin (1988) each point of the neighbourhood is weighted according to its distance from point of interest x. The neighborhood is set by estimating the distances of *q* observations x_i from *x*, where $1 \le q \le n$ Those points that are close to x are assigned large weights, while those that are far have small weights. This confirms the local element of the method (Lee *et al.*, 2005).The idea of assigning weights to each point of the dataset could be expressed as:

$$\{(x_i, y_i, w_i)\}_{i=1}^{n}, x_i \in X \subseteq R, y_i \in Y \subseteq R,\ \ 0 \le w_i \le 1 \qquad [8]$$

A quadratic function of the independent variables is fitted to the dependent variable using weighted least squares with these weights. In that way, g(x) is taken to be the value of this fitted function at x. A distance function in the space of the independent variables and a weight function to specify the neighborhood size are needed. The work of Cleveland and

Devlin (1988) provides a detailed description on how to select these. The most common approach and the one followed in this study is to use the ratio *q/n* as a smoothness factor. The practitioner should interpret the smoothing factor rather than the *q*. The reason for that is that increasing the smoothing factor provides a smoother g(x) estimate. The selected weight function is the tricubic one specified below:

$$W(u) = \begin{Bmatrix} (1-u^3)^3, 0 \le u \le 1 \\ 0, otherwise \end{Bmatrix} \quad [9]$$

Based on the above, the weight of each training data (x_i, y_i) is:

$$w_i(x_i) = W\left(\frac{\rho(x, x_i)}{d(x)}\right) \quad [10]$$

where ρ is the Euclidian distance and *d(x)* is the Euclidian distance specifically from the q^{th}-nearest x_i to *x*. [3]

From equation [9] it is verified that $w_i \in [0,1]$. The weight has its maximum value when x_i is closest to *x* and its minimum for the q^{th}-nearest x_i to *x.*

If we apply the principles of the LWR to SVR, we can obtain a locally Weighted SVR (wSVR), where the parameter *C* is not constant, but locally adjustable:

$$C_i = w_i * C \quad [11]$$

In this study, the M plots of Mallows (1973) are applied for the selection of the smoothing ratios. Their utility for the LWR is supported by Cleveland and Devlin (1988) because they illustrate the trade-off between the variance contribution and bias to the error estimate, as the smoothing ratio is changing. Once the optimal smoothing ratio is selected for each point of interest, the estimate g(x) uses *q* observations, whose x_i values are closest to *x* (defined

[3] For instance, when the daily return for ETFs is desired and the expected return for the respective time series is roughly equal to zero, outliers of 5% gain per ticker could be ignored due to major structural change rather than routine behaviour of the time series. In this example *x* is the expected return target when modelling the deviation from this point, the outlier is x_i and w_i is the weight the Euclidean distance assigns to this observation

local neighbourhood). Then, the Euclidian distance between *x* and x_i is scaled over the Euclidian distance between the qth-nearest x_i from *x* (equation [10]). This value is then inserted in the tricubic function (equation [9]) to estimate the final weight and the final appointed C_i (equation [11]).

For the *ε*-SVR, the optimization functions Equation [4] is translated to:

$$\text{Minimize} \sum_{i=1}^{n} C_i(\xi_i + \xi_i^*) + \frac{1}{2}\|w\|^2 \text{ subject to } \begin{Bmatrix} \xi_i \geq 0 \\ \xi_i^* \geq 0 \\ C_i > 0 \end{Bmatrix} \text{ and } \begin{Bmatrix} y_i - w^T\varphi(x_i) - b \leq +\varepsilon + \xi_i \\ w^T\varphi(x_i) + b - y_i \leq +\varepsilon + \xi_i^* \end{Bmatrix} \qquad [12]$$

The v-SVR optimization Equation [6] is now expressed as:

$$\text{Minimize } C_i\left(v\varepsilon + \frac{1}{n}\sum_{i=1}^{n}(\xi_i + \xi_i^*)\right) + \frac{1}{2}\|w\|^2 \text{subject to } \begin{Bmatrix} \xi_i \geq 0 \\ \xi_i^* \geq 0 \\ C_i > 0 \end{Bmatrix} \text{and } \begin{Bmatrix} y_i - w^T\varphi(x_i) - b \leq +\varepsilon + \xi_i \\ w^T\varphi(x_i) + b - y_i \leq +\varepsilon + \xi_i^* \end{Bmatrix} \qquad [13]$$

The traditional SVR attempts to track all training data with a specific model complexity through a constant *C*. Conceptually, this suggests that that the size of ξ_i, ξ_i^* does not fluctuate much. The wSVR, on the other hand, highly penalizes the errors near *x* in an attempt to increase predictability. In that way, wSVR's predictive performance is increasing gradually, as the shape of the weight function is becoming sharper (Lee *et al.*, 2005). In other words, the weight of every point for a traditional regression model is 1/n, meaning that the assigned weight is similar for every point. In LWR the importance/weight increases continuously, once we move from outliers to more central points. If the weight functions for both cases are plotted, the function of the traditional regression would be horizontal line, whereas in LWR the function would have a bell-shape around the central point. Once the importance of distance is increased through higher orders of power, the LWR function plot becomes narrower or in other words sharper. This is a clear point of superiority of LSVR over the non-locally optimized method. For more details and the full mathematical derivation of the weighted *ε*-SVR (ε-wSVR) see Huang and Shen (2008) and for the weighted v-SVR (v-wSVR) see Elattar *et al.* (2010).

3.4. Kernels

In the heart of all SVR algorithmic procedures lies a kernel function $K(x_i, x)$. Its role is to map the data in a higher dimensional space where the problem can be easily solved. Any function that satisfies Mercer's condition by Vapink (1995) can be used as the Kernel function. Different kernels lead to different mapping and thus different regression outputs. Therefore the correct choice of the kernel function is of outmost important. Unfortunately there is no formal way to determine the kernel function a priori[4].In this study, the three most promising nonlinear kernels (the Gaussian Radial Basis Function, the wavelet kernel and the Mahalanobis kernel) will be applied.

The most common non-linear kernel function is the Gaussian Radial Basis Function (RBF) and it is based on the Euclidean distance. It is specified as:

$$K(x_i, x) = \exp(-\gamma \|x_i - x\|^2), \gamma > 0 \qquad [a]$$

where *γ* represents the variance of the kernel function. RBF is a flexible kernel with a single parameter and a simple numerical solution (Hsu *et al.*, 2003).RBF has provided promising empirical results in similar problems (see amongst others Lu *et al.* (2009), Yeh *et al.* (2011) and Dunis *et al.* (2013)).

The Wavelet kernel was introduced by Zhang *et al.* (2004) and comes from the wavelet theory. It takes the form:

$$K(x_i, x) = \Pi h(\frac{x_i - c}{a}) h(\frac{x - c}{a}) \qquad [b]$$

where *a* and *c* are the wavelet dilation and translation coefficients, while h(x) denotes a mother wavelet function. Zhang *et al.* (2004) suggests: $h(x) = \cos(1.75x)\exp(-\frac{x^2}{2})$.

The Mahalanobis kernel is based on the RBF but instead of using the Euclidean distance is applying the Mahalanobis distance which takes in account the correlation among the

[4] Automatic kernel selection has been proposed by several authors (see for example Ali and Smith-Miles (2006) and Howley and Madden (2005)). However overfitting and selection bias dominate such algorithms (Cawley and Talbot, 2010).

features. If we consider that all training data belong to a cluster, the Mahalanobis distance between a datum x and the cluster is: $d(x) = \sqrt{(x-c)^T Q^{-1} (x-c)}$ where c is the center vector and Q is the covariance matrix of the data. The Mahalanobis kernel takes the form:

$$K(x_i, x) = \exp(-\frac{\delta}{m}(x_i - x)^T Q^{-1}(x_i - x)) \qquad [c]$$

where *δ>0* is the scaling factor of the Mahalanobis distance.

3.5. Parametrization

The forecasting performance of the SVRs is heavily dependent on the parameters that the practitioner applies to the algorithm. The input and parameters of the SVR along with the parameters of the kernel function need to be set up in a priori procedure. Different parameters lead to different forecasts. More specifically the practitioner need to select the inputs of the SVR from a pool of potential predictors, parameters *C*, *ε* or *v* for the *ε*-SVR and the *v*-SVR respectively and the parameters of the kernel function (the gamma parameter in the RBF, a and c for the Wavelet and δ and m for the Mahalanobis kernel).

Several different procedures have been proposed for the SVR parametrization. The most popular and one of the simpler approaches is cross validation in the in-sample (see amongst others Ince and Trafalis (2006), Pang *et al.* (2011) and Hsu *et al.* (2009)). Grid search for SVR parametrization has been applied in similar problems with success (see amongst others Hsu *et al.* (2003) and Chen (2014)). Other researchers use variants of the previous two methods (see for example Shi *et al.* (2008) and Huang and Tsai (2009)) or apply heuristics to the task (Pai *et al.*, 2006; Yuan, 2012; Dunis *et al.,* 2013). Unfortunately there is no formal theory behind the selection of the appropriate method. In this study the parameters of the SVR and the parameters of the kernel function will be selected through a 5-fold cross validation (CV) in the in-sample. This approach has provided promising empirical evidence on SVR applications in similar problems (Ince and Trafalis, 2006; Pang *et al.,* 2011; Hsu *et al.,* 2009). Model selection based on CV processes is well accepted in the econometrics literature and shields against in-sample overfitting (see amongst others Andrews (1991), Hansen (2007), An *et al.* (2007) and Arlot and Celisse (2010)).

Additionally, the researcher needs to select the inputs of the SVR. This selection depends on the problem under study and is crucial to the SVR forecasting performance. The SVR inputs should be informationally efficient and independent. Inputs with low or similar information can cripple the algorithm's performance and lead to under-fitting. The SVR studies mentioned in the previous sections applied cross validation, heuristics or objective criteria to select the input series. In this research, the selection will be based on two statistical techniques. At a first stage, a large pool of potential predictors will be created in the in-sample. This pool will comprise a series[5] of Autoregressive (AR), Simple and Exponential Moving Average (MA), Autoregressive Moving Average (ARMA) linear models and eight non-linear algorithms namely a Nearest Neighbours Algorithm (*k*-NN), two Smooth Transition Autoregressive (STAR) models, a Multi-Layer Perceptron (MLP), a Recurrent Neural Network (RNN), a Higher Order Neural Network (HONN), a Psi-Sigma Neural Network (PSN) and a Radial Basis Function Neural Network (RBFNN). Each of these models is applied to the in-sample period and generates a forecast for the same period. Then a PCA is applied to this pool of forecasts in order to reduce its dimension. Several variables in the predictors pool might be correlated and thus offering no extra information. PCA is a statistical method that reduces the dimensions of a dataset with minimal loss of information (for a detail description of PCA see Jolliffe (2002)). The first fifty principal components are selected and fed to the MCS test. The MCS procedure deduces the 'best' models from a full set of models under specified criteria as well as a given level of confidence. In this study, the criterion is the mean square error (MSE) and the level of confidence is set at 90%. MCS is a random data-dependent set of best forecasting models, as a standard confidence interval covers the population parameter, while it acknowledges the limitations of the data (Hansen *et al.*, 2011). Hence, more informative data can lead to only one best model, whilst less informative data result in a MCS including several models, because it is impossible to differentiate between the competing approaches. The relaxed confidence level (90%) allows us to deduct the best forecasters from the predictors' pool in the in-sample and these series

[5] For a description on these methods see Appendix A.

will act as inputs to the SVR algorithm[6]. This input selection approach is unique in the literature and aims to take advantage of some of the latest development in statistics/econometrics. The derived inputs should be informationally independent (through PCA) and superior (through the MCS test)[7]. Both the PCA and the MCS test are theoretically solid methods and their application should offer an advantage compared to the atheoretic approaches that dominate the relevant literature.

The initial selected SVR inputs for all periods under study are presented in Appendix B. Then the parameters of each SVR model are generated through CV in the in-sample. The SVR structural risk minimization principle (Amari et al., 1997) and the CV procedure should protect the models under study against in-sample overfitting.

4. Empirical Application

In the next sections, the statistical and the trading evaluation of the models under study is presented.

4.1 Statistical Performance

The statistical performance of all models in the out-of-sample is presented in table 3[8]. The Root Mean Square Error (RMSE), the Pesaran-Timmermann (PT) (1992) test and the findings from the Diebold Mariano (DM) (1995) test are reported. The PT test examines whether the directional movements of the real and forecast values are the same. In other words, it checks how well rises and falls in the forecasted value follow the actual rises and falls of the time series. The null hypothesis is that the model under study has no power on forecasting the ETF. The DM test checks the null hypothesis of equal predictive accuracy. In this study,

[6] In order to further validate the input selection process, the statistical performance of all models with inputs the first twenty lags of the series under study, the Exponential Moving Averages (EMA) for orders 2 to 20, the k-NN and the RNN was computed. In all cases, the in-sample statistical performance of all models under study was better with the proposed approach.

[7]The PCA analysis ensures that the MCS tests realizations are independent. For example, let us consider two input series that are highly correlated and informationally superior to their counterparts. The MCS test is likely to select both models as SVR inputs. This would cripple the SVRs' performance as the two inputs would offer similar information.

[8] The in-sample statistical performance is not reported for the sake of space. The statistical ranking of the models is similar the out-of-sample. These results are available upon request. From the in-sample performance we note that the statistical accuracy of our models do not differ in great extent from their relevant out-of-sample performance. The average RMSE for all models and periods decrease in the out-of-sample only by 0.00026. This result further validates that the models under study do not suffer from overfitting.

the Mean Squared Error is applied as DM loss function and each model is compared against a random walk[9].

Table 3. Statistical Performance

Forecasting Exercise		EWP	EWI	EWQ	EWG	EZU
3 January 2009 to 31 January 2009	ε-SVRa	0.0054 (6.93)**	0.0056 (7.25)**	0.0060 (6.05)**	0.0055 (7.56)**	0.0052 (7.14)**
	ε-SVRb	0.0056 (7.01)**	0.0055 (6.92)**	0.0061 (5.84)**	0.0059 (7.78)**	0.0058 (6.27)**
	ε-SVRc	0.0052 (6.69)**	0.0054 (7.04)**	0.0058 (6.33)**	0.0057 (7.01)**	0.0055 (6.88)**
	v-SVRa	0.0053 (7.14)**	0.0054 (8.02)**	0.0059 (6.70)**	0.0052 (8.00)**	0.0050 (8.04)**
	v-SVRb	0.0056 (7.29)**	0.0058 (7.78)**	0.0057 (7.15)**	0.0055 (7.71)**	0.0056 (7.25)**
	v-SVRc	0.0052 (7.61)**	0.0050 (7.30)**	0.0058 (7.14)**	0.0054 (7.90)**	0.0052 (7.77)**
	ε-wSVRa	0.0047 (10.54)**	0.0049 (9.86)**	0.0051 (8.90)**	0.0049 (9.26)**	0.0044 (10.48)**
	ε-wSVRb	0.0049 (10.98)**	0.0046 (10.49)**	0.0049 (9.73)**	0.0046 (9.94)**	0.0047 (10.02)**
	ε-wSVRc	0.0047 (9.82)**	0.0050 (10.08)**	0.0047 (9.55)**	0.0045 (10.33)**	0.0048 (10.93)**
	v-wSVRa	0.0044 (11.56)**	0.0048 (10.90)**	0.0049 (10.64)**	0.0047 (10.57)**	0.0042 (11.49)**
	v-wSVRb	0.0046 (10.35)**	0.0046 (9.88)**	0.0045 (11.05)**	0.0048 (10.56)**	0.0045 (10.23)**
	v-wSVRc	0.0047 (11.94)**	0.0045 (11.11)**	0.0044 (10.27)**	0.0044 (10.80)**	0.0044 (10.32)**
3 January 2012 to 31 December 2012	ε-SVRa	0.0066 (5.04)**	0.0071 (4.44)**	0.0076 (4.29)**	0.0068 (6.24)**	0.0061 (6.33)**
	ε-SVRb	0.0070 (4.78)**	0.0074 (4.83)**	0.0078 (4.56)**	0.0065 (6.06)**	0.0064 (5.80)**
	ε-SVRc	0.0067 (5.17)**	0.0072 (4.65)**	0.0074 (5.40)**	0.0069 (5.79)**	0.0062 (6.59)**
	v-SVRa	0.0069 (5.19)**	0.0067 (5.58)**	0.0070 (5.06)**	0.0064 (6.60)**	0.0058 (7.09)**
	v-SVRb	0.0064 (6.09)**	0.0065 (5.29)**	0.0068 (5.74)**	0.0065 (6.41)**	0.0057 (7.31)**
	v-SVRc	0.0065 (6.43)**	0.0062 (6.21)**	0.0066 (6.09)**	0.0064 (6.85)**	0.0055 (7.55)**
	ε-wSVRa	0.0059 (8.37)**	0.0057 (8.73)**	0.0058 (7.96)**	0.0056 (8.44)**	0.0051 (8.77)**
	ε-wSVRb	0.0055 (8.81)**	0.0059 (8.03)**	0.0057 (8.17)**	0.0058 (7.86)**	0.0054 (8.52)**

[9] A simple random walk is defined as: $E(R_t) = \mu + e_t, e_t \sim N(0,1)$ where μ is the in-sample mean.

	ε-wSVRc	0.0055 (8.10)**	0.0056 (8.31)**	0.0059 (7.91)**	0.0059 (7.39)**	0.0050 (9.30)**
	v-wSVRa	0.0052 (9.81)**	0.0050 (8.69)**	0.0054 (8.80)**	0.0054 (9.05)**	0.0049 (10.15)**
	v-wSVRb	0.0054 (9.14)**	0.0053 (9.17)**	0.0051 (9.95)**	0.0056 (8.69)**	0.0050 (9.67)**
	v-wSVRc	0.0053 (9.47)**	0.0057 (9.20)**	0.0053 (8.77)**	0.0057 (8.38)**	0.0048 (9.90)**
3 January 2014 to 31 December 2014	ε-SVRa	0.0060 (5.27)**	0.0062 (5.19)**	0.0065 (5.53)**	0.0063 (6.93)**	0.0059 (6.78)**
	ε-SVRb	0.0059 (5.55)**	0.0065 (4.97)**	0.0069 (5.17)**	0.0066 (6.56)**	0.0059 (6.15)**
	ε-SVRc	0.0064 (5.01)**	0.0061 (5.70)**	0.0070 (4.88)**	0.0065 (6.21)**	0.0060 (6.03)**
	v-SVRa	0.0057 (7.91)**	0.0060 (6.69)**	0.0063 (6.14)**	0.0058 (7.34)**	0.0056 (7.39)**
	v-SVRb	0.0061 (6.84)**	0.0063 (6.11)**	0.0062 (6.44)**	0.0059 (7.11)**	0.0057 (6.81)**
	v-SVRc	0.0057 (7.68)**	0.0062 (6.33)**	0.0063 (6.70)**	0.0057 (7.70)**	0.0055 (7.26)**
	ε-wSVRa	0.0052 (8.71)**	0.0054 (8.37)**	0.0055 (8.05)**	0.0054 (8.38)**	0.0053 (8.12)**
	ε-wSVRb	0.0054 (8.09)**	0.0055 (7.86)**	0.0058 (8.49)**	0.0055 (8.10)**	0.0054 (7.70)**
	ε-wSVRc	0.0055 (8.69)**	0.0052 (8.74)**	0.0056 (8.90)**	0.0054 (8.69)**	0.0051 (8.83)**
	v-wSVRa	0.0049 (10.29)**	0.0050 (9.22)**	0.0051 (9.71)**	0.0051 (9.30)**	0.0046 (9.68)**
	v-wSVRb	0.0051 (9.85)**	0.0053 (8.98)**	0.0054 (9.25)**	0.0052 (9.18)**	0.0048 (9.03)**
	v-wSVRc	0.0050 (10.44)**	0.0051 (9.75)**	0.0052 (9.35)**	0.0049 (9.99)**	0.0047 (9.40)**

Note: The subscript next to each model represents the SVR kernel function. In this table, the RMSE of the SVR model forecasts are presented. The values in the parenthesis are the PT statistics. ** denotes that the DM null hypothesis of equal predictive accuracy is rejected at the 1% significance level.

From the table above, we note that the statistical accuracy of our forecasts deteriorates in the second period (in the peak of the debt crisis). The locally weighted SVRs produce more accurate forecasts than their benchmarks for all periods and ETFs. It seems that the local memory feature of the wSVRs, assisted them in this forecasting exercise. Similarly, the v-SVRs perform better than their ε-SVR counterparts[10]. This implies that the *v*-trick can make the SVR algorithm more efficient. Concerning the kernel function, the results are mixed. It is not possible to clearly distinguish between the forecasting performances of the three different kernels. The RBF and the Mahalanobis kernel seem superior to the Wavelet kernel

[10] In order to confirm these findings, the DM statistics for each forecast pair was computed. The realizations of the DM test (which are provided as supplementary material) confirm our findings.

but this difference is small and not valid in all exercises. The PT statistics indicate that all models are capable of capturing the directional movements of the ETF returns. The DM test reveals that all forecasts do not have equal predictive accuracy with a simple random walk model.

In addition to the tests above, the Hansen's (2005) Superior Predictive Ability (SPA) test is performed. The SPA test compares the relative forecasting performance between multiple methodologies in a full set of models. The null hypothesis is that the benchmark forecast is not inferior to the best alternative one. The benchmark model is a random walk while the set includes the random walk and all the SVR algorithms. As performance measurement, we apply the MSE and the Information ratio. Low p-values indicate that the respective benchmark model is outperformed by at least one alternative, whereas high p-values specify the opposite. The SPA p-values are presented below.

Table 4. SPA test

Forecasting Exercise	**MSE**	**Information Ratio**
3 January 2009 to 31 January 2009	0.001**	0.000**
3 January 2012 to 31 December 2012	0.002**	0.000**
3 January 2014 to 31 December 2014	0.000**	0.000**

Note: ** denotes rejection of the null hypothesis at the 1% significance level.

Our findings demonstrate that a random walk is not superior from our models. We can conclude that the out-of-sample performance of the models under study is genuine.

4.2 Trading Application

In trading applications the practitioner's utmost interest is to produce models that can be translated to profitable trades. Trading profitability is not always synonymous to statistical accuracy. In this section, all models are evaluated through a simple trading strategy. The strategy is to go 'long' when the forecast return is above zero and go or stay 'short' when the forecast return is below zero. Therefore the position is defined by the sign of the forecast. The 'long' and 'short' ETF position is defined as buying or selling the relevant ETF at the current price respectively.

ETFs offer investors' the opportunity to trade stock indices with low transaction costs. For the five ETFs under study, the expense ratio is 0.48% per annum[11]. The annualised return and the Information ratio after transaction costs for all models and periods in the out-of-sample is presented in table 5, while the trading performance measures are described in Appendix C. As mentioned in section 2, all models are re-estimated, when the maximum drawdown of their trades reaches or exceeds the -10%. This approach keeps the maximum consecutive loss of all models close to -10%[12].

Table 5. Trading Performance

Forecasting Exercise		EWP	EWI	EWQ	EWG	EZU
3 January 2009 to 31 January 2009	ε-SVRa	5.71% (1.54)	6.47% (1.49)	5.90% (1.62)	6.63% (1.83)	6.29% (1.75)
	ε-SVRb	6.22% (1.74)	5.83% (1.32)	5.66% (1.53)	6.87% (1.92)	5.61% (1.44)
	ε-SVRc	5.95% (1.62)	6.00% (1.38)	6.18% (1.75)	6.12% (1.61)	5.88% (1.56)
	v-SVRa	6.13% (1.68)	6.89% (1.70)	6.73% (1.99)	7.05% (2.00)	7.15% (2.11)
	v-SVRb	6.57% (1.88)	6.91% (1.71)	7.09% (2.15)	6.70% (1.85)	6.64% (1.90)
	v-SVRc	6.74% (1.92)	6.38% (1.53)	6.62% (1.95)	6.99% (1.99)	6.87% (2.00)
	ε-wSVRa	8.50% (2.61)	7.75% (1.99)	7.52% (2.36)	8.81% (2.75)	9.27% (3.04)
	ε-wSVRb	8.88% (2.75)	8.23% (2.14)	8.34% (2.70)	9.23% (2.95)	8.50% (2.70)
	ε-wSVRc	8.27% (2.56)	8.12% (2.10)	8.20% (2.65)	9.58% (3.08)	8.72% (2.81)
	v-wSVRa	10.21% (3.21)	8.94% (2.39)	9.53% (3.24)	10.44% (3.45)	11.08% (3.81)
	v-wSVRb	9.76% (3.12)	8.70% (2.30)	10.44% (3.65)	10.62% (3.53)	10.57% (3.61)
	v-wSVRc	10.63% (3.45)	9.49% (2.58)	10.03% (3.47)	11.17% (3.76)	10.71% (3.69)
	ε-SVRa	4.48% (0.86)	3.85% (0.63)	4.75% (0.92)	5.08% (0.99)	3.70% (0.52)
	ε-SVRb	4.16% (0.77)	4.20% (0.73)	5.00% (1.01)	4.74% (0.86)	3.71% (0.53)
	ε-SVRc	4.59% (0.91)	4.23% (0.74)	5.11% (1.06)	4.59% (0.80)	4.07% (0.66)

[11] See, https://www.ishares.com/us/index

[12] The only case that models hit the -10% threshold is in the second period (3 January 2012 to 31 December 2012) and for the simple ε-SVR and v-SVR algorithms. This confirms the difficulty that SVRs face on extracting profitable signals on a financial crisis.

3 January 2012 to 31 December 2012	v-SVRa	5.07% (1.08)	4.88% (0.95)	5.60% (1.25)	5.44% (1.12)	4.63% (0.88)
	v-SVRb	5.80% (1.33)	4.60% (0.87)	6.13% (1.45)	5.02% (0.97)	5.14% (1.07)
	v-SVRc	6.32% (1.52)	5.51% (1.16)	6.54% (1.61)	5.78% (1.27)	5.30% (1.14)
	ε-wSVRa	7.25% (1.84)	6.90% (1.63)	7.77% (2.08)	7.31% (1.86)	7.02% (1.80)
	ε-wSVRb	7.71% (2.02)	6.63% (1.55)	7.82% (2.10)	7.00% (1.75)	6.85% (1.73)
	ε-wSVRc	6.96% (1.75)	7.08% (1.70)	7.32% (1.90)	6.83% (1.68)	7.27% (1.90)
	v-wSVRa	8.46% (2.07)	7.94% (1.99)	8.23% (2.24)	8.34% (2.26)	8.08% (2.21)
	v-wSVRb	7.90% (2.25)	8.12% (2.04)	9.42% (2.70)	7.89% (2.09)	7.75% (2.08)
	v-wSVRc	8.21% (2.18)	8.35% (2.14)	8.49% (2.35)	7.60% (1.98)	7.90% (2.12)
3 January 2014 to 31 December 2014	ε-SVRa	5.39% (1.49)	4.93% (1.44)	5.07% (1.55)	5.57% (1.73)	5.26% (1.61)
	ε-SVRb	5.44% (1.51)	5.25% (1.58)	4.70% (1.36)	5.12% (1.51)	4.88% (1.43)
	ε-SVRc	4.86% (1.28)	5.70% (1.77)	4.51% (1.27)	4.93% (1.43)	4.59% (1.28)
	v-SVRa	6.38% (1.79)	6.16% (1.98)	6.40% (2.22)	6.13% (2.01)	6.07% (2.00)
	v-SVRb	6.05% (1.95)	6.00% (1.92)	6.58% (2.31)	5.85% (1.87)	5.72% (1.85)
	v-SVRc	6.54% (1.95)	6.29% (2.04)	6.79% (2.40)	6.29% (2.10)	5.66% (1.81)
	ε-wSVRa	7.70% (2.41)	7.94% (2.75)	7.85% (2.93)	8.01% (2.94)	7.37% (2.66)
	ε-wSVRb	7.29% (2.25)	7.67% (2.63)	8.22% (3.12)	7.75% (2.82)	6.89% (2.42)
	ε-wSVRc	7.42% (2.30)	8.28% (2.91)	7.86% (2.94)	8.33% (3.11)	7.54% (2.74)
	v-wSVRa	9.18% (3.01)	8.58% (3.03)	9.22% (3.61)	9.09% (3.47)	8.93% (3.42)
	v-wSVRb	8.82% (2.89)	8.23% (2.88)	9.34% (3.68)	8.74% (3.30)	8.52% (3.23)
	v-wSVRc	9.63% (3.20)	8.80% (3.12)	9.78% (3.90)	9.91% (3.85)	8.95% (3.45)

Note: The subscript next to each model represents the SVR kernel function. The values in the table represent the annualised return after transaction costs, while the values in the parenthesis are the information ratio after transaction costs.

From the table above, we note that the v-wSVRc model presents the best trading performance for all series and periods under study. Similar with our statistical evaluation, the locally weighted SVRs models provide more profitable trades than the simple SVRs. The v-SVRs generate higher annualised return and information ratio than their ε-SVRs counterparts. This implies that the theoretical properties through the introduction of the memory feature and the *v* factor in the SVR algorithms as presented in Sections 3.2 and 3.3 are being translated to a superior out-of-sample forecasting performance. Concerning the kernel function, it is not possible to distinguish which one fits best. The Mahalanobis kernel seems to provide better trades but not in all cases under study. These results do not confirm the belief that the RBF kernel is appropriate at all financial/trading applications. A preliminary study on the SVR kernels is necessary before each forecasting exercise. It is worth noting, that all models (even the simplest SVR algorithms) are capable of extracting profitable trades. We also note that during the peak of the debt crisis all models present a deteriorating trading performance. It seems that when markets are on turmoil is more difficult to generate profitable algorithms. These results provide to some extent evidence in favour of the AMH. AMH states that the performance of trading rules varies in different periods and during financial crises it is hard to generate profitable trading signals.

5. Conclusions

In this study, a ε-wSVR, a v-wSVR, a ε-SVR and a v-SVR algorithm are generated and applied to the task of forecasting and trading five European ETFs. For each SVR algorithm, the RBF, the Wavelet and the Mahalanobis kernel are applied and tested. The SVR input selection is based on a PCA, the MCS test and a pool of two hundred eighty four linear and non-linear predictors. Concerning the trading application, it is designed to cover the recent EU debt crisis, to provide an insight on the performance of the proposed models under different market conditions and to provide evidence in favour or against the AMH.

Compared to previous studies, this research contributes to the literature with the first application of wSVR algorithm to a financial trading application. It also explores for the first time of the Wavelet and the Mahalanobis kernel in a wSVR framework. Finally, it introduces a novel SVR input selection method while it attempts to provide a comparison between the ε-SVR and the v-SVR techniques.

In terms of the experimental results, the wSVR models seem to outperform the traditional SVR models while the v-SVR algorithms provide more accurate and profitable forecasts than their *ε*-SVRs counterparts. From the three different kernels under study, it is impossible to distinguish the best. Their performance varies between the periods and the series under study. These results do not confirm the golden standard of the RBF kernel in finance studies. Concerning the trading application, we note that the performance of all models varies between periods and deteriorates in the peak of the EU debt crisis. This confirms to some extent the AMH, which states that the performance of trading rules is unstable and it worsens in financial turmoil.

These results should go forward on convincing researchers and academics on exploring different and more complicated SVR algorithms, kernels and SVR input selection techniques. It seems that the SVR performance is highly sensitive on the algorithms parameters. This research supports the notion that extensive experimentation is needed before each SVR business application.

References

Ali, S. and Smith-Miles, K. A. (2006), A meta-learning approach to automatic kernel selection for support vector machines. *Neurocomputing*, 70 (1), 173-186.

Allen, F. and Karjalainen, R. (1999), Using genetic algorithms to find technical trading rules. *Journal of Financial Economics*, 51 (2), 245-271.

Amari, S.I., Murata, N., Müller, K.R., Finke, M. and Yang, H.H. (1997), Asymptotic statistical theory of overtraining and cross-validation. *Neural Networks, IEEE Transactions on*, *8* (5), 985-996.

An, S., Liu, W. and Venkatesh, S. (2007), Fast cross-validation algorithms for least squares support vector machine and kernel ridge regression. *Pattern Recognition*, *40* (8), 2154-2162.

Andrews, D.W. (1991), Asymptotic optimality of generalized C L, cross-validation, and generalized cross-validation in regression with heteroskedastic errors. *Journal of Econometrics*, *47* (2), 359-377.

Arlot, S. and Celisse, A. (2010), A survey of cross-validation procedures for model selection. *Statistics surveys*, *4*, 40-79**.**

Bao, Y. K., Liu, Z. T., Guo, L. and Wang, W. (2005), Forecasting stock composite index by fuzzy support vector machines regression. In *Proceedings of 2005 International Conference on Machine Learning and Cybernetics*, 3, 3535-3540.

Boser, B.E., Guyon, I.M and Vapnik, V. N. (1992), A training algorithm for optimal margin classifiers. In *Proceedings of the fifth annual workshop on Computational learning theory*, 144-152.

Broomhead, D.S. and Lowe, D. (1988), Multivariable functional interpolation and adaptive networks. *Complex Systems*, 2, 321–355.

Cawley, G.C. and Talbot, N.L. (2010), On over-fitting in model selection and subsequent selection bias in performance evaluation. *The Journal of Machine Learning Research*, 11, 2079-2107.

Chang, C.C and Lin, C.B. (2002), Training v-support vector regression: Theory and algorithms. *Neural Computation*, 14 (8), 1959-1977.

Chan, K.S. and Tong, H. (1986), On Estimating Thresholds in Autoregressive Models, *Journal of Time Series Analysis*, 7 (3), 178–190.

Chen, M. Y. (2014), Using a hybrid evolution approach to forecast financial failures for Taiwan-listed companies. *Quantitative Finance*, 14(6), 1047-1058.

Chuang, C. C. (2007), Fuzzy weighted support vector regression with a fuzzy partition*, Part B: Cybernetics, IEEE Transactions on Systems, Man, and Cybernetics* , 37 (3), 630-640.

Christianini, N. and Shawe-Taylor, J. (2000), *An introduction to support vector machines and other kernel-based learning methods*. Cambridge University Press.

Cleveland, W.S. and Devlin, S.J. (1988), Locally Weighted Regression: An Approach to Regression Analysis by Local Fitting, *Journal of the American Statistical Association*, 83 (403), 596-610.

Diebold, F.X. and Mariano R.S. (1995), Comparing predictive accuracy. *Journal of Business and Economic Statistics*, 13 (3), 253–263.

Dunis, C., Laws, J. and Sermpinis, G. (2010), Modeling and Trading the EUR/USD Exchange Rate at the ECB fixing. *The European Journal of Finance*, 16 (6), 541-560.

Dunis, C., Laws, J. and Sermpinis, G. (2011), Higher order and recurrent neural architectures for trading the EUR/USD exchange rate. *Quantitative Finance*, 11 (4), 615-629.

Dunis, C., Likothanassis, S., Karathanasopoulos, A., Sermpinis, G. and Theofilatos, K. (2013), A hybrid genetic algorithm–support vector machine approach in the task of forecasting and trading. *Journal of Asset Management*, 14 (1), 52-71.

Elattar, E.E., Goulermas, J.Y. and Wu, Q.H. (2010), Electric load forecasting based on locally weighted support vector regression. *IEEE Transactions on Systems, Man, and Cybernetics. Part C: Application and Reviews*, 40 (4), 438–447.

Fix, E. and Hodges, J.L. (1951), Discriminatory analysis, nonparametric discrimination: Consistency properties. Technical Report 4, USAF School of Aviation Medicine, Randolph Field, Texas.

Frisman, R. (2001), Estimating the Value of Political Connections. *The American Economic Review*, 91 (4), 1095-1102.

Froot, K., Schaferstein, D. and Stein, J. (1992), Herd on the street: Informational inefficiencies in a market with short-term speculation. *Journal of Finance*, 47 (4), 1461–1484.

Geng, R., Bose, I. and Chen, X. (2015), Prediction of financial distress: An empirical study of listed Chinese companies using data mining. *European Journal of Operational Research*, 241 (1), 236-247.

Ghosh, J. and Shin, Y. (1991), The Pi-Sigma Network: An efficient Higher-order Neural Networks for Pattern Classification and Function Approximation. *Proceedings of International Joint Conference of Neural Networks*, 1, 13-18.

Gunn, S.R., Brown, M. and Bossley, K.M. (1997), Network performance assessment for neurofuzzy data modelling. In Liu, X., Cohen, P. and Berthold, M.: *Advances in Intelligent Data Analysis*, *Lecture Notes in Computer Science*, 1280, 313–323.

Hansen, B.E. (2007), Least squares model averaging. *Econometrica*, *75*(4), 1175-1189.

Hansen P.R. (2005), A Test for Superior Predictive Ability, *Journal of Business & Economic Statistics*, 23 (4), 365-380.

Hansen, P. R., Lunde, A. and Nason, J. M. (2011), The model confidence set. *Econometrica*, *79* (2), 453–497.

Howley, T. and Madden, M. G. (2005), The genetic kernel support vector machine: Description and evaluation. *Artificial Intelligence Review*, 24 (3-4), 379-395.

Hsu, C.W, Chang, C.C. and Lin, C.J. (2003), A practical guide to support vector classification. Technical report, Department of Computer Science and Information Engineering, National Taiwan University.

Hsu, S. H., Hsieh, J. P. A., Chih, T. C. and Hsu, K. C. (2009), A two-stage architecture for stock price forecasting by integrating self-organizing map and support vector regression. *Expert Systems with Applications*, *36*(4), 7947-7951.

Huang, K., Yang, H., King, I. and Lyu, M. R. (2006), Local support vector regression for financial time series prediction. In *Neural Networks, 2006. IJCNN'06. International Joint Conference on Neural Networks*, 1622-1627.

Huang, W. and Shen, L. (2008), Weighted support vector regression algorithm based on data description. *ISECS International Colloquium on Computing, Communication, Control, and Management,* 1, 250-254.

Huang, C. L. and Tsai, C. Y. (2009), A hybrid SOFM-SVR with a filter-based feature selection for stock market forecasting. *Expert Systems with Applications*, 36 (2), 1529-1539.

Huang, S.C., Chuang, P.J., Wu, C.F. and Lai, H.J. (2010), Chaos-based support vector regressions for exchange rate forecasting. *Expert Systems with Applications*, 37 (12), 8590-8598.

Huck, N. and Guégan, D. (2005), On the Use of Nearest Neighbors to Forecast in Finance. *Revue de l'Association française de Finance*, 26, 67-86.

Ince, H. and Trafalis, T. B. (2006), A hybrid model for exchange rate prediction. *Decision Support Systems*, *42* (2), 1054-1062.

Jasic, T. and Wood, D. (2004), The profitability of daily stock market indices trades based on neural network predictions: case study for the S&P 500, the DAX, the TOPIX and the FTSE in the period 1965–1999. *Applied Financial Economics*, 14 (4), 285-297.

Jiang, H. and He, W. (2012), Grey relational grade in local support vector regression for financial time series prediction. *Expert Systems with Applications*, 39 (3), 2256-2262.

Jolliffe, I. (2002), *Principal component analysis*. John Wiley & Sons.

Kaastra, I. and Boyd, M. (1996), Designing a Neural Network for Forecasting Financial and Economic Time Series. *Neurocomputing*, 10(3), 215-236.

Kim, H.S. and Sohn, S.Y. (2010), Support vector machines for default prediction of SMEs based on technology credit, *European Journal of Operational Research*, 201 (3), 838-846.

Kwon, Y.K. and Moon, B.R. (2007), A Hybrid Neurogenetic Approach for Stock Forecasting. *IEEE Transactions on Neural Networks,* 18 (3), 851-864.

LeBaron, B. (2000), The stability of moving average technical trading rules on the Dow Jones Index. *Derivatives Use, Trading and Regulation*, 5 (4), 324-338.

Lee, D.E., Song, J.H., Song, S.O., Yoon, E.S. (2005), Weighted support vector machine for quality estimation in the polymerization process, *Industrial & Engineering Chemistry Research*, 44 (7), 2101-2105.

Lin, K. P., & Pai, P. F. (2010), A fuzzy support vector regression model for business cycle predictions. *Expert Systems with Applications*, 37 (7), 5430-5435.

Lin, C.F.J. and Teräsvirta T. (1994), Testing the Constancy of Regression Parameters against Continuous Structural Changes. *Journal of Econometrics*, 62 (2), 211-228.

Lo, A.W. (2004), The adaptive markets hypothesis, Journal of Portfolio Management, 30 (5), 15–29.

Lu, C.J., Lee, T.S. and Chiu, C.C. (2009), Financial time series forecasting using independent component analysis and support vector regression, *Decision Support Systems*, 47 (2),115-125.

Mallows, C.L. (1973), Some comments on $C_{p,}$ *Technometrics*, 15(4), 661-675.

Pai, P. F., Lin, C. S., Hong, W. C. and Chen, C. T. (2006), A Hybrid Support Vector Machine Regression for Exchange Rate Prediction, *International Journal of Information and Management Sciences*, 17 (2), 19-32.

Pang, S., Song, L. and Kasabov, N. (2011), Correlation-aided support vector regression for forex time series prediction. *Neural Computing and Applications*, 20 (8), 1193-1203.

Pesaran, M.H. and Timmerman, A.G. (1992), A Simple Nonparametric Test of Predictive Performance. *Journal of Business and Economic Statistics*, 10(4), pp. 461-465.

Qi, M. and Wu, Y.(2006), Technical Trading-Rule Profitability, Data Snooping, and Reality Check: Evidence from the Foreign Exchange Market. *Journal of Money, Credit and Banking*, 38(8), 2135–2158.

Rosillo, R., Giner J. and De la Fuente D. (2014), Stock Market Simulation Using Support Vector Machines, *Journal of Forecasting*, 33 (6), 488–500.

Schölkopf, B., Bartlett, P. L., Smola, A. J. and Williamson, R. (1999), Shrinking the tube: a new support vector regression algorithm, In: Kearns, M.J., (ed.), Advances in neural information processing systems 11. Cambridge, Mass, MIT Press, 330-336.

Sermpinis, G., Stasinakis, C., Theofilatos, K. and Theofilatos, K. (2015), Modeling, forecasting and trading the EUR exchange rates with hybrid rolling genetic algorithms—Support vector regression forecast combinations. *European Journal of Operational Research*, 247 (3), 831–846.

Sermpinis, G., Verousis, T. and Theofilatos, K. (2016), Adaptive Evolutionary Neural Networks for Forecasting and Trading without a Data-Snooping Bias. *Journal of Forecasting*, 35 (1), 1-12.

Shapiro, A. F. (2000), A Hitchhiker's guide to the techniques of adaptive non-linear models. *Insurance: Mathematics and Economics*, 26 (2-3), 119-132.

Shi, Q. S., Zhang, C. H. and Cui, N. X. (2008), Estimation of battery state-of-charge using v-support vector regression algorithm. *International Journal of Automotive Technology*, 9 (6), 759-764.

Suykens, J. A. K., De Brabanter, J., Lukas, L. and Vandewalle, L. (2002), Weighted least squares support vector machines: robustness and sparse approximation. *Neurocomputing,* 48 (1-4), 85-105.

Tenti, P. (1996), Forecasting foreign exchange rates using recurrent neural networks. *Applied Artificial Intelligence*, 10 (6), 567-582.

Vapnik, V. (1995), The nature of statistic learning theory. Springer-Verlag, New York.

Zhang, G., Patuwo, B.E. and Hu, M.Y. (1998), Forecasting with artificial neural networks: The state of the art. *International Journal of Forecasting*, 14 (1), 35-62.

Zhang, L., Zhou, W. and Jiao, L. (2004), Wavelet support vector machine. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 34 (1), 34-39.

Yao, X., Crook, J. and Andreeva, G. (2015), Support vector regression for loss given default modeling, *European Journal of Operational Research*, 240 (2), 528-538.

Yeh, C.Y., Huang, C.W. and Lee, S.J. (2011), A multiple-kernel support vector regression approach for stock market price forecasting. *Expert Systems with Applications*, 38(3), 2177-2186

Yuan, F.C. (2012), Parameters Optimization Using Genetic Algorithms in Support Vector Regression for Sales Volume Forecasting. *Applied Mathematics*, 3 (1), 1480-1486.

Appendix

A. Predictors' pool

In this section, a brief description on the series that comprise the predictors' pool is provided. The predictor's pool is generated in the in-sample. From this pool, the SVRs' inputs are selected through a PCA and the MCS test in the in-sample. The pool is consisted by two hundred seventy six linear and eight non-linear forecasters. Matlab R2010a and a PC with Intel Processor I7 and 8GB RAM were used in all simulations.

A.1. Linear Predictors

The linear predictors are consisted by a series of SMA, EMA, AR and ARMA models. A description of these forecasters is on Table A.1 below.

Table A.1: Linear Predictors

LINEAR MODELS	DESCRIPTION	TOTAL INDIVIDUAL FORECASTS
SMA (*q*)	$E(R_t)=(R_{t-1}+\ldots+R_{t-q})/q$ *Where:* • *q=3...25*	23
EMA (*q'*)	$E(R_t)=\frac{R_{t-1}+(1-a')R_{t-2}+\ldots+(1-a')^{q'-1}R_{t-q'}}{a'+(1-a')+\ldots+(1-a')^{q'-1}}$ *Where:* • *q'=3...25* • *a'*=2/(1+N*days*), *Ndays is the number trading days*	23
AR (*q''*)	$E(R_t)=\beta_0+\sum_{i'=1}^{q''}\beta_{i'}R_{t-i'}$ *Where:* • *q''=1,...,20* • $\beta_0,\beta_{i'}$ *the regression coefficients*	20
ARMA (*m', n'*)	$E(R_t)=\bar{\varphi}_0+\sum_{j'=1}^{m'}\bar{\varphi}_{j'}R_{t-j'}+\bar{a}_0+\sum_{k'=1}^{n'}\bar{w}_{k'}\bar{a}_{t-k'}$ *Where:* • *m', n'=1,..,15* • $\bar{\varphi}_0,\bar{\varphi}_{j'}$ *the regression coefficients* • $\bar{a}_0,\bar{a}_{t-k'}$ *the residual terms* • $\bar{w}_{k'}$ *the weights of the residual terms*	210

The computational cost of the linear predictors for the series under study does not exceed seven minutes.

A.2. Non-Linear Predictors

A brief summary of the seven non-linear predictors is presented in the next sections.

A.2.1. Nearest Neighbors Algorithm (*k*-NN)

Nearest Neighbors is a class of non-linear and non-parametric models based on the work of Fix and Hodges (1951). They are based on the idea that pieces of time series in the past have patterns which might have resemblance to pieces in the future. Similar patterns of behavior are located in terms of nearest neighbors using the Euclidean distance and these patterns are used to predict behavior in the immediate future. It only uses local information to forecast and makes no attempt to fit a model to the whole time series at once. In this study, we followed the guidelines of Huck and Guégan (2005) in modelling the *k*-NN algorithm. The researcher has to define three parameters: the number of neighbors *K*, the length of the nearest neighbor's pattern *m* and the weighting of final prices in a neighbor *α'*. Huck and Guégan (2005) suggest that *K* and *m* should be dependent on the size of the information set. Parameter m should be chosen through the interval:

$$m = [R(\ln(T)), R(\ln(T)+2)] \qquad \text{[A.1]}$$

where *R* is the rounding function rounding to the immediate lower figure and *T* the size of the in-sample dataset. Parameter K should be approximately twice the value of *m*.

The computational cost of the k-NN algorithm for the series under study does not exceed five minutes.

A.2.2. Smooth Transition Autoregressive models (STAR)

STAR models were introduced by Chan and Tong (1986). A STAR algorithm combines two AR models with a function that defines the degree of non-linearity (smooth transition function). In general, they take the form:

$$E(R_t) = \Phi_1' \mathrm{X}_t (1 - F'(z_t', \zeta', \lambda')) + \Phi_2' \mathrm{X}_t F'(z_t', \zeta', \lambda') + u_t' \qquad \text{[A.2]}$$

where $\Phi_i{}' = (\tilde{\varphi}_{i,0}, \tilde{\varphi}_{i,1}, ...\tilde{\varphi}_{i,p}),\ i = 1,2$ and $\tilde{\varphi}_{i,0}, \tilde{\varphi}_{i,1}, ...\tilde{\varphi}_{i,p}$ the regression coefficients of the two AR models, $\mathrm{X}_t = (1, \tilde{\chi}_t')'$ with $\tilde{\chi}_t' = (\mathrm{R}_{t-1}, ..., \mathrm{R}_{t-p"}),\ 0 \le F'(z'_t, \zeta', \lambda') \le 1$ the smooth transition function, $z'_t = R_{t-d'}, d' > 0$ the lagged endogenous transition variable, ζ' the parameter that defines the smoothness of the transition between the two regimes, λ' the threshold parameter and $u_{t'}$ the error term. The most popular smooth transition functions are the first order logistic function (LSTAR) and the exponential function (ESTAR). In this study, the two-resime LSTAR and ESTAR are explored. The construction of the algorithms is based on the guidelines of Lin and Teräsvirta (1994). For both models the orders 1 to 20 are explored.

The computational cost of both STAR algorithms for the series under study does not exceed five minutes.

A.2.3. Neural Networks (NNs)

NNs is a class of artificial intelligence models inspired by the work and functioning of biological neurons. They exist in different forms and architectures. The simpler and most popular NN is the Multi-Layer Perceptron (MLP). A standard MLP has at least three layers. The first layer is called the input layer (the number of its nodes corresponds to the number of explanatory variables). The last layer is called the output layer (the number of its nodes corresponds to the number of response variables). An intermediary layer of nodes, the hidden layer, separates the input from the output layer. Its number of nodes defines the amount of complexity the model is capable of fitting. In addition, the input and hidden layer contain an extra node called the bias node. This node has a fixed value of one and has the same function as the intercept in traditional regression models. Each node of one layer has connections to all the other nodes of the next layer.

The network processes information as follows: the input nodes contain the value of the explanatory variables. Since each node connection represents a weight factor, the information reaches a single hidden layer node as the weighted sum of its inputs. Each node of the hidden layer passes the information through a non-linear activation function and passes it on to the output layer if the calculated value is above a threshold. The training of the network (which is the adjustment of its weights in the way that the network maps the input value of the training data to the corresponding output value) starts with randomly chosen weights and proceeds by applying a learning algorithm called back-propagation of errors (Shapiro, 2000)[13]. The learning algorithm tries to find those weights which minimize an error function (in our case the sum of all squared differences between target and actual values). Since networks with sufficient hidden nodes are able to learn the training data (as well as their outliers and their noise) by heart, it is crucial to stop the training procedure at the right time to prevent overfitting (this is called 'early stopping'). This is achieved by dividing the dataset into three subsets respectively called the training and test sets (the 80% and the 20% of the in-sample respectively) used for simulating the data currently available to fit and tune the model and the validation set used for simulating future values. The network parameters are then estimated by fitting the training data using the backpropagation of errors. The iteration length is optimized by maximizing the forecasting accuracy for the test dataset. Then the predictive value of the model is evaluated applying it to the validation dataset (out-of-sample dataset).

In addition to the MLP network, a Recurrent Neural Network (RNN) is also applied. A simple RNN has an activation feedback which embodies short-term memory. In the RNN, the inputs are (potentially) taken from all previous values. On the one hand, this leads to more connections, more memory during simulations than standard MLPs and substantially more computational time (see Tenti (1996)). On the other hand, the additional memory leads to a superior forecasting performance compared to standard backpropagation networks. The third NN model included in the feature space is the Higher Order Neural Network (HONN). HONNs are able to simulate higher frequency, higher order non-linear data, and consequently provide superior simulations. They provide some rationale for the simulations they produce and thus can be regarded as "open box" rather than "black box". Their main disadvantage is that the required number of weights increases exponentially with the number of inputs. Psi Sigma Networks (PSNs) are considered as a class of feed-forward fully connected HONNs. First introduced by Ghosh and Shin (1991), the PSN creation was motivated by the need to create a network combining the fast learning property of single layer networks with the powerful mapping capability of HONNs, while avoiding the

[13] Backpropagation networks are the most common multi-layer networks and are the most commonly used type in financial time series forecasting (Kaastra and Boyd, 1996).

combinatorial increase in the required number of weights. In these networks the weights from the hidden to the output layer are fixed to 1 and only the weights from the input to the hidden layer are adjusted, something that greatly reduces the training time. Moreover, the activation function of the nodes in the hidden layer is the summing function, while the activation function of the output layer is a sigmoid. The order of the network in the context of PSN is represented by the number of hidden nodes. A Radial Basis Function Neural Network (RBFNN) is a feedforward neural network where hidden units do not implement an activation function, but a radial basis function. An RBFNN approximates a desired function by superposition of non-orthogonal, radially symmetric functions. They have been proposed by Broomhead and Lowe (1988) as an approach to improve accuracy of artificial neural networks while decreasing training time complexity. Compared to the classical MLPs, they faster convergence, smaller extrapolation errors and higher reliability. For more details on the MLP and the HONN see Dunis *et al.* (2010 and 2011). An analysis of the RNN architecture is provided by Tenti (1996) and Ghosh and Shin (1991) provide a detail insight on the PSN model. More details on the RBFNN can be found on Broomhead and Lowe (1988)[14].

There is no formal theory behind the selection of the NN inputs and their parameters (number of hidden neurons, learning rate, momentum and iterations). For that reason, we conduct NN experiments and a sensitivity analysis on a pool of lags of the forecasted series in the in-sample dataset. For example, regarding the number of iterations we started from 5.000 iterations and stopped at the 100.000 iterations, increasing in each experiment their number by 5.000. This is a very common approach in the literature (Tenti, 1996; Zhang *et al.*, 1998). Based on these experiments and the sensitivity analysis, the sets of variables and parameters selected are those that provide the higher trading performance for each network in each in-sample period. For example, the inputs set of the five NNs in the first forecasting exercise for the EWP ETF is presented in table A.2 below.

[14] In this study, we follow the guidelines of Broomhead and Lowe (1988), Tenti (1996), Ghosh and Shin (1991) and Dunis *et al.* (2010 and 2011) in modelling our NNs. For the sake of space, the analysis of the NNs architectures is brief. More details on the training and architecture of these models is available upon request.

Table A.2: Neural Network Inputs

	MLP	RNN	HONN	PSN	RBFNN
EWP	EWP (1)	EWP (1)	EWP (1)	EWP (1)	EWP (1)
	EWP (2)	EWP (2)	EWP (2)	EWP (2)	EWP (2)
	EWP (4)	EWP (3)	EWP (3)	EWP (4)	EWP (3)
	EWP (5)	EWP (5)	EWP (4)	EWP (5)	EWP (5)
	EWP (7)	EWP (6)	EWP (6)	EWP (6)	EWP (6)
		EWP (8)	EWP (8)	EWP(7)	
				EWP(9)	

Note: The value in the parenthesis represents the order of the lag series

For the same period and series, the parameters of the NNs models are presented in table A.3.

Table A.3: Neural Network Inputs

	PARAMETERS	MLP	RNN	HONN	PSN	RBFNN
EWP	Learning algorithm	Gradient descent	Gradient descent	Gradient descent	Gradient descent	k-means clustering
	Learning rate	0.004	0.002	0.4	0.3	-
	Momentum	0.005	0.003	0.5	0.4	-
	Iteration steps	40000	30000	20000	20000	2000
	Initialisation of weights	N(0,1)	N(0,1)	N(0,1)	N(0,1)	N(0,1)
	Input nodes	5	6	6	7	5
	Hidden nodes	3	4	3	4	3
	Output node	1	1	1	1	1

The computational cost for all NN models for the series under study does not exceed twenty minutes.

B. Initial SVR Inputs

Table B.1 presents the initial set of SVR inputs selected for each forecasting exercise. The selection process (see Section 3.5) depends on the forecasting power of each predictor and not on the individual SVR characteristics. Thus, the initial sets of inputs are the same for each SVR algorithm. However, based on the trading performance of each SVR, the set of inputs might change. This depends whether the consecutive losses of a SVR reach or exceed the -10% threshold.

Table B.1: SVR Inputs

Forecasting Exercise	EWP	EWI	EWQ	EWG	EZU
1	RNN, RBFNN, *k*-NN	RNN, RBFNN, LSTAR	RNN, *k*-NN, RBFNN , MLP	RNN, *k*-NN, MLP	RNN, RBFNN , LSTAR
2	RNN, MLP, *k*-NN, ESTAR	MLP, *k*-NN, PSN, LSTAR	RNN, RBFNN, *k*-NN	RNN, *k*-NN, PSN, LSTAR	RNN, RBFNN, k-NN
3	RNN, RBFNN, *k*-NN	RNN, RBFNN, LSTAR	RNN, *k*-NN, LSTAR	RNN, RBFNN, HONN, *k*-NN	RNN, MLP, LSTAR

From the table above, it seems that the input selection process excludes all linear predictors. From the non-linear models, the RNN is the most popular and the ESTAR, HONN and the PSN the less popular choices.

C. Trading Performance Measures

The trading performance measures are described in Table C.1 below.

Table C.1: Trading Performance Measures

TRADING PERFOMANCE MEASURES	DESCRIPTION
Annualized Return	$R^A = (252 * \frac{1}{n'} \sum_{i=1}^{n'} (R_\tau)) - C^A$ where R_τ the daily return and $C^A$is the annualized transaction cost
Annualized Volatility	$\sigma^A = \sqrt{252} * \sqrt{\frac{1}{n'-1} \sum_{i=1}^{n'} (R_i - \bar{R})^2}$ where $\bar{R}$ is the mean return
Information Ratio	$IR = \frac{R^A}{\sigma^A}$
Maximum Drawdown	Maximum negative value of $\sum R_\tau$over the period $MD = \underset{\bar{i}=1,\cdots,\tau;\tau=1,\cdots,N}{Min} \left(\sum_{j=\bar{i}}^{\tau} R_{\bar{j}} \right)$