Business Intelligence

– SEMINAR WINTER SEMESTER 2013/2014 –

Support Vector Machines for Oil Price Prediction

– SEMINAR PAPER –

Submitted by:
Student-ID:
Advisor: **Prof. Dr. Dirk Neumann**

Table of Contents

1. Introduction

Support Vector Machines (SVM) are a class of supervised learning algorithms initially proposed by Vladimir Vapnik and colleagues in the 1960s for pattern recognition. The main formulation is based on the paper (Vapnik, Cortes 1995). Derived from statistical learning theory, support vector machines are used for classification and have recently been extended for regression problems and novelty detection.

Support Vector machines are one of the most effective classification algorithms in machine learning, being successfully applied in tasks such as: handwriting recognition, face detection in images or text categorization. For the regression case, SVMs show excellent performance in the field of time series forecasting, mostly for the energy and financial sectors (Müller et al. 1997). Although they have good generalization performance, SVMs are rather slow in the test phase (Burges 1998).

SVMs are based on the structural risk minimization principle. This principle estimates a function by minimizing the upper bound of a generalization error or, equivalently by maximizing the generalization error (Huson 2007). Thus, SVM achieves a better generalization performance (e.g. low error rate on test set) as compared to other neural networks (Tay, Cao 2001). Moreover, linear classifiers like the perceptron are efficiently trainable, but have low capacity and can be used only for non-symbolic instances. In contrast, non-linear classifiers like neural networks have high capacity, however, they face problems like high time complexity, local optima and over-fitting (Siebers). SVMs solve the over-fitting problem, because training the SVM is equivalent to a quadratic optimization problem which provides a solution that is unique and globally optimal. Furthermore, SVMs deal with nonlinear features without explicitly calculating them by mapping the data into a higher dimensional space by the use of a "kernel trick", and this feature allows SVM to predict well. Geometrically, SVM uses the optimal hyper plane with the maximum margin to separate between two classes of data (Shalizi 2009).

The support vector machine can also be extended for regression tasks, by using different types of loss functions. In this case, the regression is referred to as "Support Vector Regression" (SVR). In regression, numerical values for the output are predicted instead of predicting classes (2002).

This paper aims to provide a brief introduction for the SVM/SVR and observe how it can be applied in practice with a prediction task in statistical software R. The paper is organized as follows. Section 2 provides the theoretical concepts behind SVM for the task of classification, both for the linear and non-linear case. Section 3 covers the case of regression. Section 4 describes an application of SVM for prediction oil price in R. Section 5 presents the conclusions of the paper.

2. Theory of Support Vector Machines

The three main ideas behind Support Vector Machines are: the concept of margin maximization, dual representation and the kernel trick. Margin maximization assures that the best linear separator is used. Formulating the optimization as in the dual form solves the margin maximization problem and kernels allow the representation of non-linear functions using linear separators (Shalizi 2009). Each of these concepts are introduced below.

2.1 Motivation

In the following, we consider a binary classification problem, where two known class labels are given and the objective is to find to which class a new point belongs to. This can be depicted in Fig. 1, where there are given two class labels, respectively positive (purple circle) and negative (red square). The starting point of SVM are linear classifiers, or to illustrate the intuition, one linear classifier known as the perceptron, an algorithm that takes several inputs and gives one output out of several possible. The perceptron divides the input into several regions. If there are only two classes, the straightforward way to separate the data is through a line. In Fig. 1, several separating lines (A,B,C or D) make it possible to separate the two classes.

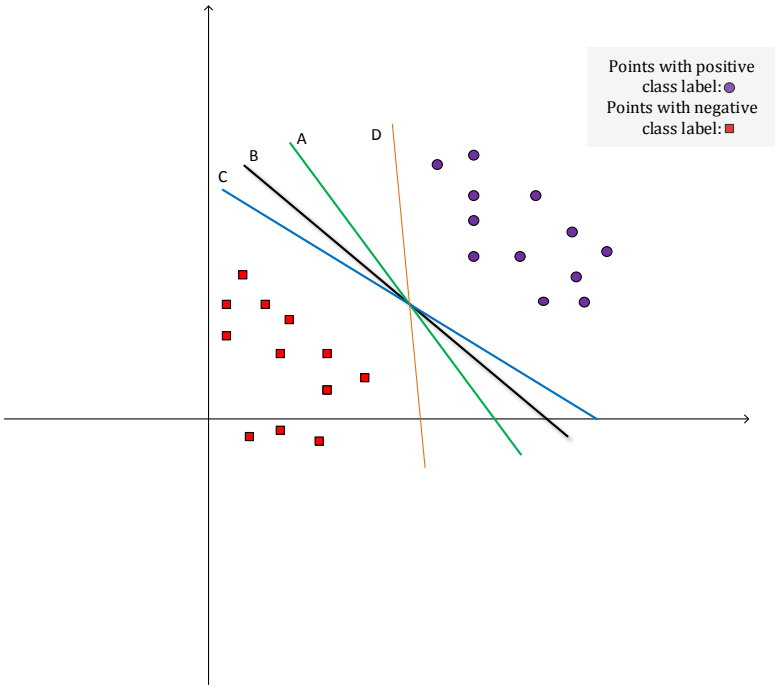

Figure 1. Perceptron Separating Possibilities.

The perceptron's task is to decide on which side of the decision boundary a point belongs to. Mathematically, this is equivalent to evaluating a function equal to the sum of the weighed inputs and checking whether the sum is positive or negative. Geometrically, "weighing" is equivalent to the slope of the separating line. In order to choose the correct separator, the

perceptron applies a "learning algorithm" which adjusts the line so that all the points lie on the correct side of the line.

However, the perceptron has two limitations. Firstly, there exist several possibilities to separate the data and the perceptron does not have a criteria on choosing the best one (Bishop 2006). For example, in Fig. 1, all of the separating hyper planes A, B, C, or D can be used to separate the two classes. Secondly, it can only solve problems which are linearly separable, so it never converges in this case.

The best separator is found to be the one with the maximum margin. This means that the distance to the closest points is maximum. The closest data points are known as the support vectors, and only these points are relevant in order to classify a new point (Ben-Hur, Weston 2010).

Statistical learning theory is the framework which helps define the maximum margin separator as the optimal separator. Statistical learning theory or VC theory refers to the ability of learning machines to generalize well to unseen data. The idea from statistical learning theory that led to SVMs is the structural risk minimization principle, which minimizes an upper bound of the generalization error rather than the training error (Tay, Cao 2001). The main concept is that of a VC dimension (Vapnik-Chervonenkis), a measure of how many classifications can be implemented from that hypothesis space. The generalization error is bounded by the sum of the training error and a confidence interval term that depends on the VC dimension (Burbidge, Buxton 2001).

2.2 Linear Support Vector Machine

In the binary linear classification problem described in section 2.1, a straight line can be used to separate data in a two dimensional space. However, in higher dimensional spaces data is separated by using hyper planes. There might exist several hyper planes to achieve this, but the SVM's goal is to find a separating hyper plane that maximizes the margin. The margin is defined as the equidistance from the separating hyper plane to the closest data points; these points are known as the "support vectors".

According to Vapnik's original formulation (Vapnik, Cortes 1995), the training data set is of the form $\{x_i, y_i\}$, with i=1,...,n, $y_i \in \{-1, 1\}$ and x_i represents the input vectors (the features) and y_i represents the class labels -1 or $+1$.

A linear classifier can be described with a linear function of the form $f(x) = w^T x + b$, where $w^T x$ denotes the dot product defined as $\sum_i w_i x_i$, w is the weight vector, b is called "bias" and

represents the distance from the origin to the separating hyper plane. The sign of $f(x)$ determines on which side of the hyper plane a new point will be classified into. In this context, we define the decision boundary of a classifier as the boundary lying between the regions classified as belonging to one class and the other (Ng 2007).

2.2.1 Optimal Separating Hyper Plane

A hyper plane can be represented by a pair (w, b), where w is the normal to the hyper plane and b is a constant. The equation describing the hyper plane is $w \cdot x + b = 0$. These can be depicted in Fig. 2.

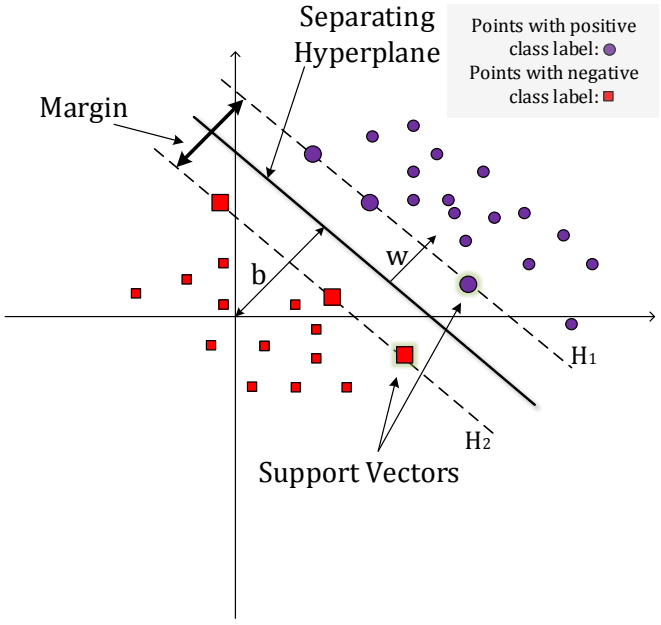

Figure 2. Optimal separating hyper plane.

To implement the SVM, we have to find the values of w and b which satisfy:

$$\begin{cases} x_i \times w + b \geq 1, & if\ y_i = 1\,, \\ x_i \times w + b \leq 1, & if\ y_i = -1\,. \end{cases}$$

These two relations can be combined into a single equation $y_i(x_i \times w + b) - 1 \geq 0$.

The support vectors are defined as the points for which the two inequalities from above hold. They lie on the hyper planes H_1: $x_i \times w + b = 1$ and H_2: $x_i \times w + b = -1$, shown with dotted lines in Fig. 2.

The margin is defined as $\frac{2}{\|w\|}$, where $\|w\|$ is the norm of vector w. In order to find the optimal separating hyper plane, the margin needs to be maximized. Through a few mathematical transformations, it can be shown that maximizing the margin is in fact equivalent to minimizing

$\min \frac{1}{2} \|w\|^2$ with respect to b and w so that the condition $y_i(x_i \times w + b) - 1 \geq 0$ is satisfied (Burges 1998). This is a constrained quadratic optimization problem (QP) also known as the primal problem. The optimal margin separator results as the solution to this optimization problem (Ben-Hur, Weston 2010).

2.2.2 Solving the Optimization Problem: Duality

In order to solve the QP problem from the previous section, we formulate it as a Lagrangian optimization problem. The idea is to use dot products to represent test and training data. These play a key role in the kernel trick described in section 2.4. We introduce Lagrange multipliers α_i, and the representation in terms of α_i is known as the dual form (Ng 2007). The QP problem therefore becomes

$$L_p = \frac{1}{2} \cdot \|w\|^2 - \alpha[y_i(x_i \times w + b) - 1] = \frac{1}{2} \cdot \|w\|^2 - \sum_{i=1}^{L} \alpha_i [y_i(x_i \cdot w + b) - 1].$$

The goal is to find α_i which maximizes, and b and w which minimize the expression of L_p. We calculate the partial derivatives of L_p with respect to w and b and set them to zero

$$\frac{\partial L_p}{\partial w} = 0 \Rightarrow w = \sum_{i=1}^{L} \alpha_i x_i y_i \,,$$

$$\frac{\partial L_p}{\partial b} = 0 \Rightarrow \sum_{i=1}^{L} \alpha_i y_i = 0 \,.$$

By substituting these relations in the formulation of L_p from above, we get the dual formulation of the primal problem. The advantage of the dual form is that it is a relation written only in terms of inner products of the input vectors x_i

$$L_D = \sum_{i=1}^{L} \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j x_i x_j$$

$$\text{s. t. } \alpha_i \geq 0 \; \forall i, \sum_{i=1}^{L} \alpha_i y_i = 0.$$

In order to solve the dual problem, we need to maximize the expression of L_D and find the values of α_i. This is done by solving the QP problem, which will return the value of w. The value of b is found using the support vectors (Bennett 2003). Solving the dual maximization problem is equivalent to training the SVM. There are several approaches available to training the SVM. One of the most popular approaches, which involves choosing 2 Lagrange multipliers at a time is called the Sequential Minimization Algorithm (SMO). The advantage of SMO lies in the fact that it

solves the problem analytically and therefore avoids numerical quadratic programming, so it provides an efficient way to deal with the dual problem (Bishop 2006).

Once we have trained the SVM, the classifier will assign a label on a new input test point x_i, depending on which side of the decision boundary the point will lie on. Hence, a new point will be classified according to the decision function $f(x_{new}) = w^T x_{new} + b$. This corresponds to the testing phase of the SVM (Burges 1998).

Both the quadratic problem and the dual formulation depend only on inner products between input vectors. This allows for the use of the "kernel trick" and for extending the algorithm for the non-linear case, which will be introduced in section 2.4.

2.3 Soft Margin SVM

For the case when data is not fully separable, we have to allow for misclassified points by introducing slack variables $\sigma_i, i = 1, \dots L$ in the constraints. Therefore, some points will lie on the wrong side of the hyper plane. This can be seen in Fig. 3, where one of the points belonging to the positive class label lies with the points belonging to the negative class label. In this so called "soft margin" SVM, data points which lie on the incorrect side of the boundary get a penalty, or "cost" which increases with the distance from it (Fletcher 2008).

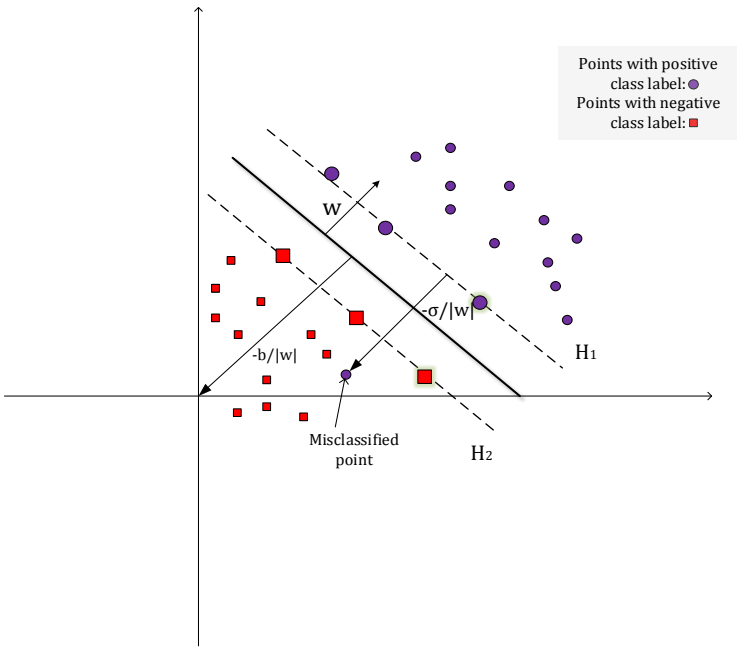

Figure 3. Soft Margin Support Vector Machine.

The constraints become

$x_i \cdot w + b \geq 1 - \sigma_i$, if $y_i = 1$,

$x_i \cdot w + b \leq -1 + \sigma_i$, if $y_i = -1$, where $\sigma_i \geq 0, \forall i$.

The objective function in the primal described in the previous section is then re-formulated as

$$\min \frac{1}{2} \|w\|^2 + C \sum_{i=1}^{L} \sigma_i$$

$$\text{w.r.t. } y_i(x_i \cdot w + b) - 1 + \sigma_i \geq 0,$$

where the parameter C represents the tradeoff between the penalty and the size of the margin, or more exactly the tradeoff between maximizing the margin and minimizing the training error (Bishop 2006). Similarly to the linearly separable case, the problem is formulated as a Lagrangian, with the only difference that a new constraint is introduced $0 \leq \alpha_i \leq C$ (Fletcher 2008). A complete derivation for this case is given in (Burges 1998).

2.4 Non-linearly Separable Data: Kernel Trick

To tackle the second limitation of the perceptron mentioned in section 2.1, mainly the non-linearity of the input data, the SVM uses a non-linear mapping to projects the data into a higher dimensional feature space where the data is linearly separable. This is achieved by applying a function called a "kernel" which computes a dot product of the input data.

For classification and regression problems where the data is not linearly separable, the SVM is adapted to the nonlinear case by a mapping $x \to \varphi(x)$ from the input space X into the into a higher dimensional feature space F where it is possible to separate the data. This is done using a non-linear function $\varphi(x): X \to F$, and in this case the SVM can be represented by the decision function $f(x) = w^T \varphi(x) + b$, where $\varphi(x)$ is known as the feature map (Ng 2007).

The weight vector can be expressed as a sum of dot products of input vectors, $w = \sum_{i=1}^{n} \alpha_i y_i \varphi(x_i)$, which allows the decision function to be formulated as $f(x) = \sum_{i=1}^{n} \alpha_i y_i \varphi(x_i)\, \varphi(x) + b$.

Transforming the data from the lower dimensional feature space into the higher dimensional space is achieved by means of a kernel function. The kernel function is defined as an inner product between the feature maps $\langle \varphi(x_i), \varphi(x_j) \rangle$. This dot product is replaced by a kernel function. By replacing the kernel function into the dual problem, we can find linear separators in high dimensional feature spaces (Dharnidharka 2012). This transformation is known as the "kernel trick".

The mapping is introduced into the objective function and the optimization problem becomes

$$L_D = \sum_{i=1}^{L} \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j \varphi(x_i) \varphi(x_j)$$

$$\text{such that } 0 \leq \alpha_i \leq C \ \forall i,$$

$$\sum_{i=1}^{L} \alpha_i y_i = 0.$$

Substituting the kernel function into the dual problem gives the nonlinear classifier

$$L_D = \sum_{i=1}^{L} \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j K(x_i, x_j)$$

$$\text{such that } 0 \leq \alpha_i \leq C \ \forall i,$$

$\sum_{i=1}^{L} \alpha_i y_i = 0$ (Bennett 2003).

Thus, in terms of the kernel function, the decision function can be written in the form

$$f(x) = \sum_{i=1}^{n} \alpha_i y_i K(x_i, x_j) + b \, .$$

Hence, operations are performed in the input space and the kernel in the input space is just the equivalent of the inner product in the feature space (Gunn 1998). The "kernel trick" means that training sets which are not linearly separable in an input space can be transformed so that they become linearly separable in the feature space. In this higher dimensional feature space we can find linear separators, but in the original space the separators are non-linear. An example can be seen in the figure below, where the function $\phi(x_1, x_2) = (x_1^2, \sqrt{2}\, x_1 x_2, x_2^2)$ is used to map the original input space $\mathbb{R}^2$ into a higher dimensional feature space $\mathbb{R}^3$ (Üstün et al. 2006).

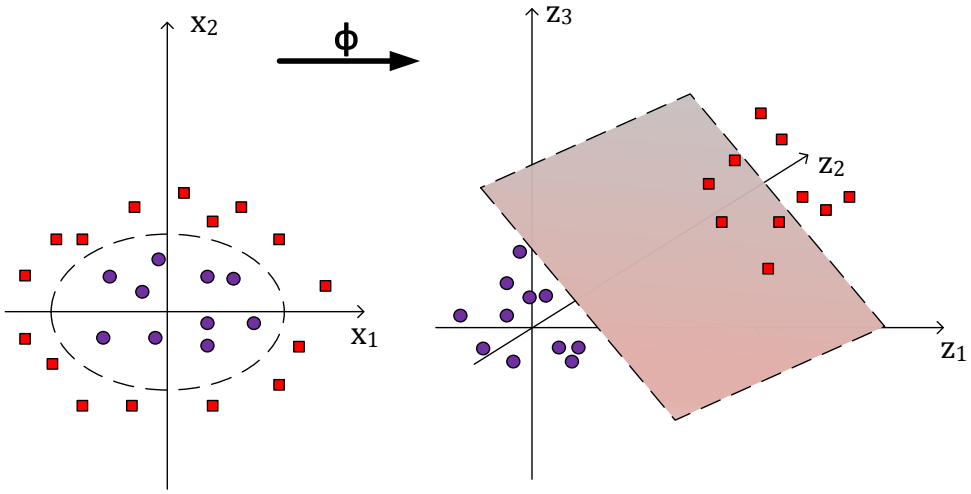

Figure 4. Illustration of the "kernel trick". Figure adapted from (Üstün et al. 2006).

As a criteria to decide which function $K(x_i, x_j)$ corresponds to a dot product in a feature space, we take into account Mercer's Theorem. This gives the necessary and sufficient condition for a valid kernel. It states that every symmetric kernel function with a positive definite corresponding kernel matrix is the equivalent of a dot product in some feature space (Russell, Norvig 2010).

The advantage of kernel functions is that one can evaluate the inner products without explicitly calculating the mapping; which otherwise can prove to be an expensive computation. If the value of the function $K(x_i, x_j)$ is known, then the mapping function does not require calculating explicitly. Instead, one can use the value of the kernel. Mathematically, the kernel function is based upon reproducing Hilbert Spaces, which are just a generalization of Euclidean spaces (Burges 1998).

The most common kernel functions for regression and classification are:

- Linear kernel $K(x_i, x_j) = \langle x_i^T x_j \rangle$,
- Polynomial kernel defined by $K(x_i, x_j) = \left(\langle x_i^T x_j \rangle + a\right)^d$,
- Gaussian kernel $K(x_i, x_j) = e^{-\left(\frac{\|x_i - x_j\|^2}{2\gamma^2}\right)}$,

where a represents a constant, d the dimension; and γ represents a weighing factor (Shalizi 2009), more exactly the width of the Gaussian (Fletcher 2008).

The SVM described above is known as C-SVM formulation. In addition to this formulation, there is an alternative one known as the v-SVM formulation. In this case, parameter C is replaced by v which represents the upper bound on the fraction of margin errors and the lower bound on the fraction of support vectors (Schölkopf et al. 2000). The advantage of this formulation is that it gives more control over the number of support vectors (Meyer 2012).

2.5 Model Selection

The parameters of the kernel function, the cost function from the soft margin SVM and for the regression described in section 3 are called the hyper-parameters of the SVM. The choice of these parameters have a significant effect on the decision boundary, and for classification tasks it strongly affects the accuracy of the classifier (Ben-Hur, Weston 2010). The most common approach to find the optimal parameters is to perform a grid search and then train the SVM with the parameters returned by the grid search. Also, finding the best kernel strongly depends on the data and it is mostly chosen by trial and error (Burges 1998).

3. Support Vector Regression

Section 2 presented the SVM for classification problems. However, for regression problems, the aim is to predict real valued ouput numerical values. Support vector regression (SVR) makes use of an ϵ-sensitive loss function or error function, which satisfies $-\epsilon \leq x_i \times w - b - y_i \leq \epsilon$. This function can be described by (Bishop 2006)

$$E(y_i) = \begin{cases} 0 & , \text{ if } |x_i \times w + b - y_i| < \epsilon, \\ |x_i \times w + b - y_i| - \epsilon, & otherwise. \end{cases}$$

Thus, the function does not allocate an error if the absolute value of the difference between the predicted value y_i and the actual value is smaller than the value of ϵ (Smola, Vishwanathan 2010). As opposed to classification, in regression the value of ϵ is given, and the objective is to find a hyper plane which has all the training points lying inside a tube bounded by $y_i \pm \epsilon$, called "ϵ-sensitive tube". This can be seen in Figure 5.

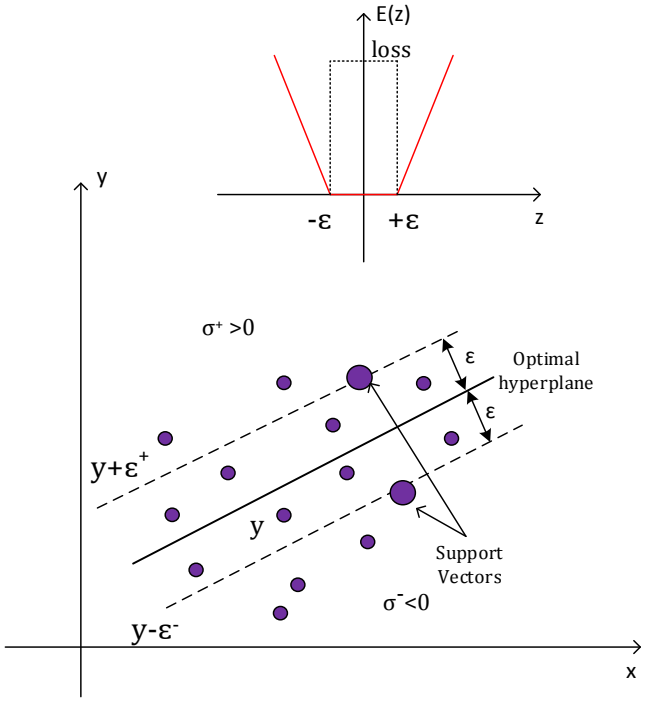

Figure 5. Support Vector Regression: loss function (above) and "ϵ-sensitive tube" (below). Figure adapted from (Üstün et al. 2006).

Analogously to the soft margin SVM defined above, the points which lie outside the tube are assigned a slack penalty, which can be either positive σ_i^+ or negative σ_i^-, depending whether they lie above or below the tube. The slack variables are zero for the points inside the tube (Fletcher 2008).

For SVR, the primal problem is formulated as follows

$$min\,\frac{1}{2}\|w\|^2 + C\sum_{i=1}^{L}(\sigma_i^+ + \sigma_i^-) \text{ s.t.}$$

$$\sigma_i^+ \geq 0, \sigma_i^- \geq 0\,,$$

$$(x_i \times w - b - y_i) + \sigma_i^+ \geq \epsilon\,,$$

$$(x_i \times w - b - y_i) - \sigma_i^- \leq -\epsilon.$$

where C represents a regularization constant. Both C and ϵ are parameters which are chosen beforehand by the user and are determined empirically (Cao, Tay 2012).

The primal can be solved as before, by introducing Lagrange multipliers α_i^+, α_i^- and by setting up the Lagrangian dual formulation. By optimizing the Lagrangian and setting the partial derivatives with respect to the primal variables to zero, we can reformulate the dual problem as maximizing L_D with respect to α_i^+, α_i^-, or, more exactly

$$L_D = \sum_{i=1}^{L}\big((\alpha_i^+ - \alpha_i^-)y_i\big) - \epsilon\sum_{i=1}^{L}(\alpha_i^+ - \alpha_i^-) - \frac{1}{2}\sum_{i,j}(\alpha_i^+ - \alpha_i^-)\,(\alpha_i^+ - \alpha_i^-)x_i x_j.$$

A complete mathematical derivation of the SVR can be found in (Fletcher 2008).

Finally, each new input point is predicted by evaluating the decision function $f(x) = \sum_{i=1}^{L}(\alpha_i^+ - \alpha_i^-)\langle x_i x\rangle + b$. The support vectors are those points lying on the boundary of the ϵ-sensitive tube or outside the tube, or mathematically, the points for which both $\alpha_i^+, \alpha_i^- \neq 0$. The decision boundary will depend once again only on the support vectors (Bishop 2006). The support vectors are usually a small subset of the training data and the property that the solution of the SVM will dependent only on these data points is known as the sparsity of the solution. Also, minimizing the Lagrange multipliers reduces the number of support vectors (Müller et al. 1997).

The support vector regression described above is known as ϵ-SVR (Vapnik, Cortes 1995). SVR can implement other loss functions such as Laplace loss, Huber loss or square loss (Smola, Vishwanathan 2010). Similar to the classification case, SVR can be extended for non-linear regression tasks by the use of kernels, as described in Section 2.4.

Other regression formulations include v-SVR, which unlike ϵ-SVR where the errors are not relevant as long as they are less than ϵ (Smola, Scholkopf 2004), uses a parameter v to control for the number of points lying outside the ϵ-sensitive tube (Bishop 2006). The v-SVR was proposed by (Schölkopf et al. 2000), its main advantage compared to the latter being that it allows the tube to adapt automatically to the data by setting a parameter $0 \leq v \leq 1$ that restricts

the points which lie outside the tube, this way controlling for the number of errors. A full derivation of the algorithm is given in (Schölkopf et al. 2000). Finally, v just represents the fraction of bad outliers and v-SVR differs from ϵ-SVR just in the choice of parameters C and v, which replace ϵ and C. v-SVR solves the problems that come up with introducing C, which might be difficult to tune. v controls the number of support vectors and margin errors.

The regression estimation is done by risk minimization, and Vapnik's ϵ-sensitive loss function is used as a measure of this risk. SVMs use a risk function which consists of two terms: the empirical error and the regularization term, both derived from the structural risk minimization principle (Tay, Cao 2001).

4. Oil Price Prediction in R

The previous sections have described the theoretical concepts behind SVMs. In the following, an example of a SVM implemented in the statistical software R is presented. More exactly, a SVR is estimated on a time series of oil price data in order to observe how different economic variables affect the price oil.

4.1 Implementation in R

In R, there are several SVM implementations available in packages like e1071, kernlab or klaR. In this paper, SVM is implemented through the function *svm()* in package e1071. This particular package offers an interface to the C++ implementation *libsvm* by Chih-Chung Chang and Chih-Jen Lin (Meyer 2012). In *libsvm*, the most popular formulations of SVM are available: C-SVM, v-SVM, v-SVR, ϵ-SVR and also novelty detection. The function *svm()* is used to train the SVM, and it can be set to perform classification, regression or novelty detection. Depending on whether the dependent variable y is a factor or a numeric vector, SVM can be used for classification or regression tasks. In addition, the function has specific configurations, allowing the user to choose different types of kernels, their parameters, and other features such as: probability values for prediction, class weighing in the case of classification, cross-validation to compute training error or shrinking heuristics. The function returns an object of class *svm* which includes the number of support vectors, the type of the SVM formulation, and the parameters of the SVM.

The package also provides a *predict()* method which predicts values based on the trained model. To tune the hyper-parameters of the SVM, the function *tune()* performs a grid search over specified parameters ranges. For classification tasks, the *plot()* function allows visualizing the data by showing the classes, the support vectors and the decision boundaries (Meyer 2012).

4.2 Data and Model Description

The practical part of this paper requires running a SVR to predict the WTI crude oil price. The dataset includes time series with a total of 2352 daily observations from 2003 until 2012 and it was retrieved from http://www.quandl.com/. The parameters used for the regression are presented in Table 1.

Parameter	Description	Frequency
WTI	West Texas Intermediate, Crude Oil Spot Price, US$/ Barrel	Daily
DJI	Dow Jones Industrial Average	Daily
GDP	GDP of USA	Monthly
AllNetOptimismHEOfWords	Oil News Sentiment Values, retrieved from dataset "Aggregate_Full"	Daily
wtilag	Lagged oil price value from previous day	Daily

Table 1. Models used for SVR.

The dataset is divided into a train and a test set, 1/3 for training the model (702 observations) and 2/3 for testing (1649 observations) the model.

Six regression models are trained, tested and tuned for best values of C and γ. Models are then trained using different kernel functions, and the effect is observed using different metrics discussed in the following. WTI oil price is predicted for a 10 day horizon on the test set. The objective is to see how the other variables influence the oil price. These models are explained in Table 2.

Model	Dependent Variable	Independent Variables
1	WTI	GDP, wtilag
2	WTI	GDP, DJI, wtilag
3	WTI	GDP, DJI, "Oil News", wtilag
4	WTI	DJI, wtilag
5	WTI	DJI, "Oil News", wtilag
6	WTI	"OilNews", wtilag

Table 2. Models used for SVR.

4.3 Prediction Performance

We want to measure the predictive power of the models and we estimate it by computing the accuracy of the prediction on the test set. The prediction performance is evaluated by using three of the most popular evaluation metrics in forecasting: mean absolute error (MAE), root mean squared error (RMSE), mean squared error (MSE). MAE is defined as the absolute value of the difference between the actual value a_i and the predicted value p_i. RMSE is just the squared root between the actual value and the predicted value, and MSE is the average of the squared errors. The smaller the MAE and the RMSE, the better the accuracy of the prediction (Bruno 2008). In addition, the Mean Absolute Percentage Error (MAPE) measures the percentage difference of the errors. The advantage of the MAPE is that it is a relative error and it provides a measure of errors as a percentage of the actual data (Makridakis, Hibon 1995). The calculation of these errors is presented in Table 3.

Evaluation metrics	Calculation
MAE	$MAE = \frac{\sum_{i=1}^{n}\lvert a_i - p_i\rvert}{n}$
RMSE	$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(a_i - p_i)^2}{n}}$
MSE	$MSE = \frac{\sum_{i=1}^{n}(a_i - p_i)^2}{n}$
MAPE	$MAPE = \frac{100\%}{n} \cdot \sum_{i=1}^{n}\left\lvert\frac{a_i - p_i}{a_i}\right\rvert$

Table 3. Evaluations metrics and their calculation.

These errors are implemented through the package "Metrics" in R, which provides evaluation metrics for machine learning. Their values for each model are summarized in Table 4. In addition, the effect of different kernel types and tuned values of model hyper parameters are observed on the model performance.

Model	Kernel and Parameters	Evaluation Metrics			
		RMSE	MSE	MAE	MAPE
Model 1	Radial	1.199	1.437	0.951	0.0105
	C=16, γ =0.5	1.437	2.066	1.106	0.012
	Polynomial	1.456	2.122	1.115	0.012
	Sigmoid	1801.61	3245803	1798.52	19.76
	Linear	1.261	1.592	1.046	0.011

Model 2	Radial	2.363	5.584	1.942	0.022
	C=100, γ=0.001	1.391	1.936	1.070	0.011
	Polynomial	1.815	3.296	1.520	0.016
	Sigmoid	1333.45	1778109	1309.54	13.985
	Linear	1.394	1.945	1.080	0.011
Model 3	Radial	2.422	5.868	1.334	0.014
	C=16, γ =0.1	1.635	2.674	0.997	0.010
	Polynomial	1.687	2.846	1.172	0.012
	Sigmoid	392.76	154267.5	314.66	3.38
	Linear	0.837	0.700	0.773	0.008
Model 4	Radial	0.882	0.778	0.770	0.882
	C=16, γ =0.5	0.899	0.809	0.767	0.008
	Polynomial	1.01	1.02	0.866	0.009
	Sigmoid	1211.55	1467859	1152.503	12.066
	Linear	0.752	0.566	0.653	0.006
Model 5	Radial	3.628	13.167	2.151	0.023
	C=8, γ =0.5	2.909	8.464	2.102	0.022
	Polynomial	1.382	1.910	1.154	0.0126
	Sigmoid	253.72	64376.66	179.67	1.966
	Linear	1.194	1.425	1.046	0.011
Model 6	Radial	1.556	2.423	1.217	0.013
	C=100, γ=0.001	1.273	1.622	1.081	0.119
	Polynomial	1.262	1.594	1.079	0.011
	Sigmoid	266.73	71148.54	259.83	2.848
	Linear	1.283	1.648	1.086	0.011

Table 4. Values for the metrics used to assess the models.

Model 4 displays the lowest values of the RMSE and MSE, so we can conclude that it is the best in terms of predictive power. Furthermore, the effect of different kernels on the models is observed. When trained with the radial kernel and linear kernel, the RMSE and MSE of all models improve significantly. The accuracy also improves for some of the models when training again with the radial kernel and the tuned parameters resulted from the grid search.

In addition to these criteria, which measure how far the actual value is from the predicted value, a further evaluation criteria is used. A Diebold-Mariano Test (DM Test) compares the forecasting

accuracy between models, in terms of the significance of differences between them (Bruno 2008). This is implemented through the library *forecast()* in R. We compare the models, testing the null hypothesis that one model has greater prediction power than the others. Table 5 displays the p-values of the test performed between all two combinations of models.

DM Test: p-value	**Model 1**	**Model 2**	**Model 3**	**Model 4**	**Model 5**	**Model 6**
Model 1	-	0.999	0.895	0.991	0.952	0.850
Model 2	0.0003	-	0.050	0.065	0.028	0.016
Model 3	0.104	0.949	-	0.833	0.601	0.342
Model 4	0.008	0.935	0.166	-	0.112	0.049
Model 5	0.047	0.971	0.398	0.887	-	0.222
Model 6	0.149	0.983	0.657	0.950	0.777	-

Table 5. Diebold-Mariano Test and returned p-values.

From Table 5, we can see that for Model 1, $p>0.1$, so we fail to reject the null hypothesis. Hence, model 1 has greater predictive performance than all other models at the 10% significance level. For Model 2, we reject the null hypothesis that model 2 has greater performance than model 1, 3, 5 and 6 at the 5% level, but we fail to reject the hypothesis that model 2 has greater predictive performance than model 4, thus model 2 is significantly better at the 5% significance level. Model 3 has greater performance than model 1 at the 10% significance level, and greater than all other models at the 5% significance level. Model 4 has greater performance than model 2 and model 6 at the 5% level, and greater than model 3 and 5 at the 10% significance level; however it's not better than model 1 at the 1% significance level. Model 5 is significantly better than model 2, 3, 4 and 6 at the 10% level, but not significantly better than model 1 at the 5% level. Model 6 is significantly better than all other models at the 10% significance level.

Relationships between variables can be visualized with a plot called a "heat map". Oil price values, DJI values and sentiment values on oil prices are put together in a matrix which is then plotted. The heat map is a way to visualize the data in the matrix. Colors are used to display numerical values. High values are displayed with hot orange and red tones, and lower values with yellow tones. Similar rows and columns are grouped together by hierarchical cluster analysis. There are a lot of patterns of about the same color in the heat map, this suggests that rows are correlated with columns.

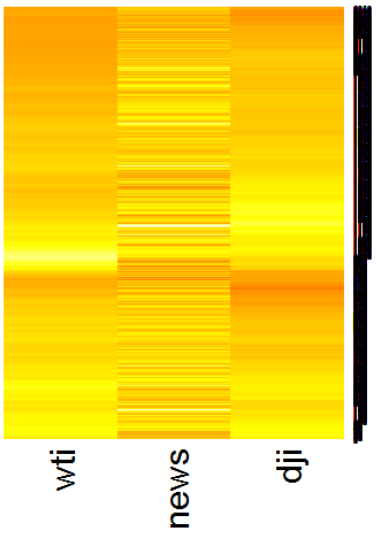

Figure 6. Heat map displaying Oil News, DJI and WTI Oil Price.

5. Conclusion

In this paper, the theoretical background of Support Vector Machines were explained for the tasks of classification and regression. SVMs have several advantages compared to other machine learning techniques. However, tuning the SVM remains rather an "art", with the choice of hyper parameters visibly affecting the performance of the SVM. A support vector regression was applied for the task of predicting the WTI crude oil price. Different models were evaluated in terms of their prediction performance and the effect of different kernel functions and model hyper parameters was observed on the accuracy of the prediction. Finally, a recommendation was made in favor of Model 4.

6. References

Ben-Hur, Asa; Weston, Jason (2010): A User's guide to SVM. In *Methods Mol. Biol.* Volume: 609, pp. 223–239, checked on 12/30/2013.

Bennett, Kristin (2003): Support Vector Machines: Hype or Hallelujah? In *SIGKDD Explorations* 2 (December 2000), p. 2000, checked on 12/31/2013.

Bishop, Christopher M. (2006): Pattern Recognition and Machine Learning. New York: Springer (Information science and statistics), checked on 12/30/2013.

Boswell, Dustin (2002): Introduction to Support Vector Machines. File number 15. Scientific Paper.

Bruno, Giancarlo (2008): Forecasting Using Functional Coefficients Autoregressive Models. In *ISAE - Institute for Studies and Economic Analyses* MPRA Paper. Available online at http://mpra.ub.uni-muenchen.de/42335/, checked on 1/19/2014.

Burbidge, Robert; Buxton, Bernard (2001): An Introduction to Support Vector Machines for Data Mining. In *KEYNOTE PAPERS, YOUNG OR12, UNIVERSITY OF NOTTINGHAM, OPERATIONAL RESEARCH SOCIETY*, pp. 3–15.

Burges, Christopher (1998): A Tutorial on Support Vector Machines for Pattern Recognition. Data Mining and Knowledge Discovery 2, 121-167, 1998: Kluwer Academic Publishers Hingham, MA, USA.

Cao, Lijuan; Tay, Francis (2012): ε-Descending Support Vector Machines for Financial Time Series Forecasting. In *Neural Processing Letters* 15 Issue 2 (2), checked on 1/3/2014.

Dharnidharka, Piyush (2012): Forecasting Foreign Currency Exchange Data using SVM Based Models, checked on 1/7/2014.

Fletcher, Tristan (2008): Support Vector Machines Explained.

Gunn, Steve R. (1998): Support Vector Machines for Classification and Regression.

Huson, D. (2007): SVMs and kernel functions. University of Tubingen (Algorithms in Bioinformatics). Available online at http://ab.inf.uni-tuebingen.de/teaching/ss07/albi2/script/svm.pdf, checked on 1/29/2014.

Makridakis, S.; Hibon, M. (1995): Evaluating accuracy measures: Fontainebleau (R&D / INSEAD ; 95,18).

Meyer, David (2012): Support Vector Machines.

Müller, K.-R.; Smola A.J.; Rätsch, G.; Schölkopf B. (1997): Predicting Time Series with Support Vector Machines. Advances in Kernel Methods - Support Vector Learning: Cambridge, MA, MIT Press, checked on 1/4/2014.

Ng, Andrew (2007): CS229 Machine Learning. Stanford University. Stanford, CA (Stanford Machine Learning lecture notes). Available online at http://cs229.stanford.edu/notes/cs229-notes3.pdf, checked on 12/30/2013.

Russell, Stuart; Norvig, Peter (2010): Artificial Intelligence. A modern approach. 3rd. 1 volume. Upper Saddle River, NJ, USA ©2009: Prentice Hall Press.

Schölkopf, Bernhard; Smola, Alex J.; Williamson, Robert C. (2000): New Support Vector Algorithms. Cambridge, MA, USA: MIT Press (12). Available online at http://users.cecs.anu.edu.au/~williams/papers/P115.pdf, checked on 1/3/2014.

Shalizi, Cosma (2009): Support Vector Machines. Statistics 36-350, Data Mining, Fall 2009. 36-350: Data Mining. Carnegie Mellon University, Statistics Dept., Pittsburgh, PA 15213-3890 USA, 2009. Available online at http://www.stat.cmu.edu/~cshalizi/350/.

Siebers, M.: Lecture 10: Support Vector Machines and their Applications. Part II: Special Aspects of Concept Learning. Cognitive Systems - Machine Learning. Universitat Bamberg. Available online at http://www.cogsys.wiai.uni-bamberg.de/teaching/ws1314/ml/slides/cogsysII-10.pdf, checked on 1/4/2014.

Smola, Alex; Vishwanathan, S.V.N. (2010): An Introduction to Machine Learning. The Pitt Building, Trumpington Street, Cambridge, United Kingdom: Cambridge University Press. Available online at http://alex.smola.org/drafts/thebook.pdf, checked on 1/2/2014.

Smola, Alex J.; Scholkopf, Bernhard (2004): A Tutorial on Support Vector Regression. In *Statistics and Computing* 14 (3).

Tay, Francis E.H.; Cao, Lijuan (2001): Application of support vector machines in financial time series forecasting 29 (4), pp. 309–317.

Üstün, B.; Melssen, W. J.; Buydens, L.M.C. (2006): Facilitating the application of Support Vector Regression by using a universal Pearson VII function based kernel. In *Chemometrics and Intelligent Laboratory Systems* 81 (1), pp. 29–40. DOI: 10.1016/j.chemolab.2005.09.003.

Vapnik, Vladimir; Cortes, Corinna (1995): Support Vector Networks. In *Machine Learning Journal* 20 (3), pp. 273–297, checked on 1/2/2014.