ENSEMBLE SVR FOR PREDICTION OF TIME SERIES

YU-FENG DENG, XING JIN, YI-XIN ZHONG

Beijing University of posts and telecommunications, beijing, 100876, P.R.China
E-MAIL: jx9802@126.com

Abstract:

Recently, Support Vector Machine(SVM) as a new kernel learning algorithm has successfully been used in nonlinear time series prediction. To improve the prediction performance of SVM, We concentrate on ensemble method. Bagging and boosting, two famous ensemble methods, will be examined in this paper. Experiments on two data sets (sunspots and Mackey-Glass) have shown that bagging SVR and boosting SVR could all improve the performance when compared with single SVR. And for boosting, weighted median is a better choice for combining the regressors than the weighted mean.

Keywords:

Time series prediction; Ensemble method; bagging; Adaboost; SVR

I. Introduction

The reliable prediction of future values for discrete-time or continue-time series has many important applications ranging from finance time series analysis to signal processing, from dynamic system control to natural phenomena prediction. To obtain a good predictor (could be a classification function, regression function or density estimation function) after some observations (called training samples) have been given, we should make extensive use of the past evolution of series. A prediction is commonly expressed as a function of the recent history of the time series, $\widetilde{x_{t+s}} = f(x_{t-1}, x_{t-2}, \ldots)$, $s \geq 0, s \in Z$, we'll consider only one-step prediction, i.e. $s = 0$. Building a model then amounts to finding an appropriate function f . Traditional time series models are linear, i.e. AR(autoregressive), MA (moving average) and ARMA(autoregressive moving average)[16]. However, recent work by several investigators [17][18] has suggested that nonlinearities can improve forecasting models in some cases. This causes the research of nonlinear prediction in modern time series analysis. Illuminated by the success of BP neural network and RBF network applied in time series prediction, SVM as a new kernel learning algorithm has also been successfully used in nonlinear prediction of time series.

The philosophical principle of SVM is that it is based on the Vapnik's SRM (Structural Risk Minimization) principle which seeks to minimize an upper bound of the generalization error consisting of the sum of the training error(on training sample) and a capacity (measure for the richness of hypothesis space)term. This induction principle is different from classical empirical risk minimization (ERM) principle which only minimizes the training error. Established on the unique principle, SVM usually achieves higher generalization performance than traditional neural networks that implement the ERM principle in solving many machine learning problems. Another key characteristic of SVM is that training SVM is equivalent to solving a linearly constrained quadratic programming problem so that the solution of SVM is always unique and globally optimal, unlike other neural networks' training which requires nonlinear optimization with the danger of getting stuck into local minima. In SVM, the solution to the problem is only dependent on a subset of training data points which are referred to as support vectors. Using only support vectors, the same solution can be obtained as using all the training data points. One disadvantage of SVM is that the training time scales somewhere between quadratic and cubic with respect to the number of training samples. So a large amount of computation time will be involved when SVM is applied for solving large-size problems.

To improve upon the performance obtained by SVM, we can attempt to develop a more appropriate learning algorithm by making use of prior information regarding the application under study. We can also adapt general procedures that were found to enhance the accuracy of various basic learners. Ensemble method could be thought of as important method for improving performance by composing many "weak" learners, which have prediction performance just a little better than random guessing. Bagging and boosting, two famous different ensemble mind, have been successfully demonstrated by several authors [3][4][7][10] in the applications of machine learning and pattern analysis. The common characteristic for them is to manipulate the training set. However for bagging, the training sample at each round is achieved by just uniform sampling the original sample; for boosting, the training set have different distribution according to their prediction results from previous weak learners, and after a round of training the training examples with bad prediction will be strengthened in next learner. Bagging works especially well for unstable learning algorithms——algorithms whose output regression undergoes major change in response to small changes in the training data. However it can slightly degrade the performance of stable procedures. To our surprise, the performance of boosting on the test set could be improved and then level off as more learners are included. It seems to contradict Occam's principle, an important principle in machine learning, which tells us that the simpler learner is the better. Some authors have used margin theory (from Vapnik's statistical learning theory) to give quantity explanation for this phenomenon and others [8][19]argue that boosting is a gradient descent optimization method for special object function.

Recently, several authors [2][20][21]have used ensemble method for improving the performance of time series prediction. It's necessary to refer the work in [2] which is similar to our method in boosting method. However, in [2] the base learner is RNN (Recurrent Neural Network). We'll get the same result as [2] that for the combination of boosting regressors, weighted median is a better choice than weighted mean.

The paper is organized as following: the theory of Support Vector Regression algorithm will be presented in section II;Our ensemble methods for SVR are described in section III; section IV shows our experimental results; which is followed by conclusions and future work..

2. Support Vector Regression

In our learning task of regression, the learning procedure could be described as follows: Suppose l training samples $(x_i, y_i) \in X \times Y, X \subseteq R^n$, $Y \subseteq R, (i = 1,...,l)$, which are generated independently according to an unknown joint distribution $P(x, y)$ by a system, have been given. The goal of our learning is to select a function, which approximates best the system's response, from the possible function set $f(x,\alpha)\alpha \in \Theta$ (also called hypothesis space).

Support Vector Regression (SVR) has embodied the idea of capacity control. It maps the input space X into a high dimension (or infinite) feature space F, and then constructs a linear function with small weight under some constraints in feature space. The detail is as follows:

The loss function that SVR chooses is $\varepsilon-$ insensitive loss function, that is

$$L(y, f(x,\alpha)) = \begin{cases} |y - f(x,\alpha)| - \varepsilon & if\ |y - f(x,\alpha)| > \varepsilon \\ 0, & others \end{cases}$$

The feature space F is obtained from map function $\phi(x)$, and SVR uses the following to approximate our learning aim:

$$f(x) = w'\phi(x) + b \tag{3}$$

The weight $w \in F$ and bias $b \in R$ are gained by optimizing the following problem:

$$minimize \quad \frac{1}{2}\|w\|^2 \tag{4}$$

$$s.t \quad \begin{cases} y_i - w'\phi(x_i) - b \le \varepsilon \\ w'\phi(x_i) + b - y_i \le \varepsilon \end{cases}$$

The tacit assumption in the above optimization problem is that the training samples could be approximated within precision ε by our linear function in feature space. However in practice, there always exists various noise or imprecision for our training samples, so it's hard to realize this ideal case. Instead we solve the following soft margin optimization:

$$minimize \quad \frac{1}{2}\|w\|^2 + C\sum_i(\xi_i + \xi_i^*) \tag{5}$$

$$s.t \quad \begin{cases} y_i - w'\phi(x_i) - b \le \varepsilon + \xi_i \\ w'\phi(x_i) + b - y_i \le \varepsilon + \xi_i^* \\ \xi_i, \xi_i^* \ge 0 \end{cases}$$

ξ_i and ξ_i^* are slack variables. Constant C is a parameter determining the trade-off between the complexity of $f(x)$ and the amount up to which deviations larger than ε are tolerated. The optimization problem (5) is usually translated into its dual form by the Lagrange multiplier, after some computation. The dual problem of (5) is to

$$maximize \begin{cases} -\frac{1}{2}\sum_{i,j}(\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*)\phi(x_i)'\phi(x_j) \\ -\varepsilon\sum_i(\alpha_i + \alpha_i^*) + \sum_i y_i(\alpha_i - \alpha_i^*) \end{cases}$$

$$s.t \quad \begin{cases} \sum_i (\alpha_i - \alpha_i^*) = 0 \\ 0 \le \alpha_i, \alpha_i^* \le C \end{cases}$$

And the final decision function is that $w = \sum_i (\alpha_i - \alpha_i^*) \phi(x_i)$

$$f(x) = \sum_i (\alpha_i - \alpha_i^*) \phi(x_i)' \phi(x) + b \qquad (6)$$

Now another problem comes up. We have mentioned that the dimension of feature space is always high or even infinite. So it's impossible to compute the inner product directly in feature space. Kernel idea is a key step for SVR. Kernel function could make the computation in feature space more easier in the in original space.

A kernel function $K(x, y)$ defined on the input space $X \times X$ is that

$$K(x, y) = \phi(x)' \phi(y)$$

So (6) is changed to

$$f(x) = \sum_i (\alpha_i - \alpha_i^*) k(x_i, x) + b \qquad (7)$$

According KKT conditions, the bias b can be got by solving the following four equations:

$$\alpha_i(\varepsilon + \xi_i - y_i + w'\phi(x_i) + b) = 0$$

$$\alpha_i^*(\varepsilon + \xi_i^* + y_i - w'\phi(x_i) - b = 0$$

$$(C - \alpha_i)\xi_i = 0$$

$$(C - \alpha_i^*)\xi_i^* = 0$$

Gaussian kernel, Polynomial kernel, sigmoid kernel and ANOVA kernel are commonly used kernels in practical application. They're listed in table 1. For the existence theorem of kernel and kernel construction, you could see [7].

Table 1 commonly used kernel

Type	definition
Polynomial kernel	$k(x, x') = (x^T x' + c)^d$
Gaussian kernel	$k(x, x') = \exp(-\frac{\lVert x - x' \rVert^2}{2\sigma^2}), \sigma$ denotes kernel width
Sigmoid kernel	$k(x, x') = \tanh(k \cdot (x^T x') + b)$

3. Ensemble method

For our learning task of regression, An ensemble of regressions is a set of regressions whose individual decisions are combined in some way(typically by weighted or unweighted combining)to predict new examples. The main discovery is that ensembles are often much more accurate than the individual regression that make them up. Bagging and boosting, two popular ensemble methods, will be used to combine SVMs. The common strategy for them is to manipulate the training set. The learning algorithm is run several times, each time with a different subset (or distribution) of the training examples. This technique works especially well for unstable learning algorithms—— algorithms whose output regression undergoes major change in response to small changes in the training data.

A. Bagging

The most straightforward way of manipulating the training set is called bagging. On each run, Bagging presents the leaning algorithm with a training set that consists of a sample of l training examples drawn randomly with replacement from the original training set of l items. Such a training set is called a bootstrap replicate of the original training set, and the technique is called bootstrap aggregation by Breiman[4]. The pseudo code for bagging SVR is described as follows:

```
Input: Training samples $S = (x_i, y_i), i = 1,...,l$.
Output: learner $SVR_{S^i}, i = 1,...,T$.

For i=1 to T
    Resampling $S$ with uniform distribution, and get training sample $S_i$;
    Training SVR with $S_i$ and get learner $SVR_{S^i}$.
End
```

The final decision of bagging SVR is $SVR_{bagging}(x) = \frac{1}{T}\sum_i SVR_i(x)$.

According to [4] [5], both theoretical analysis and experiments have demonstrated that bagging can push a good but unstable procedure a significant step towards optimality. The main cause for bagging to improve generalization performance is that it could reduce variance of bias/variance conflict but not bias. In other words, it can slightly degrade the performance of stable procedures.

B. Boosting

Boosting is a general method for improving the accuracy of any given learning algorithm, which could be thought of as one important advance in recent computation learning theory. The great virtue of boosting is that it could boost "weak" learning algorithm, which performs just slightly better than random guessing, into a relatively "strong" learning algorithm. Each time it is called, the base learning algorithm generates a new weak prediction rule, and after many rounds, the boosting algorithm must combine these weak rules into a single prediction rule that, hopefully, will be much more accurate than any one of the weak rules.

The Boosting algorithm has evolved several times. When it was first been proposed by Schapire in his seminar paper[9] for classification problem, the combining aim was only three classifiers. Trained on progressively difficult part of the available data; their decisions are combined by a majority vote. The main defect for this method is that huge amount of training samples are needed and the training is time-consuming. In practical world, training samples are always hard to achieve. Subsequent representatives of boosting method concern how to get a more convenient method for the training of boost and extend it to regression.

AdaBoost algorithm, introduced in 1995 by Freund and Schapire [22], solved many of the practical difficulties of the earlier boosting algorithms. And it has undergone intense theoretical study and empirical testing. Much has been reported about the success of AdaBoost in producing accurate classifiers. Interestingly, in many examples the test error seems to consistently decrease and then level off as its size becomes very large, rather than ultimately increase. It seems to conflict with Occam's principle, one of the fundamental principles in the theory of machine learning. This principle states that in order to achieve good prediction error, the learner should be as simple as possible. By "simple," we mean that the learner is chosen from a restricted space of learners. When the space is finite, we use its cardinality as the measure of complexity and when it is infinite we use the VC dimension [23] which is often closely related to the number of parameters that define the learner.

There must be some reason for AdaBoost to be resistant to overfitting. Several authors [10][24][25] attempt to use the insights gleaned from the theory of margins to give a qualitative explanation of the effectiveness of boosting. In addition, the margin theory points to a strong connection between boosting and the support-vector machines of Vapnik. Friedman, Hastie and Tibshirani [8] have found that the boosting is a gradient descent optimization algorithm for special object function.

To make boosting algorithm work in our prediction of time series, there are two fundamental questions that must be answered: first, how should each distribution be chosen on each round, and second, how should the weak regression be combined into a single regression? Regarding the choice of distribution, the technique that we advocate is to place the most weight on the examples most often predicted badly by the preceding weak regression; this has the effect of forcing the base learner to focus its attention on the "hardest" examples.

To combine the regressors we'll use the weighted median [2][6], which is demonstrated to be less sensitive to outliers, rather than the weighted mean. Latter experiments also show this. For the updating of the distribution on the training set, we'll evaluate linear, squared and exponential loss functions suggested in [1][15]. Illuminated by the success of gradient descent optimization method applying in the explanation of classifier boosting, researchers have extended it to regression case and a new approach to regressor boosting as residual-fitting [6][19][26]was developed. The main idea for this method is that each round a regressor is trained on a new training set having different target values (e.g. the residual error of the sum of the previous regressors), instead of on a different distribution of the same training set. However, we don't plan to use this new technique. According to the characteristic of SVR, the weighted ε -insensitive loss is used to train our SVR.

Our boosting SVR algorithm is described as follows:

Input: Training samples $S=(x_i, y_i)$, $i=1,...,l$.

Output: learner SVR_{S^i} , $i=1,...,T$ and ε_t .

Initialize $D_1(i)=\frac{1}{l}$ for $i=1,...,l$,set t=1.

For i=1 to T until $\hat{L}$ <1/2:

(1)Develop an SVR by using the entire training set and weighting the $\varepsilon-$ insensitive loss computed for example i by $D_t(i)$, the weight of example i for the iteration t.

(2)Update the distribution of the training samples:

One of the three loss function is used:

—Linear: $L_i = | SVR_t(x_i) - y_i | / L_{\max}$,

—Square: $L_i = | SVR_t(x_i) - y_i |^2 / L_{\max}^2$,

—

Exponential: $L_i = 1-\exp[-| SVR_t(x_i) - y_i | / L_{\max}]$,

Here, $L_{\max} = \sup_i | SVR_t(x_i) - y_i |$;

Compute $\hat{L} = \sum_{i=1}^{l} L_i D_t(i)$, $\varepsilon_t = (1-\hat{L})/\hat{L}$;

The distribution is updated as:

$$D_{t+1}(i) = \frac{D_t(i)\varepsilon_t^{1-L_i}}{Z_t}$$

Where Z_t is a normalization constant chosen such that D_{t+1} is a distribution.

Training SVR with weighted S_i and get learner SVR_{S^i}.

End.

The final decision of boosting SVR

A. Weighted median combination:

$$SVR_{boosting}(x) =$$

$$\inf\{y : \sum_{t:SVM_t(x)\le y} \log\varepsilon_t \ge \frac{1}{2}\sum_t \log\varepsilon_t\}.$$

B. Weighted mean combination:

$$SVR_{boosting}(x) = \sum_{t=1}^{T} [\varepsilon_t \cdot SVR_t(x)] / \sum_t \varepsilon_t .$$

4. Experiment and result

We'll perform our ensemble SVR algorithm on two different datasets, sunspots (a natural one) and Mackey-Glass(a synthetic one). The sunspots dataset consists of a total of 280 year sunspots recording from 1700 to 1979. The data from 1700 to 1920 is used for the training sample and the data from 1921 to 1979 for the test sample. The Mackey-Glass benchmarks are generated by a nonlinear differential equation (8) and are well-known for the evaluation of forecasting methods. Same as [2], we considered here the data generated according to [27] for a time constant τ =17 and a sampling period of 6 (MG17). The first 500 values were employed for the training sample and the next 100 values for the test sample.

$$\frac{ds(t)}{t} = -bs(t) + a\frac{s(t-\tau)}{1+s(t-\tau)^{10}},$$

with $a = 0.2$, $b = 0.1$ (8)

The prediction performance is evaluated using the following statistical metrics, namely, the normalized mean squared error (NMSE), which is defined as

$$\text{NMSE}=\frac{1}{n\sigma^2}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

Here

$$\sigma^2 = \frac{1}{n-1}\sum_{i=1}^{n}(y_i - \bar{y})^2, \quad \bar{y} = \sum_{i=1}^{n} y_i$$

For the training of weighted SVR, we use the sequential minimal optimization (SMO) algorithm developed by Platt[16].SMO is the simplest case of decomposition methods. In each iteration, it solves a quadratic problem of size two .This can be done analytically and thus no quadratic optimizer is required. The Gaussian kernel is used as the kernel function of SVR. The optimal values of the Gaussian parameter σ, trade-off coefficient C and insensitive loss ε are chosen based on the validation set.

Method	NMSE
SVR	0.64
Bagging SVR	0.58
BoostingSVR(weighted median)(linear/square/exponential)	0.48/0.36/0.27
Boosting SVR(weighted mean) (linear/square/exponential)	0.52/0.44/0.33

Figure 1 Results for sunspot on the test set

Method	NMSE
SVR	4.5E-3
Bagging SVR	2.0E-3
BoostingSVR(weighted median)(linear/square/exponential)	1.2E-3/6.4E-4/8.2E-4
Boosting SVR(weighted mean) (linear/square/exponential)	1.0E-3/8.0E-4/9.6E-4

Figure 2 Results for Mackay- Glass on the test set

From the above results we could see that bagging and boosting could all improve the performance of single SVR, and comparing with bagging, boosting showed a better performance. This is in consistent with the theoretical analysis. That is bagging could only reduce the variance; however boosting could reduce variance and bias. For the final combination of boosting, the weighted median is a better choice for combining the regressors than the weighted mean.

5. Conclusions and future work

We adapted two famous ensemble methods, bagging and boosting, to combine SVR in the nonlinear prediction of time series.

The experimental results we obtained show that bagging and boosting could all improve the performance of single SVR. And for boosting SVR, the weighted median is a better choice for combining the regressors than the weighted mean.

As proposed in [28], the performance of boosting method will degrade as the amount of noise increases. Further research will be in the design of robust boosting algorithm under the background of time series. Another direction of our future work could be concentrated on ensemble SVR for different kernel functions.

Acknowledgments

Thank Mr. Li wei for helpful discussion. The work was partially supported by Research Center of Intelligence Technology, BUPT.

References

[1] Drucker H. Improving Regressors using Boosting Techniques. Fourteenth International Conference on Machine Learning, pp. 107-115, 1997.

[2] R. Boné, M. Assaad, M. Crucianu.Boosting Recurrent Neural Networks for Time Series Prediction. RFAI Publication, Artificial Neural Nets and Genetic Algorithms, Proceedings of the International Conference in Roanne (France), Springer Computer Science, April 2003, pp. 18-22.

[3] Robert E. Schapire. The Boosting Approach to Machine Learning:An Overview. MSRI Workshop on Nonlinear Estimation and Classification, 2002.

[4] Leo Breiman. Bagging predictors. Technical Report 421, Department of Statistics, University of California at Berkeley, 1994.

[5] Leo Breiman. Bias, variance, and arcing classifiers. Unpublished manuscript,1996.

[6] Nigel Duffy and David Helmbold. Boosting methods for regression. Machine Learning, 49(2/3), 2002.

[7] Yoav Freund and Robert E. Schapire. Experiments with a new boosting algorithm. In Machine Learning: Proceedings of the Thirteenth International Conference, pages 148–156, 1996.

[8] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. Additive logistic regression: A statistical view of boosting. The Annals of Statistics, 38(2):337–374, April 2000.

[9] Robert E. Schapire. The strength of weak learnability. Machine Learning, 5(2):197–227, 1990.

[10] Robert E. Schapire, Yoav Freund, Peter Bartlett, and Wee Sun Lee. Boosting the margin: A new explanation for the effectiveness of voting methods. The Annals of Statistics, 26(5):1651–1686, October 1998.

[11] J.Platt. Fast training of support vector machines using sequential minimal optimization. In B.Schölkopf, C.J.C.Burges, and A.J.Smola,editors,Advances in Kernel Methods—support vector Learning, pages 185-208, Cambridge, MA,1999.MIT Press.

[12] K. R. Muller, J. A. Smola, G. Ratsch, B. Scholkopf, and J. Kohlmorgen, "Prediction time series with support vector Machine," in Advances in Kernel Methods. London, U.K.: MIT Press, 1999.

[13] Freund,Y. (1995). Boosting a weak learning algorithm by majority. Information and Computation, 121:2, 256–285.

[14] Kearns, M. J., & Vazirani, U. V. (1994). An introduction to computational learning theory. Cambridge, MA: The MIT Press.

[15] Harris Drucker. Boosing using neural network. In A.J.C. Sharkey, editor, Combining Artificial Neural Nets: Ensemble and Modular Learning, pages 51-77. Springer, 1999.

[16] G.E.P.Box and G.M.Jenkins. Time Series Analysis,Forescating, and Control. Holden-Day,San Fransisco,2nd edition 1976.

[17] M.B.Priestley. Non-linear and Non-stationary Time series. Academic Press, London,1988.

[18] James M. Hutchinson. A Radial Basis Function Approach to Financial Time Series Analysis. Ph.D. Thesis, Massachusetts Institute of Technology, Department of Electrical Engineering and Computer Science.

[19] Friedman, J. H. Greedy Function Approximation: a Gradient Boosting Machine. Technical Report, Dept. of Statistics, Stanford University, 36 p, 2000.

[20] Cook, G.D., Robinson, A.J. Boosting the Performance of Connectionist Large Vocabulary Speech Recognition. In International Conf. in Spoken Language Processing, pp. 1305-1308. Philadelphia, 1996.

[21] Avnimelech R., Intrator N. Boosting Regression Estimators. Neural Computation 11: 491-513, 1999.

[22] Yoav Freund and Robert E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. Journal of Computer and System Sciences,55(1):119–139, August 1997.

[23] V. Vapnik. Statistical Learning Theory. John Wiley and Sons, New York, 1998.

[24] Vladimir Koltchinskii, Dmitriy Panchenko, and Fernando Lozano. Some new bounds on the generalization error of combined classifiers. In Advances in Neural Information Processing Systems 13, 2001.

[25] Llew Mason, Peter Bartlett, and Jonathan Baxter. Direct optimization of margins improves generalization in combined classifiers. In Advances in Neural Information Processing Systems 12, 2000.

[26] Ridgeway G., Madigan D., Richardson T. Boosting Methodology for Regression Problems. Artificial Intelligence and Statistics, pp. 152-161, 1999.

[27] Back A., Wan E.A., Lawrence S., Tsoi A.C. A Unifying View of some Training Algorithms for MLPs with FIR Filter Synapses. Neural Networks for Signal Processing IV, Ermioni, Greece, pp.146-154, 1994.

[28] Rätsch Gunnar. Robust Boosting via convex optimization.phd thesis.2003.