Support Vector Regression (SVR)

for regression

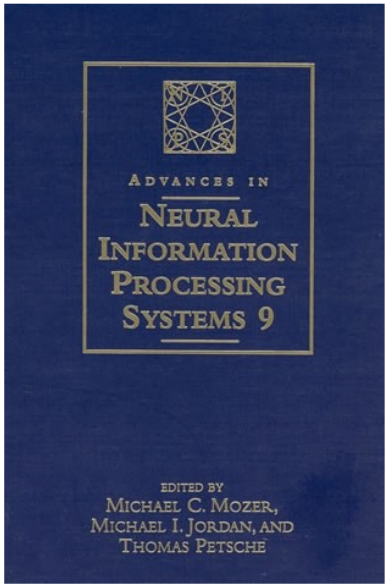
Freddy Hernández

11 de octubre de 2019

Support Vector Regression

SVR were developed by Drucker et al. (1996)

Url: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.21.5909



Support Vector Regression Machines

Harris Drucker* Chris J.C. Burges Linda Kaufman****
Alex Smola Vladimir Vapnik [+]**

*Bell Labs and Monmouth University
Department of Electronic Engineering
West Long Branch, NJ 07764
**Bell Labs [+]AT&T Labs

Abstract

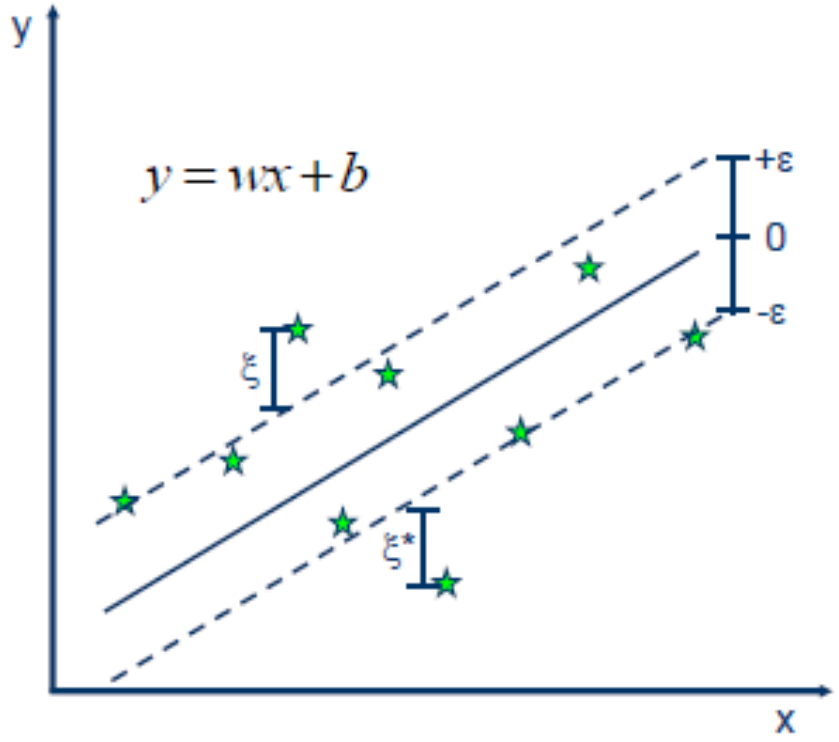
A new regression technique based on Vapnik's concept of support vectors is introduced. We compare support vector regression (SVR) with a committee regression technique (bagging) based on regression trees and ridge regression done in feature space. On the basis of these experiments, it is expected that SVR will have advantages in high dimensionality space because SVR optimization does not depend on the dimensionality of the input space.

An overview of Support Vector Machines

https://www.svm-tutorial.com/2017/02/svms-overview-support-vector-machines/

Idea

The objective of the Support Vector Machine algorithm is to find a margin of tolerance (ϵ) to minimize error, individualizing the hyperplane which maximizes the margin, keeping in mind that part of the error is tolerated.



- Minimize:

$$\frac{1}{2}\|w\|^2 + C\sum_{i=1}^{N}\left(\xi_i + \xi_i^*\right)$$

- Constraints:

$$y_i - wx_i - b \le \varepsilon + \xi_i$$

$$wx_i + b - y_i \le \varepsilon + \xi_i^*$$

$$\xi_i, \xi_i^* \ge 0$$

R package

The package **e1071** contains the `svm` function used for SVMs. To install the package:

```
install.packages("e1071")
```

To load the package:

```
library(e1071)
```

The main function is:

```
svm(formula, data, scale=TRUE,
    kernel=linear or polynomial or radial or sigmoid,
    degree=3, gamma=1/n, coef0=0, cost=1, ...)
```

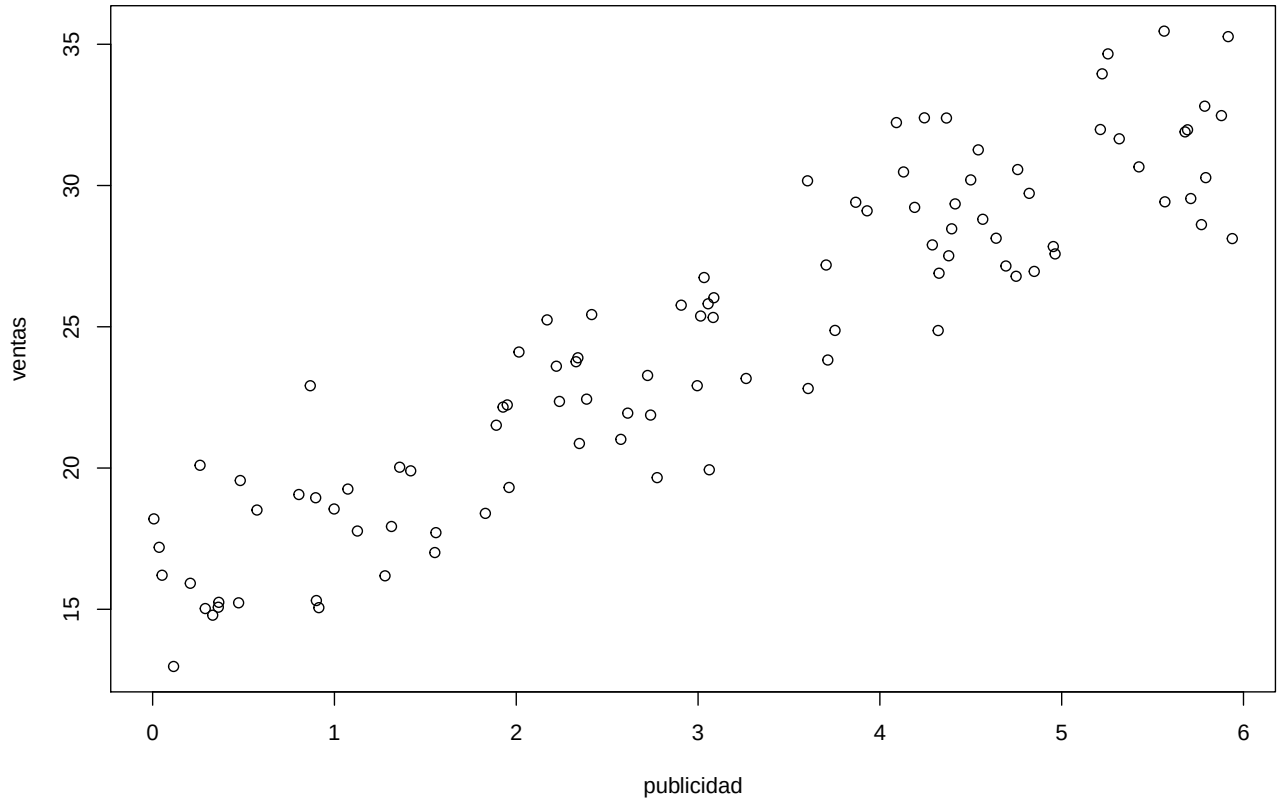
Consult the vignette:

https://cran.r-project.org/web/packages/e1071/vignettes/svmdoc.pdf

Example 1

Let's first download some `Train` data.

```
url <- "https://raw.githubusercontent.com/rdaymedellin/
tutoriales_Rday_2019/master/Machine%20Learning/publi_ventas.txt"
Train <- read.table(url, sep=";", header=TRUE)
with(Train, plot(x=publicidad, y=ventas))
```



Example 1

Exploring the dimensions and the content of `Train` dataset.

```
dim(Train)
```

```
## [1] 100   2
```

```
head(Train)
```

```
##   publicidad   ventas
## 1  4.3254230 26.89550
## 2  5.2546390 34.65930
## 3  4.5658940 28.80486
## 4  5.3167470 31.65357
## 5  2.7388860 21.87470
## 6  0.9982307 18.55060
```

Example 1

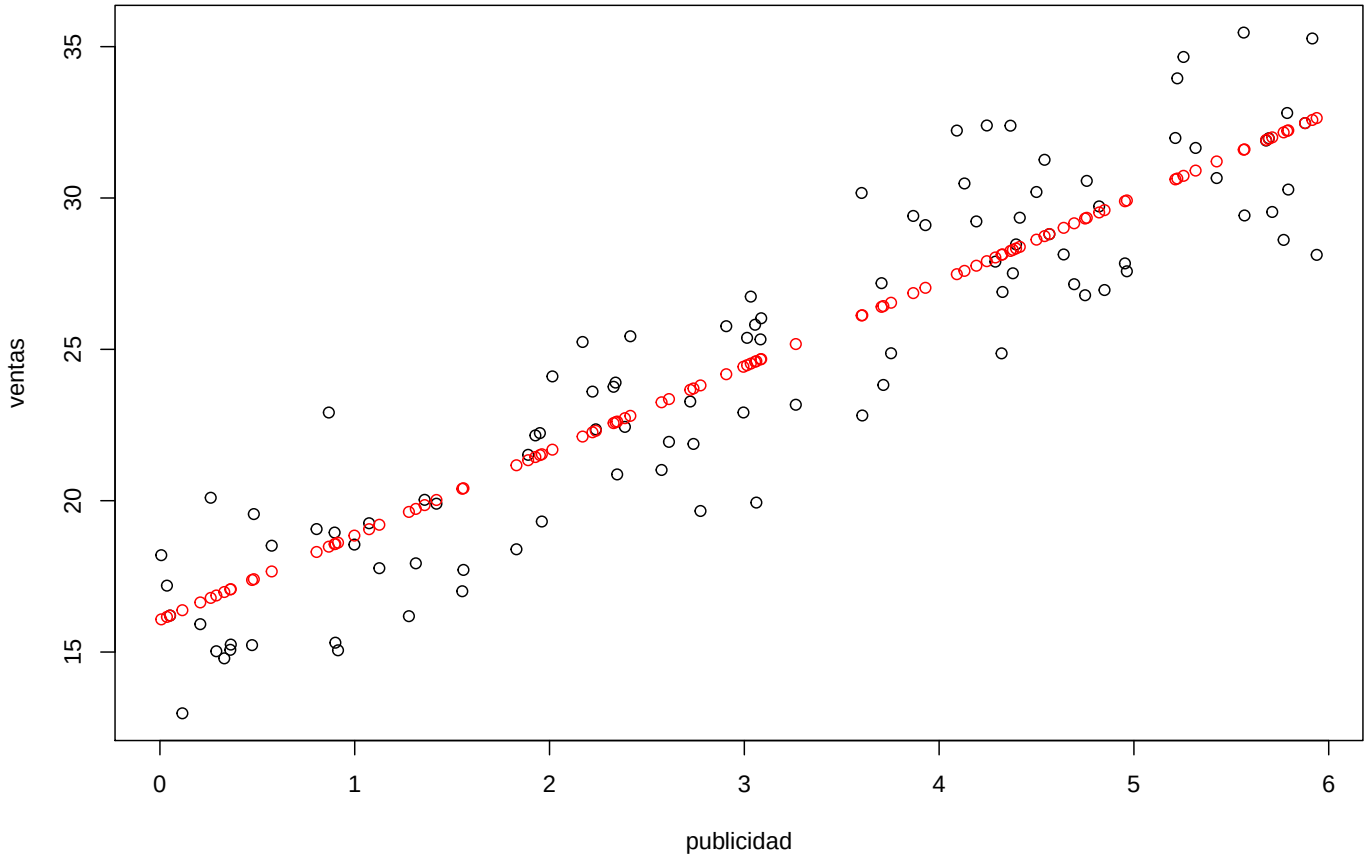
```
svm_lin <- svm(ventas ~ publicidad, data=Train, epsilon=0,
               kernel="linear", scale=TRUE)
print(svm_lin)
```

```
##
## Call:
## svm(formula = ventas ~ publicidad, data = Train, epsilon = 0,
##     kernel = "linear", scale = TRUE)
##
##
## Parameters:
##    SVM-Type:  eps-regression
##  SVM-Kernel:  linear
##        cost:  1
##       gamma:  1
##     epsilon:  0
##
##
## Number of Support Vectors:  100
```

Example 1

The midline.

```
predY <- predict(svm_lin, Train)
with(Train, plot(x=publicidad, y=ventas))
points(x=Train$publicidad, y=predY, col="red")
```



Tuning parameters

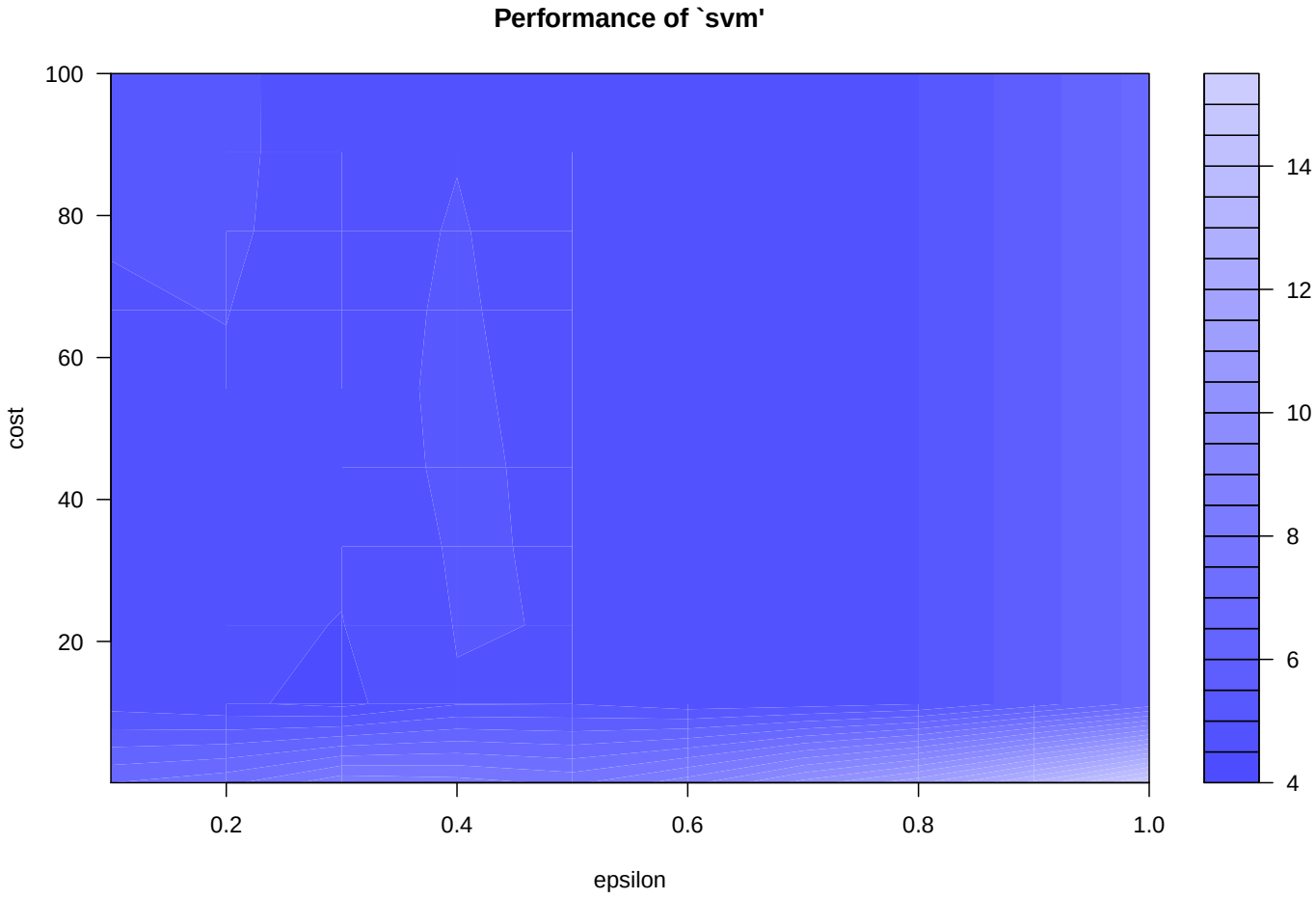
The tune parameters can be found using the tune function. As performance measure, the classification error is used for classification, and the mean squared error for regression.

```
obj <- tune(svm, ventas ~ publicidad, data=Train,
            ranges = list(epsilon=seq(from=0.1, to=1, length.out=10),
                          cost=seq(from=0.1, to=100, length.out=10)),
            tunecontrol = tune.control(sampling="fix"))
obj
```

```
##
## Parameter tuning of 'svm':
##
## - sampling method: fixed training/validation set
##
## - best parameters:
##  epsilon cost
##      0.3 11.2
##
## - best performance: 4.358442
```

Example 1

```
plot(obj)
```



Pros and Cons associated with SVM

Pros:

- It works really well with clear margin of separation.
- It is effective in high dimensional spaces.
- It is effective in cases where number of dimensions is greater than the number of samples.
- It uses a subset of training points in the decision function (called support vectors), so it is also memory efficient.

Cons:

- It doesn't perform well, when we have large data set because the required training time is higher.
- It also doesn't perform very well, when the data set has more noise i.e. target classes are overlapping.
- SVM doesn't directly provide probability estimates, these are calculated using an expensive five-fold cross-validation. It is related SVC method of Python scikit-learn library.