02471 Machine Learning for Signal Processing

Week 13: Support vector regression

This exercise is based on S. Theodoridis: Machine Learning, A Bayesian and Optimization Perspective 2nd edition, section(s) 11.8.

The objective of this exercise is to get more familiar with kernel methods and expand on the exercise from last week. In particular, we analyze the support vector regression method. We will use support vector regression to perform de–noising and anomaly detection on a time–series signal.

In this exercise, we assume all inner product operations are performed in Hilbert spaces, and all kernel functions er reproducing kernels such that we operate in reproducing kernel Hilbert space.

Overview

The exercise have the following structure:

13.1 will analyze the support vector regression method, and go from the linear ϵ–insensitive loss function to the optimization problem, and gain insights to the of the method.

13.2 will perform de–noising and anomaly detection on a time–series signal using kernel ridge regression and support vector regression

You can carry out the exercises in any order. To get full benefit of exercise 13.2, you should carry out the preceding exercises.

Notation

- $J(\boldsymbol{\theta})$ is a cost function that we are seeking to minimize with respect to $\boldsymbol{\theta}$.
- $\langle \cdot, \cdot \rangle$ denotes the inner product, e.g. in $\mathbb{R}^N$, we have $\langle \boldsymbol{x}, \boldsymbol{y} \rangle = \boldsymbol{x}^T \boldsymbol{y} = \boldsymbol{x} \cdot \boldsymbol{y}$. If we have $\langle \boldsymbol{x}, \boldsymbol{x} \rangle$ (and we have a Hilbert space), the inner product denotes the norm of the space. In $\mathbb{R}^N$ we have $\langle \boldsymbol{x}, \boldsymbol{x} \rangle = \|\boldsymbol{x}\|_2^2$.
- $\kappa(\boldsymbol{x}, \boldsymbol{y})$ denotes a kernel function. $\kappa(\boldsymbol{x}, \boldsymbol{y})$ is symmetric, i.e. $\kappa(\boldsymbol{x}, \boldsymbol{y}) = \kappa(\boldsymbol{y}, \boldsymbol{x})$.
- $\mathcal{K}$ denotes a kernel matrix (defined in Eq. 11.11 in the book). $\mathcal{K}$ is symmetric: $\mathcal{K} = \mathcal{K}^T$.

Code

The code can be found in the .m and .py files named in the same way as exercises, ie. the code for exercise 13.x.y is in the file 13_x_y.m (or .py).

For coding exercises that requires implementation we will usually write `complete this line` where the implementing should be done.

Solutions

The solution is provided for all derivation exercises, and often hints are provided at the end of the document. If you get stuck, take a look at the hints, and if you are still stuck, take a look in the solution to see the approach being taken. Then try to do it on your own.

Solutions are also provided for some coding exercises. If you get stuck, take a look at the solution, and then try to implement it on your own.

13.1 Support vector regression (SVR)

In this exercise we are going to analyze the solution offered by the support vector regression method.

Make sure to skim section 11.8 in the book before carrying out this exercise.

At stated in the lecture, a regression task on the form

$$y_n = g(\boldsymbol{x}_n) + \eta_n, \quad n = 1, 2, \cdots, N$$

where η_n is i.i.d. noise, with the loss

$$\mathcal{L}(y, f(\boldsymbol{x})) = \begin{cases} |y - f(\boldsymbol{x})| - \epsilon, & \text{if} \quad |y - f(\boldsymbol{x})| > \epsilon \\ 0, & \text{if} \quad |y - f(\boldsymbol{x})| \leq \epsilon \end{cases} \tag{1}$$

can be written as the following optimization problem

$$\underset{\boldsymbol{\theta}, \boldsymbol{\xi}, \tilde{\boldsymbol{\xi}},}{\arg\min} \quad J(\boldsymbol{\theta}, \boldsymbol{\xi}, \tilde{\boldsymbol{\xi}}) := \frac{1}{2}\|\boldsymbol{\theta}\|^2 + C\left(\sum_{n=1}^{N} \xi_n + \sum_{n=1}^{N} \tilde{\xi}_n\right) \tag{2}$$

$$\text{s.t.} \qquad y_n - f(\boldsymbol{x}_n) \leq \epsilon + \tilde{\xi}_n \tag{3}$$

$$-(y_n - f(\boldsymbol{x}_n)) \leq \epsilon + \xi_n \tag{4}$$

$$\tilde{\xi}_n \geq 0 \tag{5}$$

$$\xi_n \geq 0 \tag{6}$$

To give you an idea of where we are going, the final optimization problem for the support vector regression method can be written as

$$\hat{y} = \sum_{n=1}^{N} (\tilde{\lambda}_n - \lambda_n) \kappa(\boldsymbol{x}_n, \boldsymbol{x}) + \hat{\theta}_0 \tag{7}$$

$$\underset{\boldsymbol{\lambda}, \tilde{\boldsymbol{\lambda}},}{\arg\max} \quad \sum_{n=1}^{N} (\tilde{\lambda}_n - \lambda_n) y_n - \epsilon(\tilde{\lambda}_n - \lambda_n) \tag{8}$$

$$-\frac{1}{2} \sum_{n=1}^{N} \sum_{m=1}^{N} (\tilde{\lambda}_n - \lambda_n)(\tilde{\lambda}_m - \lambda_m) \kappa(\boldsymbol{x}_n, \boldsymbol{x}_m) \tag{9}$$

$$\text{s.t.} \quad 0 \leq \tilde{\lambda}_n \leq C, \quad 0 \leq \lambda_n \leq C, \quad n = 1, 2, \cdots, N \tag{10}$$

$$\sum_{n=1}^{N} \tilde{\lambda}_n = \sum_{n=1}^{N} \lambda_n \tag{11}$$

Inspection of this problem reveals 3 choices of importance, the choice of C, which bounds the value of $\tilde{\lambda}_n$ and λ_n, the choice of ϵ, and the choice of kernel function, $\kappa(\boldsymbol{x}_n, \boldsymbol{x}_m)$.

We are going to analyze the solution in order to get a better understanding of the parameter choice of ϵ and C, and their impact on the predictions.

Exercise 13.1.1 (This is a difficult exercise)

For now, we will return to the original stated problem in eq (1), and we assume a linear model $f(\boldsymbol{x}) = \boldsymbol{\theta}^T\boldsymbol{x} + \theta_0$.

Argue why this loss makes sense (compared to the least squares loss) if the data is contaminated by outliers, and show the loss stated in eq (1) can be rewritten to eq. (2–6) (with an added regularization on $\boldsymbol{\theta}$).

Exercise 13.1.2

The learning problem in eq. (2–6) is stated in a version where it can readily be solved by using Lagrange multipliers.

From appendix C.2, we have the following result:

$$\begin{aligned} \min_{\boldsymbol{\theta}} \quad & J(\boldsymbol{\theta}) \\ \text{s.t.} \quad & f_i(\boldsymbol{\theta}) \geq 0, \quad i = 1, 2, \cdots, m \end{aligned}$$

then the Lagrangian is

$$\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\lambda}) = J(\boldsymbol{\theta}) - \sum_{i=1}^{m} \lambda_i f_i(\boldsymbol{\theta})$$

And is solved by:

$$\begin{aligned} & \frac{\partial}{\partial \boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\lambda}) \bigg|_{\theta=\theta_*} = \mathbf{0} \\ & \lambda_i \geq 0 \qquad\quad i = 1, 2, \cdots, m \\ & \lambda_i f_i(\boldsymbol{\theta}_*) = 0 \quad i = 1, 2, \cdots, m \end{aligned}$$

That tells us immediately how to convert this problem. Show that, by introducing Lagrange multipliers, that you arrive at the following problem:

$$\begin{aligned} \mathcal{L}(\boldsymbol{\theta}, \theta_0, \boldsymbol{\xi}, \tilde{\boldsymbol{\xi}}, \boldsymbol{\lambda}, \tilde{\boldsymbol{\lambda}}, \boldsymbol{\mu}, \tilde{\boldsymbol{\mu}}) \quad & = \frac{1}{2}\|\boldsymbol{\theta}\|^2 + C\left(\sum_{n=1}^{N} \xi_n + \sum_{n=1}^{N} \tilde{\xi}_n\right) \\ & + \sum_{n=1}^{N} \tilde{\lambda}_n (y_n - \boldsymbol{\theta}^T\boldsymbol{x}_n - \theta_0 - \epsilon - \tilde{\xi}_n) \\ & + \sum_{n=1}^{N} \lambda_n (-y_n + \boldsymbol{\theta}^T\boldsymbol{x}_n + \theta_0 - \epsilon - \xi_n) \\ & - \sum_{n=1}^{N} \tilde{\mu}_n \tilde{\xi}_n - \sum_{n=1}^{N} \mu_n \xi_n \qquad (12) \end{aligned}$$

$$\text{s.t.} \quad \tilde{\lambda}_n (y_n - \boldsymbol{\theta}^T\boldsymbol{x}_n - \theta_0 - \epsilon - \tilde{\xi}_n) = 0, \quad n = 1, 2, \cdots, N \qquad (13)$$

$$\lambda_n (\boldsymbol{\theta}^T\boldsymbol{x}_n + \theta_0 - y_n - \epsilon - \xi_n) = 0, \quad n = 1, 2, \cdots, N \qquad (14)$$

$$\tilde{\mu}_n \tilde{\xi}_n = 0, \quad \mu_n \xi_n = 0, \quad n = 1, 2, \cdots, N \qquad (15)$$

$$\tilde{\lambda}_n \geq 0,\ \lambda_n \geq 0,\ \tilde{\mu}_n \geq 0,\ \mu_n \geq 0, \quad n = 1, 2, \cdots, N \qquad (16)$$

Exercise 13.1.3

The next step is to find the derivative w.r.t. the learning parameters, and set those derivatives to zero, that is

$$\frac{\partial}{\partial \boldsymbol{\theta}}\mathcal{L}(\cdot) = \mathbf{0}, \quad \frac{\partial}{\partial \theta_0}\mathcal{L}(\cdot) = 0, \quad \frac{\partial}{\partial \tilde{\xi}_n}\mathcal{L}(\cdot) = 0, \quad \frac{\partial}{\partial \xi_n}\mathcal{L}(\cdot) = 0$$

Show that solving these equations leads to the following solutions

$$\hat{\boldsymbol{\theta}} = \sum_{n=1}^{N}(\tilde{\lambda}_n - \lambda_n)\boldsymbol{x}_n \tag{17}$$

$$\sum_{n=1}^{N}\tilde{\lambda}_n = \sum_{n=1}^{N}\lambda_n \tag{18}$$

$$C = \tilde{\lambda}_n + \tilde{\mu}_n \tag{19}$$

$$C = \lambda_n + \mu_n \tag{20}$$

Exercise 13.1.4 (This is a difficult exercise)

We have a lot of equations that must be fulfilled, and this has some side-effects on the solution.

Work through all the following statements, and show they are true by identifying and manipulating the correct constraints.

First we look at the case where $\xi_n > 0$ (that is, y_n is outside the ϵ-tube), this implies

$$\xi_n > 0 \Rightarrow \mu_n = 0 \Rightarrow \lambda = C$$

For the case $\xi_n = 0$, y_n is either on the margin of the ϵ-tube or inside the ϵ-tube.

We define $e_n := |f(\boldsymbol{x}_n) - y_n|$ and first look on the case where e_n is on the margin, ie. we have $e_n = \epsilon$. This implies λ_n is a free parameter, i.e. one constraint reduces to

$$e_n = \epsilon \Rightarrow \lambda_n \cdot 0 = 0$$

This implies λ_n is "free" to be any value. However due to the previous non-negativity constraints, we have $0 \leq \lambda_n \leq C$ when $e_n = \epsilon$.

In the third situation, we have the prediction inside the margin, i.e. $e_n < \epsilon$. This situation does not result in the previous reduction $\lambda_n \cdot 0 = 0$, but instead

$$e_n < \epsilon \Rightarrow \lambda_n \cdot \delta\epsilon = 0$$

where $\delta\epsilon = \epsilon - e_n > 0$. This forces $\lambda_n = 0$, so that means, for points that are within the ϵ-tube, $\lambda_n = 0$. That means they do not contribute to the prediction.

For this reason, we can distinguish between points that contribute to the prediction, and point that does not. The points that contribute, have positive λ_n values, and these points are called *support vectors*.

13.2 Smoothing using kernel methods

In this exercise we will do de-noising and anomaly detection using kernel methods. The corresponding material in the book is example 11.3, page 560 and exercise 11.15.

We will work on a piece of audio from the Blade Runner movie, and add both noise and anomalies (outliers) to the data. We will then use and Support Vector Regression (SVR) to reconstruct the data.

Exercise 13.2.1

Make sure to skim sec 11.8 in the book before carrying out this exercise.

Run and inspect the code associated with the exercise.

The code will load sound, extract 100 samples, and then add white Gaussian noise and randomly "hit" 10 of the data samples with outliers (set the outlier values to 80% of the maximum value of the data samples). As kernel the rbf/Gaussian kernel is used.

The code finds the reconstructed data samples , and plots the fitted curves of the reconstructed samples together with the data used for training.

Compare the solution to Kernel ridge regression from last week and explain why the SVR method works better in the presence of outliers. Experiment with the parameters C, σ, and ϵ and comment on the results.