02471 Machine Learning for Signal Processing

Week 12: Kernel methods

This exercise is based on S. Theodoridis: Machine Learning, A Bayesian and Optimization Perspective 2nd edition, section(s) 11.5–11.7.

The objective of this exercise is to get familiar with kernel methods, and in particular the kernel ridge regression where we perform smoothing (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 are reproducing kernels such that we operate in reproducing kernel Hilbert space (RKHS).

Overview

The exercise have the following structure:

12.1 will present the kernel methods in a simplified setting, and show how kernels can make non-linear separable points separable in RKHS.

12.2 will derive the solution for the kernel ridge regression method.

12.3 will perform de–noising and anomaly detection on a time–series signal using kernel ridge regression.

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 $\langle \boldsymbol{x}, \boldsymbol{x} \rangle = \|\boldsymbol{x}\|_2$.
- $\kappa(\boldsymbol{x}, \boldsymbol{y})$ 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 12.x.y is in the file 12_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.

12.1 Obtaining linear separability using kernels

This exercise will introduce you to the basic ideas of kernels. We will work in a two–dimensional toy-example to make the processes clear. We will generate two categories of points in a circular disc, so that they are not linearly separable but are clearly non-linearly separable.

Exercise 12.1.1

Run and inspect the code for this exercise. The program generates two sets of points. Identify how the points are generated and write down the formulas.

Exercise 12.1.2

The points can be transformed using the mapping function

$$\mathbb{R}^2 \ni \boldsymbol{x} \longmapsto \boldsymbol{\phi}(\boldsymbol{x}) = \left[x_1^2, \sqrt{2}x_1x_2, x_2^2\right] \in \mathbb{R}^3$$

Implement the mapping function in the code and plot the points in $3d$ and validate they are now linearly separable.

Exercise 12.1.3

A useful connection of the mapping function can be made. If we take the inner product of the mapping function, this corresponds to the squared inner product of the vectors themselves:

$$\boldsymbol{\phi}^T(\boldsymbol{x})\boldsymbol{\phi}(\boldsymbol{y}) = \left(\boldsymbol{x}^T\boldsymbol{y}\right)^2$$

Show that this relation is true.

Exercise 12.1.4

This inner product is referred to as the *homogeneous polynomial* kernel

$$\kappa(\boldsymbol{x}, \boldsymbol{y}) = (\boldsymbol{x}^T\boldsymbol{y})^r$$

Implement the kernel in the code associated with this exercise.

Exercise 12.1.5

The data has now been transformed into an inner product format where the representer theorem can now be used. The theorem loosely states that, if we have the following optimization problem

$$\min_{f\in\mathbb{H}} J(f) := \sum_{n=1}^{N} \mathcal{L}(y_n, f(\boldsymbol{x}_n)) + \lambda\Omega\left(\|f\|^2\right)$$

Then each minimizer $f \in \mathbb{H}$ of the minimization task admits a representation of the form:

$$f(\cdot) = \sum_{n=1}^{N} \theta_n \kappa(\cdot, x_n)$$

For now, we ignore the loss function and focus our attention of the form $f(\cdot)$. What operation is this function effectively carrying out? Implement and apply the function using $\boldsymbol{\theta} = \mathbf{1}$ and validate that the function is able to linearly separate our training data.

Exercise 12.1.6

Repeat the exercise using the Gaussian kernel:

$$\kappa(\boldsymbol{x}, \boldsymbol{y}) = \exp\left(-\frac{\|\boldsymbol{x} - \boldsymbol{y}\|^2}{2\sigma^2}\right)$$

Apply the representer theorem again, choosing $\boldsymbol{\theta} = \mathbf{1}$ using $\sigma^2 = 1$. Verify that this choice results in a correct separation.

Find values for σ^2 where the data is no longer separable. Explain why this happens.

12.2 Derivation of the kernel ridge regression

Make sure to skim section 11.7 in the book before carrying out this exercise.

Recall from section 3.8 that the ridge regression (without bias) minimizes the following function:

$$J_{RR}(\boldsymbol{\theta}) = \sum_{n=1}^{N} \left(y_n - \sum_{i=1}^{l} \theta_i x_{ni} \right)^2 + \lambda \sum_{i=1}^{l} |\theta_i|^2$$

The kernel ridge regression (without bias) instead minimizes

$$J(\boldsymbol{\theta}) := \sum_{n=1}^{N} \left(y_n - \sum_{m=1}^{N} \theta_m \kappa(\boldsymbol{x}_n, \boldsymbol{x}_m) \right)^2 + C\langle f, f \rangle$$

where $C \in \mathbb{R}$ is a regularization parameter. On inspection we see two things have changed from the original ridge regression loss function; the function has now changed according to the representer theorem, and we are regularizing the norm of f instead of the norm of $\boldsymbol{\theta}$.

Exercise 12.2.1

From section 8.16 (only available online), definition 8.15, we have:

Definition 8.15 (Inner product). Let V be a linear space. The inner product is a function

$$f : V \times V \to \mathbb{C}$$

which assigns a value in $\mathbb{C}$, denoted as $\langle \boldsymbol{x}, \boldsymbol{y} \rangle$, to every point of elements $\boldsymbol{x}, \boldsymbol{y} \in V$, with the following properties:

- $\langle \boldsymbol{x}, \boldsymbol{x} \rangle \geq 0$, and $\langle \boldsymbol{x}, \boldsymbol{x} \rangle = 0$ if and only if $\boldsymbol{x} = 0$.

- $\langle \boldsymbol{x} + \boldsymbol{y}, \boldsymbol{z} \rangle = \langle \boldsymbol{x}, \boldsymbol{z} \rangle + \langle \boldsymbol{y}, \boldsymbol{z} \rangle$.
- $\langle a\boldsymbol{x}, \boldsymbol{y} \rangle = a\langle \boldsymbol{x}, \boldsymbol{y} \rangle$.
- $\langle \boldsymbol{x}, \boldsymbol{y} \rangle = \langle \boldsymbol{y}, \boldsymbol{x} \rangle^*$.

Use the properties of an inner product to show that $\langle f, f \rangle = \boldsymbol{\theta}^T \mathcal{K}^T \boldsymbol{\theta}$.

Exercise 12.2.2

From the previous result it follows that

$$J(\boldsymbol{\theta}) = (\boldsymbol{y} - \mathcal{K}\boldsymbol{\theta})^T (\boldsymbol{y} - \mathcal{K}\boldsymbol{\theta}) + C\boldsymbol{\theta}^T \mathcal{K} \boldsymbol{\theta}$$

where the first term has been rewritten according to

$$\sum_{n=1}^{N} \left(y_n - \sum_{m=1}^{N} \theta_m \kappa(\boldsymbol{x}_n, \boldsymbol{x}_m) \right)^2 = (\boldsymbol{y} - \mathcal{K}\boldsymbol{\theta})^T (\boldsymbol{y} - \mathcal{K}\boldsymbol{\theta})$$

as has been done in previous exercises.

To solve the optimization task, show that we obtain

$$\frac{\partial J(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = -2(\boldsymbol{y}^T \mathcal{K})^T + 2\mathcal{K}^T \mathcal{K} \boldsymbol{\theta} + 2C\mathcal{K}^T \boldsymbol{\theta}$$

and that $\frac{\partial J(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = \mathbf{0}$ leads to the solution

$$\hat{\boldsymbol{\theta}} = (\mathcal{K} + CI)^{-1} \boldsymbol{y}$$

12.3 Smoothing using kernel ridge regression

In this exercise we will do de-noising and anomaly detection using kernel methods. The corresponding material in the book is example 11.2, page 553 and exercise 11.14.

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 kernel ridge regression to reconstruct the data.

Exercise 12.3.1

Make sure to skim sec 11.7 in the book before carrying out this exercise.

Run and inspect the code associated with the exercise. Relate the equations in section 11.7 to the code and complete the missing lines in the code.

The code will load the 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 using both the biased and unbiased kernel ridge regression method, and plots the fitted curves of the reconstructed samples together with the data used for training.

Try different values for the parameters C and σ, and comment on the results. Make sure you investigate their impact on the smoothing of the regression.

HINTS

Exercise 12.2.1

For ridge regression we define $f(\boldsymbol{x}) = \sum_{n=1}^{N} \theta_n \kappa(\boldsymbol{x}, \boldsymbol{x}_n)$. Insert this expression into the inner product and reduce to obtain the result.