Kernels

Optimization for Machine Learning — Exercise #3

Monday 15th May, 2023

Part I: Theory

I.1. Kernel Regression

Exercise I.1 (Kernel derivations)**.** Given a dataset $\mathcal{D} = \{x^{(j)}, t^{(j)}\}_{j\in[N]}$, where, for $j \in [N]$, $(x^{(j)}, t^{(j)}) \in \mathbb{R}^d \times \mathbb{R}$. Let $X \in \mathbb{R}^{d\times N}$ be the stacked samples as columns, and $t \in \mathbb{R}^N$ be the stacked targets. Denote by $\operatorname{span}(X)$ the set of vectors spanned by the columns of X: $\operatorname{span}(X) = \{y \mid \exists(\alpha_j)_{j\in[N]} \in \mathbb{R}^N,\ y = \sum_{j\in[N]} \alpha_j x^{(j)}\}$.

1. Show that any weight for a linear prediction $y(x) = \langle x, w\rangle$ is such that $w \in \operatorname{span}(X)$.
2. What if we had features, i.e. $\phi\colon \mathbb{R}^d \to \mathbb{R}^M$, such that $w \in \mathbb{R}^M$ and $y(x) = \langle \phi(x), w\rangle$?

Exercise I.2 (Primal representation – Curse of dimensionality)**.** This exercise portrays an example of a model that scales badly (exponentially) with the dimension of the samples. This phenomenon is called the curse of dimensionality.

Imagine regressing a function $f\colon \mathcal{C} \to \mathbb{R}$ on the unit cube in dimension d: $\mathcal{C} = [0,1]^d$.

One way to do it is by using M basis functions $\{\phi_i\}_{i\in[M]}$, and try to express the unknown f as a sum of those basis functions, i.e. find $w \in \mathbb{R}^M$ such that $f = \sum_{i\in[M]} w_i\phi_i$.

In Exercise Sheet #1 I.1, we have seen the polynomial example $\phi_i(x) = x^i$ for $d = 1$. An issue of the polynomial is the behaviour when x grows or shrinks at infinity.

Another typical family of functions, which does not suffer from this behaviour, is called the Radial Basis Functions (RBF). They are of this form

$$\begin{array}{cccc} \phi_i & : & \mathbb{R}^d & \longrightarrow & \mathbb{R} \\ & & x & \longmapsto & \exp\left(-\frac{\|x-\mu^{(i)}\|^2}{2\sigma^2}\right) \end{array},$$

with $i \in [M]$, $\mu^{(i)}$ different points at which the exponential are centred, and σ is the variance of the Gaussian functions.

Notice how the functions ϕ_i have to be “dense” in the cube in order to approximate the target correctly.

1. What is the issue with this formulation?
2. Using the result from Ex. I.1, what could be a solution to this problem?

Exercise I.3 (Dual representation)**.** With the notations from Ex. I.1, in the case we use a feature map $\phi\colon \mathbb{R}^d \to \mathbb{R}^M$, so that features $\Phi = \{\phi_i(x^{(j)})\}_{(i,j)\in[M]\times[N]} \in \mathbb{R}^{M\times N}$, we know that $w \in \operatorname{span}\Phi \subseteq \mathbb{R}^M$.

Let $\alpha \in \mathbb{R}^N$ be such that $w = \sum_{j\in[N]} \alpha_j \phi(x^{(j)}) =: \sum_{j\in[N]} \alpha_j \phi^{(j)}$.

For two points $(x, y) \in \mathbb{R}^d \times \mathbb{R}^d$, define the kernel $\mathcal{K}$ as

$$\mathcal{K}(x,y) := \langle \phi(x), \phi(y)\rangle = \phi(x)^\top \phi(y).$$

1. What becomes with the prediction $y(x) = \langle \phi(x), w\rangle$?
2. Define the Gram matrix K as $K_{k\ell} := \mathcal{K}(x^{(k)}, x^{(\ell)})$. What are the prediction on the complete training set Φ?
3. Using the squared-norm error $E(\alpha) = \frac{1}{2}\|y - K\alpha\|_2^2$, and assuming that K is invertible, show that
$$\alpha^* \in \operatorname*{argmin}_\alpha E(\alpha) \iff \alpha^* = \left(K^\top K\right)^{-1} K^\top y.$$
Compare with the primal problem $w^* \in \operatorname{argmin}_w \frac{1}{2}\|y - \Phi^\top w\|_2^2 \iff w^* = \left(\Phi\Phi^\top\right)^{-1}\Phi y$.
4. Compute $\mathcal{K}$ when, for all $i \in [M]$,
   a) $x \in \mathbb{R}$, $\phi_i(x) = (x)^{i-1}$
   b) $x \in \mathbb{R}^d$, $\phi_i(x) = \exp\left(-\frac{\|x-\mu^{(i)}\|^2}{2\sigma^2}\right)$
   c) $x \in \mathbb{R}$, $\phi_i(x) = \exp\left(-\frac{x^2}{2\sigma^2}\right)\frac{\left(\frac{x}{\sigma}\right)^i}{\sqrt{i!}}$, $i \to \infty$.

I.2. Kernel SVMs

Exercise I.4. Support Vector Machines can leverage kernels in order to deal with non-linearly separable data. Let $\mathcal{D} = \{(x^{(i)}, t^{(i)})\}_{i\in[n]}$ a dataset with, for $i \in [n]$, $(x^{(i)}, t^{(i)}) \in \mathbb{R}^d \times \{-1, 1\}$. Let $\phi\colon \mathbb{R}^d \to \mathbb{R}^m$ be a feature map, and $(w, b) \in \mathbb{R}^m \times \mathbb{R}$ be the weights and biases parametrizing the hyperplane in the feature space. The model is therefore $y(x) = \langle w, \phi(x)\rangle + b$.

1. Show that the distance of a point to the decision boundary is $|y(x)|/\|w\|$.
2. Explain why we can choose to make the model equal to ± 1 for the closest points to the decision boundary and conclude that the margin is $\frac{2}{\|w\|}$.
3. Write the constrained optimization problem of minimizing $\frac{1}{2}\|w\|^2$ while classifying the points correctly.

4. Write the Lagrangian $L(w, b, \alpha)$ of the problem, where $\alpha \in \mathbb{R}^n$ are the Lagrange multipliers.

5. Show that the associated dual problem is

$$\begin{aligned}
\text{maximize} \quad & g(\alpha) = \sum_{j\in[n]} \alpha_j - \frac{1}{2} \sum_{i\in[n]} \sum_{j\in[n]} t_i t_j \alpha_i \alpha_j \langle \phi(x^{(i)}), \phi(x^{(j)}) \rangle && \text{(1a)} \\
\text{subject to} \quad & \sum_{j\in[n]} \alpha_j t_j = 0, && \text{(1b)} \\
& \forall j \in [n], \alpha_j \geqslant 0. && \text{(1c)}
\end{aligned}$$

6. Is the solution found by solving the dual problem also solving the primal problem? State the condition that allows the conclude.

7. How could one go and solve the dual problem? (*Hint:* what is the nice property about g?) Why would a simple gradient ascent *not* work?

Part II: Programming

Exercise II.1 (SMO algorithm)**.** The goal of this exercise is to implement and solve the SVM problem from Ex. I.4. We will do it by solving the dual problem (1).

In order to solve (1), one possible solution is to look at g as depending on only two variables, α_k and α_ℓ, and see how one can improve the objective g while satisfying the constraints. This method is called Sequential Minimal Optimization.

Recall the constraints from (1):

$$\begin{aligned}
\sum_{j\in[n]} \alpha_j t_j = 0, && \text{(1a)} \\
\forall j \in [n], \alpha_j \geqslant 0 && \text{(1b)}
\end{aligned}$$

We only update two coordinates, k and ℓ. From the condition (1a), one sees that given two indices k and ℓ, and denoting by α_k^{old} and α_ℓ^{old} the coefficients before the update, one has

$$\alpha_k t_k + \alpha_\ell t_\ell = \alpha_k^{\text{old}} t_k + \alpha_\ell^{\text{old}} t_\ell$$

From this, we can write

$$\begin{aligned}
\alpha_k &= \alpha_k^{\text{old}} + \alpha_\ell^{\text{old}} t_\ell t_k - \alpha_\ell t_\ell t_k \\
&= \gamma - s\alpha_\ell,
\end{aligned}$$

with $s := t_k t_\ell$ and $\gamma := \alpha_k^{\text{old}} + s\alpha_\ell^{\text{old}}$.

The second constraint (1b) allows to write

$$\alpha_k \geqslant 0 \implies \gamma - s\alpha_\ell \geqslant 0$$
$$\implies \begin{cases} \alpha_\ell \leqslant \alpha_k^{\text{old}} + \alpha_\ell^{\text{old}} & \text{if } s = 1, \\ \alpha_\ell \geqslant \max(0, \alpha_\ell^{\text{old}} - \alpha_k^{\text{old}}) & \text{if } s = -1. \end{cases}$$

Therefore, let

$$L = \begin{cases} 0 & \text{if } s = 1, \\ \max(0, \alpha_\ell^{\text{old}} - \alpha_k^{\text{old}}) & \text{if } s = -1. \end{cases}, \text{and } H = \begin{cases} \alpha_k^{\text{old}} + \alpha_\ell^{\text{old}} & \text{if } s = 1, \\ \infty & \text{if } s = -1. \end{cases} \tag{2}$$

be the lower and upper bounds for α_ℓ

Consider the function g in (1) as a function of α_k and α_ℓ only. With $\phi_i := \phi(x^{(i)})$, we have

$$g(\alpha_k, \alpha_\ell) = \alpha_k + \alpha_\ell - \frac{1}{2}\alpha_k^2\langle\phi_k, \phi_k\rangle - \frac{1}{2}\alpha_\ell^2\langle\phi_\ell, \phi_\ell\rangle - \alpha_k\alpha_\ell t_k t_\ell\langle\phi_k, \phi_\ell\rangle - \alpha_k t_k v_k - \alpha_\ell t_\ell v_\ell,$$

with $v_i := \sum_{j\in[n]\setminus\{k,\ell\}} \alpha_j t_j \langle\phi_i, \phi_j\rangle$.

Plugging-in $\alpha_k = \gamma - s\alpha_\ell$ gives

$$g(\alpha_\ell) = (\gamma - s\alpha_\ell) + \alpha_\ell - \frac{1}{2}(\gamma - s\alpha_\ell)^2\langle\phi_k, \phi_k\rangle - \frac{1}{2}\alpha_\ell^2\langle\phi_\ell, \phi_\ell\rangle - (\gamma - s\alpha_\ell)\alpha_\ell t_k t_\ell\langle\phi_k, \phi_\ell\rangle - (\gamma - s\alpha_\ell)t_k v_k - \alpha_\ell t_\ell v_\ell,$$

and solving $\frac{\partial}{\partial\alpha_\ell}g(\alpha_\ell^*) = 0$ gives

$$\alpha_\ell^* = \alpha_\ell^{\text{old}} + \frac{t_l(E_k - E_\ell)}{\langle\phi_k, \phi_k\rangle + \langle\phi_\ell, \phi_\ell\rangle - 2\langle\phi_k, \phi_l\rangle},$$

where $E_i := f(x^{(i)}) - t_i$.

Therefore, α_ℓ^* is the value for α_ℓ that maximizes $g(\alpha_k, \alpha_\ell)$, subject to the constraint (1a). It could be that α_ℓ^* does not satisfy the bounds (2); that's why we clip it and obtain

$$\alpha_\ell^{\text{new}} = \begin{cases} L & \text{if } \alpha_\ell^* < L, \\ \alpha_\ell^* & \text{if } \alpha_\ell^* \in [L, H], \\ H & \text{if } \alpha_\ell^* > H. \end{cases} \tag{3}$$

The pseudo-code of the SMO algorithm can be summarized as

```
1: initialization $\alpha = \mathbf{0}$
2: while $g(\alpha)$ not converged do
3:     choose a training sample $x_k$ that violates the KKT conditions
4:     choose a second training sample $x_\ell$ so that $|E_k - E_\ell|$ is maximized (heuristic)
5:     $\alpha_\ell \leftarrow \alpha_\ell^{\text{new}}$ from (3)
6:     $\alpha_k \leftarrow \alpha_k^{\text{old}} + t_k t_\ell(\alpha_\ell^{\text{old}} - \alpha_\ell^{\text{new}})$
   output $\alpha$
```

1. Implement the SMO algorithm.
2. Test it on some data.