CMU 10-715: Homework 3

Kernel methods

DUE: Oct. 3, 2020, 11:59 PM.

Instructions:

- **Collaboration policy:** Collaboration on solving the homework is allowed, after you have thought about the problems on your own. It is also OK to get clarification (but not solutions) from books, again after you have thought about the problems on your own. Please don't search for answers on the web, previous years' homeworks, etc. (please ask the TAs if you are not sure if you can use a particular reference). There are two requirements: first, cite your collaborators fully and completely (e.g., "Alice explained to me what is asked in Question 4.3"). Second, write your solution *independently*: close the book and all of your notes, and send collaborators out of the room, so that the solution comes from you only.

- **Submitting your work:** Assignments should be submitted as PDFs using Gradescope unless explicitly stated otherwise. Each derivation/proof should be completed on a separate page. Submissions can be handwritten, but should be labeled and clearly legible. Else, submission can be written in LaTeX.

- **Late days:** For each homework you get three late days to be used only when anything urgent comes up. No points will be deducted for using these late days. We will consider an honor system where we will rely on you to use the late days appropriately.

1 Validity of a kernel [30]

Prove that K_1 an K_2 are valid kernel functions.

(a) (15 points) $K_1(\boldsymbol{x_1}, \boldsymbol{x_2}) = exp(\frac{-||\boldsymbol{x_1}-\boldsymbol{x_2}||^2}{\sigma^2})$, where σ is a constant, $K_1 : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$.

(b) (15 points) $K_2(x_1, x_2) = \sigma^2 exp(cos(\frac{2\pi(x_1-x_2)}{p}))$, where σ, p are constants, $K_2 : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$.

You can assume the following statements hold without proving them:

1. Kernels are closed under summation, multiplication, combination of polynomials with non-negative coefficients, and exponentiation.
2. $K(\boldsymbol{x_1}, \boldsymbol{x_2}) = \boldsymbol{x_1}^T \boldsymbol{x_2}$ is a valid kernel function, where $K : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$.
3. $K(\boldsymbol{x_1}, \boldsymbol{x_2}) = f(\boldsymbol{x_1}) f(\boldsymbol{x_2})$ is a valid kernel function, where $f : \mathbb{R}^d \to \mathbb{R}$, $K : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$.

Proof:

(1). $K_1(\boldsymbol{x_1}, \boldsymbol{x_2}) = exp(\frac{-||\boldsymbol{x_1}-\boldsymbol{x_2}||^2}{\sigma^2}) = exp(\frac{-||\boldsymbol{x_1}||^2}{\sigma^2}) exp(\frac{-||\boldsymbol{x_2}||^2}{\sigma^2}) exp(\frac{2\boldsymbol{x_1}^T\boldsymbol{x_2}}{\sigma^2})$. $exp(\frac{2\boldsymbol{x_1}^T\boldsymbol{x_2}}{\sigma^2})$ is a kernel by point 2 and exponentiation. $exp(\frac{-||\boldsymbol{x_1}||^2}{\sigma^2}) exp(\frac{-||\boldsymbol{x_2}||^2}{\sigma^2})$ is a kernel by point 3.

(2) It suffices to show that $cos(2\pi(x_1 - x_2)/p)$ is a valid kernel. In fact, $cos(2\pi(x_1-x_2)/p) = cos(\frac{2\pi}{p}x_1)cos(\frac{2\pi}{p}x_2) + sin(\frac{2\pi}{p}x_1)sin(\frac{2\pi}{p}x_2) = f(x_1)f(x_2) + g(x_1)g(x_2)$. By point 3 and the addition rule, K_2 is a valid kernel function. They may also prove using the definition of kernels.

2 Kernelized soft SVM [40]

Given a training set $S = \{(\boldsymbol{x_1}, y_1), \cdots, (\boldsymbol{x_n}, y_n)\}$ where $\boldsymbol{x_i} \in \mathbb{R}^d, y_i \in \{-1, 1\}$. Recall that the soft SVM can be rewritten as follows:

$$\underset{\mathbf{w}}{\text{minimize}} \;\; \frac{1}{2}||\mathbf{w}||_2^2 + C\sum_{i=1}^{n} \max\left(0, 1 - y_i(\mathbf{w}^T\mathbf{x}_i)\right) \tag{1}$$

(a) (10 points) Given a kernel function $K(\boldsymbol{x_1}, \boldsymbol{x_2}) = \langle\psi(\boldsymbol{x_1}), \psi(\boldsymbol{x_2})\rangle$. Rewrite equation (1) to solve for the kernelized soft SVM problem.

(b) (30 points) Write down the stochastic gradient descent algorithm (with mini-batch) for solving the problem in (a), assuming the batch size is b. Use the kernel trick instead of directly applying soft SVM to the transformed data, $\psi(\boldsymbol{x_i})$. The algorithm should include the initialization step, the update step, and the output. Only the pseudo code of the algorithm is needed. No coding is required.

Answer:

(a) The problem in the feature space is:

$$\underset{\mathbf{w}}{\text{minimize}} \;\; \frac{1}{2}||\mathbf{w}||_2^2 + C\sum_{i=1}^{n} \max\left(0, 1 - y_i\langle\mathbf{w}, \psi(\mathbf{x}_i)\rangle\right) \tag{2}$$

which can be written as:

$$\underset{\alpha\in\mathbb{R}^d}{\text{minimize}} \;\; \sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j K(\mathbf{x}_i, \mathbf{x}_j) + C\sum_{i=1}^{n}\max\left(0, 1 - y_i\sum_{j=1}^{n}\alpha_j K(\mathbf{x}_i, \mathbf{x}_j)\right) \tag{3}$$

(b): The subgradient of the objective in 3 on a batch of indices $\mathbf{b}$ is:

$$\frac{\partial L}{\partial \alpha_i} = \sum_{j=1}^{n}\alpha_j K(\mathbf{x}_i, \mathbf{x}_j) + C\sum_{j\in\mathbf{b}}(-y_j K(\mathbf{x}_i, \mathbf{x}_j))\mathbb{1}\{(1 - y_j(\sum_{k=1}^{n}\alpha_k K(\mathbf{x}_j, \mathbf{x}_k))) > 0\} \tag{4}$$

The algorithm is similar to that of the SGD for soft SVM:

Algorithm 1: SGD with mini-batch for kernel soft SVM

```
Input: Number of steps $T$, learning rate $\lambda$ and batch size $b$
Initialize parameters $\alpha = \mathbf{0}$, step $t = 0$;
for $t = 1, \dots, T$ do
    Sample batch indices $\mathbf{b}$ with $|\text{batch}| = b$;
    Compute the subgradient $\nabla\hat{L}$ on the batch using equation 4;
    Update the parameters $\alpha = \alpha - \lambda\nabla\hat{L}$ ;
end
Ouput: $\mathbf{w} = \sum_{i=1}^{n}\alpha_i\psi(\mathbf{x}_i)$
```

3 Kernels and linear separability [30]

(a) (15 points) Show an example of a separator $g : \mathbb{R} \to \{-1, 1\}$ such that it is not perfectly separable even with infinite data using an identity kernel and a linear separator, but is separable if using either K_1 or K_2 from Q1 and a linear separator. (You may assume that the domain of g is an interval in $\mathbb{R}$.)g

(b) (15 points) Show an example of a separator $g : \mathbb{R} \to \{-1, 1\}$ such that it is not perfectly separable even with infinite data using K_1 and a linear separator, but is separable if using K_2 and a linear separator.

Note: You can either write it down algebraically or show a hand-drawn plot of data representing the separator.

Examples:

For a kernel K to linearly separate g, there exists real numbers w, b such that $sign(K(w, x) + b) = g$.

(a). $g(x) = 1$ if $x \in [-a/2, a/2]$, $g(x) = -1$ if $x \in [-a, -a/2) \cup (a/2, a]$ for $a > 0$. An identity kernel cannot linearly separate g because its decision boundary is a threshold. Both K_1 and K_2 can linearly separate g because, by selecting appropriate constants for each kernel and setting $w = 0$, $K(w, x) + b$ can be monotonic with $|x|$ in $[-a, a]$. Selecting the appropriate threshold, b, will ensure that $sign(K(w, x) + b) = g$.

(b). $g(x) = 1$ if $cos(x) > 0$, $g(x) = -1$ otherwise. The identity kernel doesn't linearly separate g for the same reason as in (a). As for K_1, since $K(w, x) + b$ can be monotonic with $|x - w|$, no matter how b is chosen, when $|x - w|$ is large enough the prediction will be $+1$. This won't be an issue for K_2 because the *cosine* in the kernel guarantees $K(w, x) + b$ can be periodic, and with properly chosen constants, $sign(K(w, x) + b) = g$.