

Compsci 571 HW4

Yilin Gao (yg95)

February 27, 2018

1 Constructing Kernels

Let K_1 be kernels over $\mathbb{R}^n \times \mathbb{R}^n$, let $a \in \mathbb{R}$, $\langle a, b \rangle$ denotes the dot product, $a^T b$.

(a) $K(x, z) = aK_1(x, z)$

This is not a kernel. Because K_1 is a valid kernel, $K_1(x, z) \geq 0$. When $a < 0$, $K(x, z) = aK_1(x, z) \leq 0$. Because inner products $\langle \cdot, \cdot \rangle \geq 0$, there won't be an inner product such that $K(x, z) = \langle \Phi(x), \Phi(z) \rangle_{H_k}$.

(b) $K(x, z) = \langle x, z \rangle^3 + (\langle x, z \rangle - 1)^2$

This is not a valid kernel.

From lecture notes we've proved $\langle x, z \rangle^3$ can be expressed as inner product in a new vector space, thus a valid kernel.

For $(\langle x, z \rangle - 1)^2$:

$$(\langle x, z \rangle - 1)^2 = (x^T z - 1)^2 = (\sum_{i=1}^n x^{(i)} z^{(i)} - 1)(\sum_{j=1}^n x^{(j)} z^{(j)} - 1) = \sum_{i=1}^n \sum_{j=1}^n x^{(i)} x^{(j)} z^{(i)} z^{(j)} - 2 \sum_{i=1}^n x^{(i)} z^{(i)} + 1 = \sum_{i=1}^n \sum_{j=1}^n x^{(i)} x^{(j)} z^{(i)} z^{(j)} - \sum_{i=1}^n (\sqrt{2} x^{(i)}) (\sqrt{2} z^{(i)}) + 1$$

This equation cannot be expressed as an inner product of some $\Phi(x)$ and $\Phi(z)$. So $(\langle x, z \rangle - 1)^2$ is not a valid kernel.

From lecture notes we know that when $k(x, z) = k_1(x, z) + k_2(x, z)$, only if both $k_1(x, z)$ and $k_2(x, z)$ are valid kernels, will $k(x, z)$ be a valid kernel. So in this case $K(x, z)$ is not a kernel.

(c) $K(x, z) = \langle x, z \rangle^2 + \exp(-\|x\|^2) \exp(-\|z\|^2)$

This is a valid kernel.

From lecture notes we've proved $\langle x, z \rangle^2$ can be expressed as inner product in a new vector space, thus a valid kernel.

From lecture notes we've proved $k(x, z) = g(x)g(z)$, $g : \mathbb{R}^n \rightarrow \mathbb{R}$ is a valid kernel, so $\exp(-\|x\|^2) \exp(-\|z\|^2)$, in the same form, is a valid kernel.

So $K(x, z) = \langle x, z \rangle^2 + \exp(-\|x\|^2) \exp(-\|z\|^2)$, a sum of two valid kernels, is a valid kernel.

2 Reproducing Kernel Hilbert Spaces

For \mathcal{F} to be the RKHS for kernel $K(x, y) = xy$, it should satisfy:

1. $K(x, y)$ spans \mathcal{F} , i.e., $\mathcal{F} = \text{span}\{K(\cdot, x), x \in [0, 1]\} = \{f : f(\cdot) = \sum_{i=1}^m \alpha_i K(\cdot, x_i)\}$

2. $K(x, y)$ has the reproducing kernel property: $f(x) = \langle f(\cdot), K(\cdot, x) \rangle_{\mathcal{F}_K}$

If \mathcal{F} satisfies condition 1, for any function $f(x) = ax$ in \mathcal{F} , $f(x) = \sum_{i=1}^m \alpha_i K(x, x_i) = \sum_{i=1}^m \alpha_i x x_i = (\sum_{i=1}^m \alpha_i x_i) x$, and this means the real number $a = \sum_{i=1}^m \alpha_i x_i$ for the specific m and α_i .

Under this condition, for any $f(\cdot) = a \cdot = (\sum_{i=1}^m \alpha_i x_i) \cdot$, $\langle f(\cdot), K(\cdot, x) \rangle_{\mathcal{F}_K} = \langle \sum_{i=1}^m \alpha_i K(\cdot, x_i), K(\cdot, x) \rangle_{\mathcal{F}_K} = \sum_{i=1}^m \alpha_i K(x, x_i) = \sum_{i=1}^m \alpha_i x x_i = (\sum_{i=1}^m \alpha_i x_i) x = ax = f(x)$. So condition 2 is also satisfied.

So \mathcal{F} is the RKHS for kernel $K(x, y) = xy$. \square

3 Convexity and KKT Conditions

(a) The Lagrangian function for the primal form is:

$$\begin{aligned} \min L(\mathbf{w}, \eta, \eta^*, \mathbf{a}, \mathbf{b}, \mathbf{c}, \mathbf{d}) = & \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i=1}^n (\eta_i + \eta_i^*) + \sum_{i=1}^n a_i [y_i - \mathbf{w}^T \mathbf{x}_i - \epsilon - \eta_i] \\ & + \sum_{i=1}^n b_i [\mathbf{w}^T \mathbf{x}_i - y_i - \epsilon - \eta_i^*] - \sum_{i=1}^n c_i \eta_i - \sum_{i=1}^n d_i \eta_i^* \end{aligned}$$

It's KKT conditions are:

- Primal feasibility:

$$\begin{aligned} y_i - \mathbf{w}^T \mathbf{x}_i - \epsilon - \eta_i &\leq 0, i = 1, \dots, n \\ \mathbf{w}^T \mathbf{x}_i - y_i - \epsilon - \eta_i^* &\leq 0, i = 1, \dots, n \\ \eta_i &\geq 0, i = 1, \dots, n \\ \eta_i^* &\geq 0, i = 1, \dots, n \end{aligned}$$

- Dual feasibility:

$$\begin{aligned} a_i &\geq 0, i = 1, \dots, n \\ b_i &\geq 0, i = 1, \dots, n \\ c_i &\geq 0, i = 1, \dots, n \\ d_i &\geq 0, i = 1, \dots, n \end{aligned}$$

- Complementary slackness:

$$\begin{aligned} a_i [y_i - \mathbf{w}^T \mathbf{x}_i - \epsilon - \eta_i] &= 0, i = 1, \dots, n \\ b_i [\mathbf{w}^T \mathbf{x}_i - y_i - \epsilon - \eta_i^*] &= 0, i = 1, \dots, n \\ c_i \eta_i &= 0, i = 1, \dots, n \\ d_i \eta_i^* &= 0, i = 1, \dots, n \end{aligned}$$

- Lagrangian stationary:

$$\begin{aligned} \nabla_{\mathbf{w}} L &= \mathbf{w} - \sum_{i=1}^n (a_i - b_i) \mathbf{x}_i = 0 \\ (\nabla_{\eta} L)_i &= C - a_i - c_i = 0, i = 1, \dots, n \\ (\nabla_{\eta^*} L)_i &= C - b_i - d_i = 0, i = 1, \dots, n \end{aligned}$$

With these conditions, we can transform Lagrangian function into dual form:

$$\max L(\mathbf{a}, \mathbf{b}) = \sum_{i=1}^n (a_i - b_i) y_i - \epsilon \sum_{i=1}^n (a_i + b_i) - \frac{1}{2} \sum_{i,j=1}^n (a_i - b_i)(a_j - b_j) \mathbf{x}_i^T \mathbf{x}_j$$

subject to

$$0 \leq a_i, b_i \leq C, i = 1, \dots, n$$

(b) Support vectors are the points i such that $|y_i - \mathbf{w}^T \mathbf{x}_i| \leq \epsilon$.

- (c) Increasing ϵ makes the model less likely to overfit. Because the model penalizes the points that have training error larger than ϵ . If ϵ increases, the allowed/unpenalized training error increases, and the model tends to overfit less.
- (d) Increasing C makes the model more likely to overfit. C is the penalty for each point that has training error larger than ϵ . If the penalty increases, the model will try to make points have smaller training error, and thus overfits.
- (e) Assume we've computed the optimal dual variables as \mathbf{a}^* and \mathbf{b}^* .

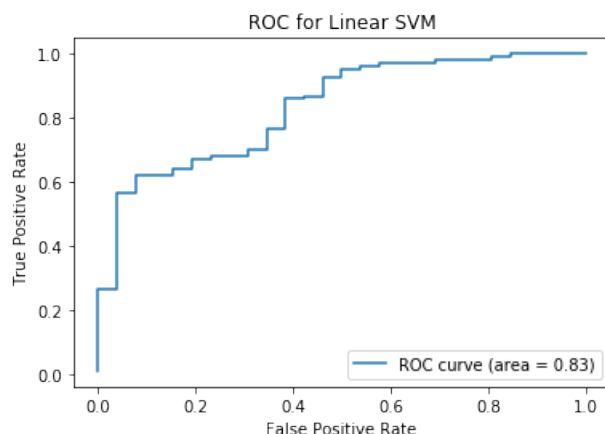
From one of the KKT conditions, we can get the optimal primal variable is $\mathbf{w}^* = \sum_{i=1}^n (a_i^* - b_i^*) \mathbf{x}_i$.

So for a new point \mathbf{x}^{new} , its evaluation is $f(\mathbf{x}^{new}) = \sum_{j=1}^p w_j^* x_j^{new} = \sum_{j=1}^p \sum_{i=1}^n (a_i^* - b_i^*) x_{ij} x_j^{new} = \sum_{i=1}^n (a_i^* - b_i^*) \mathbf{x}_i \cdot \mathbf{x}^{new}$.

4 SVM Implementation

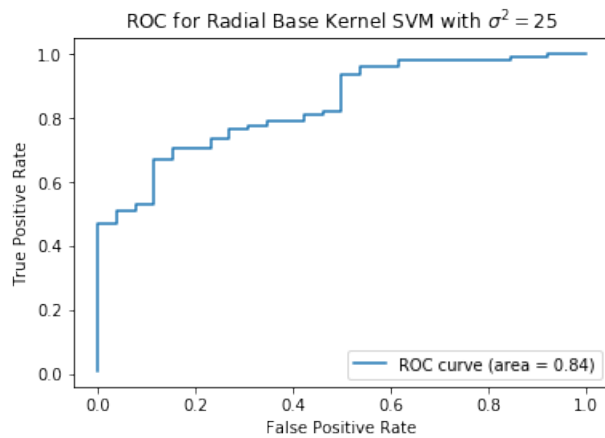
- (a) See `svm_classifier.py`.
- (b) **Note: for questions (b) and (c), I use `sklearn.model_selection.train_test_split` to split the training and testing set with 2018 as the random seed. And if I use numpy with 2018 as the random seed to generate indices for training and testing data and then split, the split is different.** Code for these 2 questions are in `q4.ipynb`.

The accuracy of the classifier on testing data is 0.8636363636363636. The ROC curve is like following:



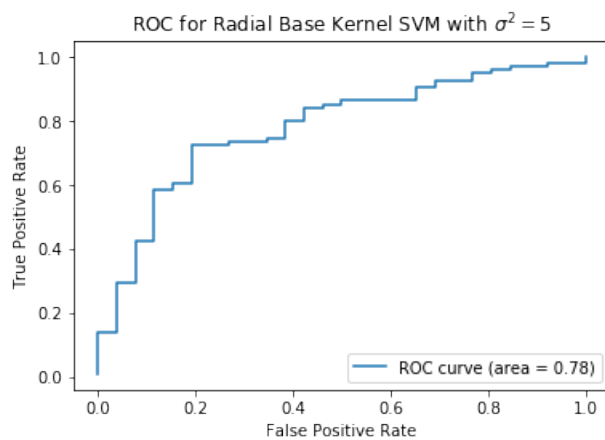
The AUC on testing data is 0.8316400580551523.

- (c) For $\sigma^2 = 25$, the accuracy of the classifier on testing data is 0.8484848484848485. The ROC curve is like:



The AUC on testing data is 0.8388969521044993.

For $\sigma^2 = 5$, the accuracy for the classifier on testing data is 0.7954545454545454. The ROC curve is like:



The AUC on testing data is 0.7790275761973875.

The comparison between 2 σ^2 values suggests that for Gaussian kernel if we set σ^2 too small we may overfit.