Machine Learning **Chao Lan**
COSC4555/5555 Instructor
MWF 9-9:50am, EN 2105 clan@uwyo.edu

1 Kernel Methods

1.1 Other Kernel Methods

We will briefly introduce kernelized linear SVM [1], PCA and k-means. We will revisit them in the mapped feature space, with feature mapping function ϕ and its induced kernel function κ.

Kernelized Linear SVM

Consider hard-margin linear SVM. Recall its optimal β is a linear combination of training instances. When we plug this fact back (together with other constraints), we obtain a wolf dual optimization problem:

$$\begin{aligned} &\min_{\alpha} \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j \phi(x_i)^T \phi(x_j) \\ &\text{s.t. } \sum_{i=1}^{n} \alpha_i y_i = 0, \text{ and } \alpha_i \geq 0, \forall i. \end{aligned} \tag{1}$$

Note training instances are involved only in form of inner product. Thus we can apply kernel trick and rewrite above problem in the mapped feature space as

$$\begin{aligned} &\min_{\alpha} \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j \kappa(x_i, x_j) \\ &\text{s.t. } \sum_{i=1}^{n} \alpha_i y_i = 0, \text{ and } \alpha_i \geq 0, \forall i. \end{aligned} \tag{2}$$

Above is the hard-margin SVM.

Kernelized PCA

Assume sample mean is zero, thus data covariance is

$$\Sigma = \frac{1}{n} \sum_{i=1}^{n} \phi(x_i)\phi(x_i)^T. \tag{3}$$

We know the optimal PCA projection vector w satisfies

$$\begin{aligned} \Sigma w = \lambda w \Rightarrow \frac{1}{n} \sum_{i=1}^{n} \phi(x_i)\phi(x_i)^T w = \lambda w \Rightarrow &\sum_{i=1}^{n} \phi(x_i) \left(\frac{1}{\lambda n} \phi(x_i)^T w \right) = w \\ \Rightarrow &\sum_{i=1}^{n} \alpha_i \phi(x_i) = w, \end{aligned} \tag{4}$$

where $\alpha_i = \frac{1}{\lambda n} \phi(x_i)^T w$.

This shows optimal w is a linear combination of training instances.

[1] Kernelized Linear SVM is the commonly known SVM.

Plugging this back to the objective function of PCA, we have

$$w^T \Sigma w = \frac{1}{n} \sum_{i,k,j=1}^{n} \alpha_i \alpha_j \kappa(x_i, x_k) \kappa(x_k, x_j) = \frac{1}{n} \alpha^T K K \alpha, \tag{5}$$

where $\alpha = [\alpha_1, \ldots, \alpha_n]$ and $K \in \mathcal{R}^{n \times n}$ is the Gram matrix with $K_{i,j} = \kappa(x_i, x_j)$.
[Exercise] Verify (5).
[Exercise] Verify the constraint $w^T w = \alpha^T K \alpha = 1$.
Thus the kernel PCA method solves the following problem

$$\max_{\alpha} \frac{1}{n} \alpha^T K K \alpha \quad \text{s.t. } \alpha^T K \alpha = 1. \tag{6}$$

Kernelized K-means

Since there is no model to learn in K-means, we don't need to show optimal solution is a linear combination of training instances. Consider the two alternate steps in K-means:

- update cluster center

$$\mu_k = \sum_{i \in I_k} \phi(x_i). \tag{7}$$

- update cluster assignment based on

$$\begin{aligned} & ||\phi(x_i) - \mu_k||^2 \\ & = \phi(x_i)^T \phi(x_i) - \phi(x_i)^T \mu_k + \mu_k^T \mu_k \\ & = \phi(x_i)^T \phi(x_i) - \sum_{j \in I_k} \phi(x_i)^T \phi(x_j) + \sum_{i,j \in I_k} \phi(x_i)^T \phi(x_j) \\ & = \kappa(x_i, x_i) - \sum_{j \in I_k} \kappa(x_i, x_j) + \sum_{i,j \in I_k} \kappa(x_i, x_j). \end{aligned} \tag{8}$$

We see instances are involved in the assignment update only in form of inner product. (This is not true in the cluster center update; but that update is not required in implementation.)

Representer Theorem

We have shown several algorithms have their optimal solutions expressed as linear combination of (training) instances. In fact, this holds generally based on the Representer Theorem.
Consider the following optimization problem

$$\min_f D[f(x_1), \ldots, f(x_n)] + P(||f||_{\mathcal{F}}^2), \tag{9}$$

where D is a function depending on x_i only through $f(x_i)$, P is a non-decreasing function and $\mathcal{F}$ is a Reproducing Kernel Hilbert Space associated with some kernel κ, and $f \in \mathcal{F}$.
The Representer theorem states if one minimizer of (9) (if exists) has the form

$$f = \sum_{i=1}^{n} \alpha_i \kappa(\cdot, x_i). \tag{10}$$

And if P is strictly increasing, then every minimizer has the above form.

[Reading] ELS, Chapter 12.3 & PRML, Chapter 12.3.

1.2 Gaussian Process

Gaussian Process is a machine learning model that introduces kernel methods to probabilistic predictive model. It can be applied to both regression and classification tasks. We will introduce it in the context of regression task.

Let $(\phi(x)_1, t_1) \ldots, (\phi(x)_n, t_n)$ be a set of labeled training instances. Assume they are generated from a linear regression model

$$y_i = \phi(x)_i^T \beta \tag{11}$$

with a data-independent additive noise ϵ so that

$$t_i = y_i + \epsilon. \tag{12}$$

Let $\mathbf{t_n} = [t_1, \ldots, t_n]$ be the collection of training labels.
Let $\phi(x)_{n+1}$ be a testing instance. We want to predict its label t_{n+1} based on probability

$$p(t_{n+1} \mid \mathbf{t_n}); \tag{13}$$

for instance, we may calculate its predicted label by

$$\hat{t}_{n+1} = E[t_{n+1} \mid \mathbf{t_n}] = \int t_{n+1} \; p(t_{n+1} \mid \mathbf{t_n}) \; dt_{n+1}. \tag{14}$$

Let $\mathbf{t_{n+1}} = (\mathbf{t_n}, t_{n+1})$. Since

$$p(t_{n+1} \mid \mathbf{t_n}) = \frac{p(\mathbf{t_{n+1}})}{p(\mathbf{t_n})}, \tag{15}$$

all we need is to figure out the marginal probability $p(\mathbf{t_n})$ with generic n.

Now, you may wonder where does randomness in the above probabilities come from.
It will come from two sources:

- we will assume β is random and follows a normal distribution $N(0, \alpha^{-1} I)$
- we will assume ϵ is random and follows a normal distribution $N(0, \beta^{-1})$

We will first characterize distribution of $\{y_i\}$. Let $\mathbf{y_n} = [y_1, \ldots, y_n]^T$ and $X_\phi = [\phi(x_1), \ldots, \phi(x_n)]^T$. We can write the regression model over training set as

$$\mathbf{y_n} = X_\phi \beta. \tag{16}$$

Since $\mathbf{y_n}$ is a linear combination of normally distributed elements in β, it also follows a normal distribution. We only need the mean and covariance. From the first assumption, we have

$$E[\mathbf{y_n}] = 0 \tag{17}$$

and

$$\begin{aligned} cov[\mathbf{y_n}] &= E[(\mathbf{y_n} - E[\mathbf{y_n}])(\mathbf{y_n} - E[\mathbf{y_n}])^T] \\ &= E[\mathbf{y_n}\mathbf{y_n}^T] \\ &= E[(X_\phi \beta)(X_\phi \beta)^T] \\ &= X_\phi E[\beta\beta^T] X_\phi^T \\ &= X_\phi E[(\beta - 0)(\beta - 0)^T] X_\phi^T \\ &= X_\phi cov[\beta] X_\phi \\ &= \frac{1}{\alpha} X_\phi X_\phi^T = K, \end{aligned} \tag{18}$$

where $K_\alpha \in \mathcal{R}^{n\times n}$ is a Gram matrix with element $K_{i,j} = \frac{1}{\alpha}k(x_i, x_j)$.
Therefore we have

$$\mathbf{y_n} \sim N(0, K). \tag{19}$$

We now characterize conditional distribution of $\{t_i\}$. Recall $\mathbf{t_n} = [t_1, \ldots, t_n]^T$ and ϵ is data independent. From the second assumption we have

$$\mathbf{t_n} \mid \mathbf{y_n} \sim N(\mathbf{y_n}, \beta^{-1}I). \tag{20}$$

We finally characterize marginal distribution of $\mathbf{t_n}$. This will involve some properties of probability which we will not prove. See [PRML, Chapter 6.4] for detailed discussions.
Putting (19) and (20) together, we have

$$p(\mathbf{t_n}) = \int p(\mathbf{t_n} \mid \mathbf{y_n}) \cdot p(\mathbf{y_n})\, d\mathbf{t_n} = N(0, C_n), \tag{21}$$

where $C_n \in \mathcal{R}^{n\times n}$ is covariance matrix with

$$C_{i,j|n} = k(x_i, x_j) + \beta^{-1}\delta_{i,j}. \tag{22}$$

We also have

$$p(\mathbf{t_{n+1}}) = \int p(\mathbf{t_{n+1}} \mid \mathbf{y_{n+1}}) \cdot p(\mathbf{y_{n+1}})\, d\mathbf{y_{n+1}} = N(0, C_{n+1}), \tag{23}$$

where all variables are based on $n+1$ instances, including n training and one testing.
Based on above results, one can verify $p(t_{n+1} \mid \mathbf{t_n}) \sim N(\mu, \Sigma)$, with mean

$$\mu = k^T C_n^{-1} \mathbf{t_n} \tag{24}$$

and covariance matrix

$$\Sigma = c - k^T C_n^{-1} k, \tag{25}$$

where

$$k = [k(x_1, x_{n+1}), k(x_2, x_{n+1}), \ldots, k(x_n, x_{n+1})] \tag{26}$$

and

$$c = k(x_{n+1}, x_{n+1}) + \beta^{-1}. \tag{27}$$

We have introduced Gaussian process for a probabilistic linear regression model. For probabilistic classification model, analysis is similar but one cannot find closed-form solution as because the conditional integral of $p(t_{n+1} \mid \mathbf{t_n})$ is analytically intractable. We can consider finding approximate solutions using e.g. variational inference, expectation propagation or Laplace approximation. See [PRML Chapter 6.4.6] for detailed introduction to the third method.

[Reading] PRML, Chapter 6.4 (6.4.1, 6.4.2)