MML lecture extra notes, Nov 26, 2021

The contents in this note is non-examinable, but you might find it useful as preparation for the "Probabilistic Inference" course in Spring.

Full derivations for kernel PCA

In non-linear PCA, we wish to find orthonormal projection directions $\{\mathbf{b}_1, ..., \mathbf{b}_M\}$ for the non-linear features $\{\phi(\boldsymbol{x}_1), ..., \phi(\boldsymbol{x}_N)\}$, $\phi(\boldsymbol{x}) \in \mathbb{R}^{p\times 1}$ such that $\sum_{m=1}^{M} \mathbb{V}[\phi(\boldsymbol{x}_n)^\top \mathbf{b}_m]$ is maximised. Here we assume that the dataset $\mathbf{X} = \{\boldsymbol{x}_1, ..., \boldsymbol{x}_N\}$ is pre-processed such that $\mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}[\phi(\boldsymbol{x})] = \mathbf{0}$. From previous discussions of PCA we see that the optimal solution of $\{\mathbf{b}_1, ..., \mathbf{b}_m\}$ is the leading M eigenvectors of matrix $\mathbf{S} = \frac{1}{N}\sum_{n=1}^{n} \phi(\boldsymbol{x}_n)\phi(\boldsymbol{x}_n)^\top \in \mathbb{R}^{p\times p}$.

Now consider eigendecomposition of $\mathbf{S}$, which requires solving for $\mathbf{b}_m$ and λ_m the following equation:

$$\frac{1}{N}\sum_{n=1}^{n} \phi(\boldsymbol{x}_n)\phi(\boldsymbol{x}_n)^\top \mathbf{b}_m = \lambda_m \mathbf{b}_m. \tag{1}$$

Notice that both λ_m and $\phi(\boldsymbol{x}_n)^\top \mathbf{b}_m$ are scalars. So the optimal solution for $\mathbf{b}_m$ has the following form

$$\mathbf{b}_m := \sum_{n=1}^{N} \alpha_{mn}\phi(\boldsymbol{x}_n) \tag{2}$$

for a particular set of $\{\alpha_{mn}\}$ parameters. In the eigendecomposition problem (1) it means $\alpha_{mn} = \frac{\phi(\boldsymbol{x}_n)^\top \mathbf{b}_m}{N\lambda_m}$. But instead of solving for λ_m directly (as an eigenvalue of $\mathbf{S}$) which typically has $\mathcal{O}(p^3)$ computational cost, we can directly use the alternative parameterisation (2) and compute the objective for $\mathbf{b}_m$ as follows:

$$\begin{aligned}
\mathbf{b}_m^\top \mathbf{S}\mathbf{b}_m &= \frac{1}{N}\sum_{n=1}^{n} \mathbf{b}_m^\top \phi(\boldsymbol{x}_n)\phi(\boldsymbol{x}_n)^\top \mathbf{b}_m \\
&= \frac{1}{N}\sum_{n=1}^{n} \left(\phi(\boldsymbol{x}_n)^\top \sum_{n'=1}^{N} \alpha_{mn'}\phi(\boldsymbol{x}_{n'})\right)^2 \\
&= \frac{1}{N}\sum_{n=1}^{n} \left(\sum_{n'=1}^{N} \alpha_{mn'}\mathcal{K}(\boldsymbol{x}_n, \boldsymbol{x}_{n'})\right)^2.
\end{aligned} \tag{3}$$

Now define $\boldsymbol{\alpha}_m = [\alpha_{m1}, ..., \alpha_{mN}]^\top$, and define the kernel matrix as $\mathbf{K}_{\mathbf{XX}} \in \mathbb{R}^{N\times N}$ with entries $\mathbf{K}_{\mathbf{XX}}[i,j] = \mathcal{K}(\boldsymbol{x}_i, \boldsymbol{x}_j)$. This allows us to further rewrite the objective as

$$\mathbf{b}_m^\top \mathbf{S}\mathbf{b}_m = \frac{1}{N}\sum_{n=1}^{n} (\mathbf{K}_{\mathbf{XX}}[n,\cdot]\boldsymbol{\alpha}_m)^2 = \frac{1}{N}||\mathbf{K}_{\mathbf{XX}}\boldsymbol{\alpha}_m||_2^2. \tag{4}$$

For the constraints on $\mathbf{b}_m$, they also translates to the constraints on $\boldsymbol{\alpha}_m$:

$$\begin{aligned}
||\mathbf{b}_m||_2^2 = 1 \quad &\Rightarrow \quad \sum_{n,n'} \alpha_{mn}\alpha_{mn'}\mathcal{K}(\boldsymbol{x}_n, \boldsymbol{x}_{n'}) = \boldsymbol{\alpha}_m^\top \mathbf{K}_{\mathbf{XX}}\boldsymbol{\alpha}_m = 1, \\
\mathbf{b}_i \perp \mathbf{b}_j, \forall i \neq j \quad &\Rightarrow \quad \sum_{n,n'} \alpha_{in}\alpha_{jn'}\mathcal{K}(\boldsymbol{x}_n, \boldsymbol{x}_{n'}) = \boldsymbol{\alpha}_i^\top \mathbf{K}_{\mathbf{XX}}\boldsymbol{\alpha}_j = 0.
\end{aligned} \tag{5}$$

Readers are encouraged to solve this problem themselves using the Lagrange multiplier method. Such method returns solutions $\{\boldsymbol{\alpha}_1, ..., \boldsymbol{\alpha}_M\}$ as the leading M eigenvectors of matrix $\frac{1}{N}\mathbf{K}_{\mathbf{XX}} \in \mathbb{R}^{N\times N}$. In prediction, given a new datapoint $\hat{\boldsymbol{x}}$, dimension reduction can be done by

$$\mathbf{b}_m^\top \phi(\hat{\boldsymbol{x}}) = \sum_{n=1}^{N} \alpha_{mn}\mathcal{K}(\hat{\boldsymbol{x}}, \boldsymbol{x}_n) = \mathbf{K}_{\hat{\boldsymbol{x}}\mathbf{X}}\boldsymbol{\alpha}_m. \tag{6}$$

Compare with performing eigendecomposition for $\mathbf{S}$ directly, eigendecomposition of $\mathbf{K}_{\mathbf{XX}}$ typically has cost $\mathcal{O}(N^3)$, resulting in significant speed-up if $N << p$ (or even when $p \Rightarrow +\infty$). The kernel PCA can also be extended to settings where $\phi(\boldsymbol{x})$ lies in a non-Euclidean space, see the next section for further discussions.

From Bayesian linear regression to Gaussian processes

Recall the parametric non-linear regression model:

$$p(y|\boldsymbol{x},\boldsymbol{\theta}) = \mathcal{N}(y;\phi(\boldsymbol{x})^\top\boldsymbol{\theta},\sigma^2), \quad p(\boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{\theta};\mathbf{0},\lambda^{-1}\mathbf{I})$$

Now we derive a Gaussian process model from the above Bayesian linear regression model by lifting the prior from parameter space to function space. Recall that $f(\boldsymbol{x}) = \phi(\boldsymbol{x})^\top\boldsymbol{\theta}$. This means for any $N > 0$ and $\mathbf{X} = [\boldsymbol{x}_1, ..., \boldsymbol{x}_N]^\top \in \mathbb{R}^{N\times d}$, we have

$$\mathbf{f} = [f(\boldsymbol{x}_1), ..., f(\boldsymbol{x}_N)]^\top = \Phi\boldsymbol{\theta} \in \mathbb{R}^{N\times 1}, \quad \Phi = [\phi(\boldsymbol{x}_1), ...\phi(\boldsymbol{x}_N)]^\top.$$

Also notice that for a Gaussian variable $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0},\Sigma)$, we have $\mathbf{A}\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0},\mathbf{A}\Sigma\mathbf{A}^\top)$. This means we can lift the prior on $\boldsymbol{\theta}$ to a prior on $\mathbf{f}$:

$$p(\boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{\theta};\mathbf{0},\lambda^{-1}\mathbf{I}) \quad \Rightarrow \quad p(\mathbf{f}|\mathbf{X}) = \mathcal{N}(\mathbf{f};\mathbf{0},\mathbf{K}_{\mathbf{XX}}),\ \mathbf{K}_{\mathbf{XX}} = \lambda^{-1}\Phi\Phi^\top.$$

Importantly, this lifting procedure $\mathbf{f} = \Phi\boldsymbol{\theta}$ makes $f(\boldsymbol{x}_i)$ and $f(\boldsymbol{x}_j)$ correlated. Since we can perform such lifting for any $N > 0$, this results in a Gaussian process (GP) prior written as follows:

$$p(\boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{\theta};\mathbf{0},\lambda^{-1}\mathbf{I}) \quad \Rightarrow \quad f(\cdot) \sim \mathcal{GP}(0,\mathcal{K}(\cdot,\cdot)),\ \mathcal{K}(\boldsymbol{x}_i,\boldsymbol{x}_j) = \lambda^{-1}\phi(\boldsymbol{x}_i)^\top\phi(\boldsymbol{x}_j).$$

For the likelihood terms we can perform similar lifting as well:

$$\prod_{n=1}^{N} p(y_n|\boldsymbol{x}_n,\boldsymbol{\theta}) = \prod_{n=1}^{N} \mathcal{N}(y_n;\phi(\boldsymbol{x}_n)^\top\boldsymbol{\theta},\sigma^2) = \mathcal{N}(\mathbf{y};\Phi\boldsymbol{\theta},\sigma^2\mathbf{I}) \quad \Rightarrow \quad p(\mathbf{y}|\mathbf{f}) = \mathcal{N}(\mathbf{y};\mathbf{f},\sigma^2\mathbf{I}).$$

Combining both results returns a GP regression model as

$$p(y|f,\boldsymbol{x}) = \mathcal{N}(y;f(\boldsymbol{x}),\sigma^2), \quad f(\cdot) \sim \mathcal{GP}(0,\mathcal{K}(\cdot,\cdot)).$$

In general $\phi(\boldsymbol{x}_i)$ might be infinite-dimensional (or even in non-Euclidean space), so one would rather directly define the kernel function $\mathcal{K}(\cdot,\cdot)$ instead of specifying the form of $\phi(\cdot)$. Specifically, if $\mathcal{K}(\cdot,\cdot)$ is a *reproducing kernel* of a *reproducing kernel Hilbert space* (RKHS) $\mathcal{H}$ equipped with inner product $\langle\cdot,\cdot\rangle_{\mathcal{H}}$, then one can use $\phi(\boldsymbol{x}) = \mathcal{K}(\boldsymbol{x},\cdot)$ as the non-linear feature (which is a function so is infinite-dimensional) and compute the inner product as $\langle\phi(\boldsymbol{x}_i),\phi(\boldsymbol{x}_j)\rangle_{\mathcal{H}} = \mathcal{K}(\boldsymbol{x}_i,\boldsymbol{x}_j)$.