CSCI 5525 Machine Learning Fall 2019

Lecture 19: Kernel PCA and MLE

Nov 7th 2019

Lecturer: Steven Wu *Scribe: Steven Wu*

Principal Component Analysis

Principal component analysis aims to solve the following optimization problem: given as input a data matrix $X \in \mathbb{R}^{n\times d}$, find an encoder E and decoder D to minimize the following reconstruction error:

$$\min_{D\in\mathbb{R}^{k\times d}, E\in\mathbb{R}^{d\times k}} \|X - XED\|_F^2 \tag{1}$$

where $\|\cdot\|_F$ denotes the Frobenius norm: for any matrix A,

$$\|A\|_F = \sqrt{\sum_{(i,j)} A_{ij}^2} = \sqrt{\text{tr}(A^\intercal A)}$$

The PCA method solves the problem with the following procedure: compute $X = USV^\intercal$, then return encoder $E = V_k$, decoder $D = V_k^\intercal$, encoded data $XV_k = U_k S_k \in \mathbb{R}^{n\times k}$, and decoded data $XV_kV_k^\intercal$. Note that $V_kV_k^\intercal \in \mathbb{R}^{d\times d}$ performs orthogonal projection onto subspace spanned by V_k.

Last lecture, we showed that the optimization problem can be re-written as

$$\min_{D\in\mathbb{R}^{k\times d}, E\in\mathbb{R}^{d\times k}} \|X - XED\|_F^2 = \min_{D\in\mathbb{R}^{d\times k}, D^\intercal D = I} \|X - XDD^\intercal\|_F^2$$

We also showed that this new objective can be further decomposed

$$\|X - XDD^\intercal\|_F^2 = \|X\|_F^2 - \|XD\|_F^2$$

This means,

$$\min_{D\in\mathbb{R}^{d\times k}, D^\intercal D = I} \|X - XDD^\intercal\|_F^2 \Leftrightarrow \max_{D\in\mathbb{R}^{d\times k}, D^\intercal D = I} \|XD\|_F^2$$

Finally, the objective value of the maximization problem is singular values squared.

$$\max_{D\in\mathbb{R}^{d\times k}, D^\intercal D = I} \|XD\|_F^2 = \|XV_k\|_F^2 = \sum_{j=1}^{k} s_j^2$$

where $s_1, \ldots, s_k$ are the top singular values of X.

Centered PCA. Typically, before running PCA, we replace each x_i with $x_i' = x_i - \overline{x}$, where $\overline{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$. The objective then becomes

$$\|X'D\|_F^2 = \text{tr}\left((X'D)^\intercal (X'D)\right) = \sum_{i=1}^{k} (X'De_i)^\intercal (X'De_i)$$

Note that $\frac{1}{n}(X'De_i)^\intercal (X'De_i)$ corresponds to the variance on the i-th coordinate after the projection. Therefore, PCA is maximizing the resulting per-coordinate variances.

Power method. How to compute the SVD of $X \in \mathbb{R}^{n\times d}$? We can use the power method to first compute v_1, u_1 and s_1. The idea is to compute the top eigenvector and eigenvalue of the matrix $X^\intercal X$:

- Start with a random vector $y_0 \sim \mathcal{N}(0, I_d)$
- For $t = 1, \dots, T$:
  $y_t \leftarrow X^\intercal X\, y_{t-1}$
- $v_1 \leftarrow y_T / \|y_T\|_2$: the first column of V and also the top eigenvector of $X^\intercal X$
- $s_1 \leftarrow \|Xv_1\|_2$ top singular value
- $u_1 \leftarrow Xv_1/s_1$

To compute the remainder of triplets (u_i, s_i, v_i), repeat the same for the residual matrix $X - s_1 u_1 v_1^\intercal$. Note that we can also apply the power method to the matrix $XX^\intercal$ for computing its top eigenvector, which is u_1. This will be useful for the next kernel PCA method.

Kernel PCA

We can find the "high variance" directions in a richer feature space by first apply some feature mapping $\phi\colon \mathbb{R}^d \to \mathbb{R}^m$ and then runs PCA over the transformed data. Let $\Phi \in \mathbb{R}^{n\times m}$ such that each row of Φ is given by $\phi(x_i)$. Let $k(\cdot,\cdot)$ be the kernel such that $k(x, y) = \phi(x)^\intercal \phi(y)$. Kernel PCA then does the following:

- Compute the Gram matrix $G = \Phi\Phi^\intercal$ and the centered Gram matrix

$$\begin{aligned}\overline{G} &= (\Phi - E\Phi)(\Phi - E\Phi)^\intercal \\ &= \Phi\Phi^\intercal - E\Phi\Phi^\intercal - \Phi\Phi^\intercal E + E\Phi\Phi^\intercal E \\ &= G - EG - GE + EGE\end{aligned}$$

  where $E \in \mathbb{R}^{n\times n}$ is the matrix with all entries of $1/n$.

- Find the top k eigenvectors of $\overline{G}$ with normalization: call it $A \in \mathbb{R}^{n\times k}$

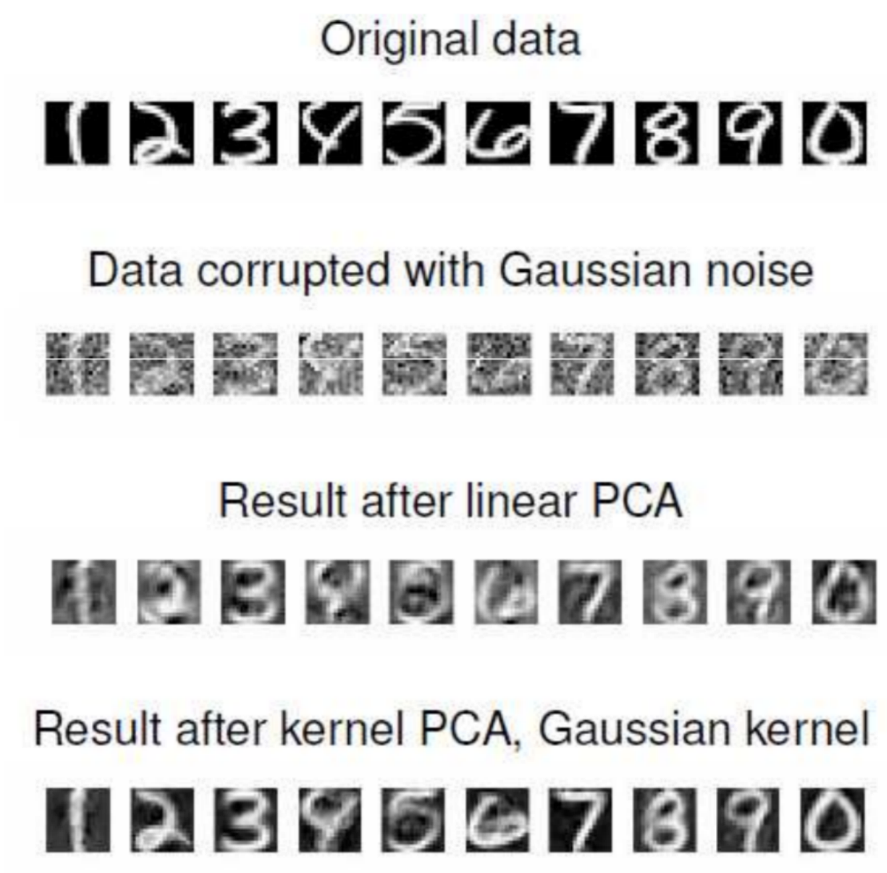

Figure 1: Denoising application of kernel PCA on the digits data set. Image from Haipeng Luo's lecture slide. Another application here.

- Construct the encoded dataset

$$(\Phi - E\Phi)(\Phi - E\Phi)^{\intercal} A = \overline{G} A$$

Maximum Likelihood Estimation

Now we will switch over to a different unsupervised learning problem that aims to model the underlying probability distribution P from which the observed examples are drawn. As we discussed before in the lecture on logistic regression, there is a general principle called *maximum likelihood estimation* (MLE):

- Pick a set of probability models for the data: $\mathcal{P} := \{p_\theta : \theta \in \Theta\}$.

- Given samples $x_1, \ldots, x_n$, pick the model that maximized the likelihood

$$\max_{\theta \in \Theta} L(\theta) = \max_{\theta \in \Theta} \ln \prod_{i=1}^{n} p_\theta(x_i) = \max_{\theta \in \Theta} \sum_{i=1}^{n} \ln p_\theta(x_i)$$

Example 0.1 (Coin flips). *Heads: $x_i = 1$ and tails $x_i = 0$. The Bernoulli distribution has the parameter–the bias or the probability of heads $\theta \in [0, 1]$. We can write*

$$p_\theta(x_i) = \theta^{x_i}(1-\theta)^{1-x_i}$$

Let $H = \sum_i x_i$ and $T = \sum_i (1 - x_i)$ be the number of heads and tails.

$$L(\theta) = \sum_{i=1}^{n} (x_i \ln \theta + (1 - x_i) \ln(1 - \theta)) = H \ln \theta + T \ln(1 - \theta).$$

By using the first-order condition, we derive that the solution is

$$\theta = \frac{H}{T + H} = \frac{H}{n}.$$

Gaussian Mixture Model

Gaussian mixture model (GMM) is the following generative model:

- Draw a latent class Y such that $\mathbf{Pr}[Y = j] = \pi_j$
- Then draw X conditioned on Y: $X \mid Y = j \sim \mathcal{N}(\mu_j, \Sigma_j)$.

The parameter $\theta = ((\pi_1, \mu_1, \Sigma_1), \ldots, (\pi_k, \mu_k, \Sigma_k))$ and the probability density at each point x is

$$p_\theta(x) = \sum_{j=1}^{k} p_{\mu_j, \Sigma_j}(x)\ \pi_j$$

where p_{μ_j, Σ_j} denotes the multivariate Gaussian density function:

$$p_{\mu_j, \Sigma_j}(x) = \frac{1}{\sqrt{(2\pi)^d \text{det}(\Sigma_j)}} \exp\left(-\frac{1}{2}(x - \mu_j)^\intercal \Sigma_j (x - \mu_j)\right)$$

The MLE problem becomes minimization of

$$L(\theta) = \sum_{i=1}^{n} \ln\left(\sum_{j=1}^{k} p_{\mu_j, \Sigma_j}(x_i)\ \pi_j\right)$$

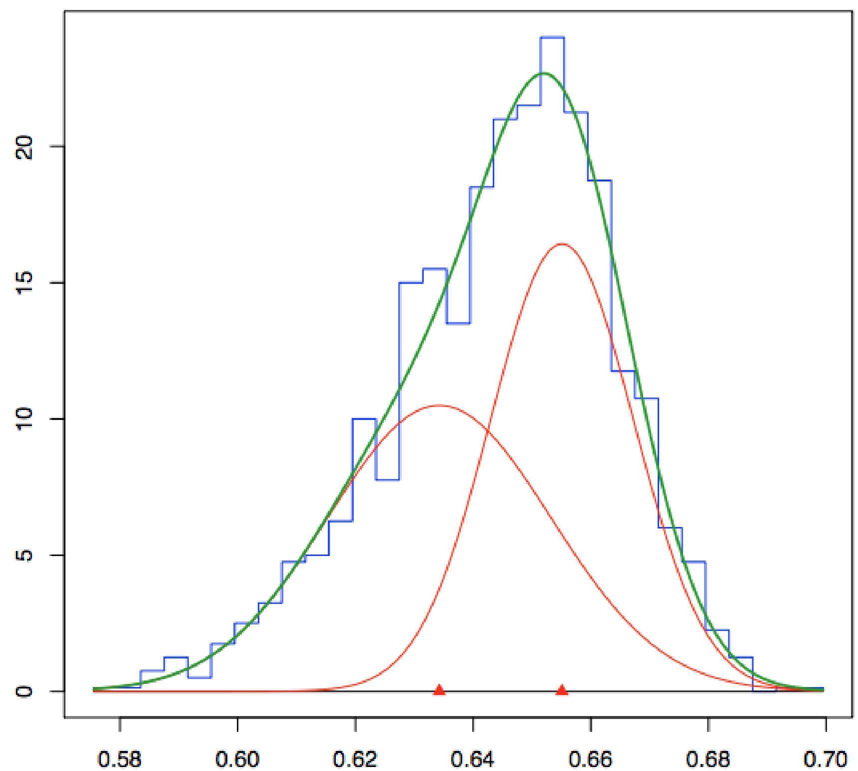

Figure 2: Statistician Karl Pearson wanted to understand the distribution of the ratio between forehead breadth and body length for crabs. He fit a mixture of two Gaussians. Figure due to Peter Macdonald.