Exam 2: Unsupervised Learning (600.692)

Instructor: René Vidal

May 12, 2017

1. **Manifold Learning.** Let $\{\boldsymbol{x}_j \in \mathbb{R}^D\}_{j=1}^N$ be a set of points that lie approximately in a manifold of dimension d embedded in $\mathbb{R}^D$. Imagine you have applied KPCA with kernel κ and LLE with K-NN to the data. Assume now you are given a new point $\boldsymbol{x} \in \mathbb{R}^D$ and you wish to find its corresponding point $\boldsymbol{y} \in \mathbb{R}^d$ according to KPCA and LLE. How would you compute $\boldsymbol{y} \in \mathbb{R}^d$ without applying KPCA or LLE from scratch to the $N+1$ points? Under what conditions the solution you propose is equivalent to applying KPCA or LLE to the $N+1$ points?

   ANSWER: In the case of PCA, recall that given a subspace with parameters $\boldsymbol{\mu}_N$ and U_N estimated from the N data points, the low-dimensional coordinates associated with a new point $\boldsymbol{x}$ are given by $\boldsymbol{y} = U_N^\top(\boldsymbol{x} - \boldsymbol{\mu}_N)$. Therefore, $\boldsymbol{y}$ can be estimated directly without having to recompute the subspace anew. Now, if we choose to recompute $\boldsymbol{\mu}$ and U from the $N+1$ points including $x_{N+1} \doteq \boldsymbol{x}$, the new mean will be given by

$$\boldsymbol{\mu}_{N+1} = \frac{1}{N+1}\sum_{j=1}^{N+1}\boldsymbol{x}_j = \frac{1}{N+1}(\sum_{j=1}^{N}\boldsymbol{x}_j + \boldsymbol{x}_{N+1}) = \frac{N}{N+1}\boldsymbol{\mu}_N + \frac{1}{N+1}\boldsymbol{x}_{N+1},$$

   which is equal to $\boldsymbol{\mu}_N$ only if $\boldsymbol{x}_{N+1} = \boldsymbol{\mu}_N$. In this case, the new covariance matrix $\Sigma_{N+1} = \frac{1}{N+1}\sum_{j=1}^{N+1}(\boldsymbol{x}_j - \boldsymbol{\mu}_{N+1})(\boldsymbol{x}_j - \boldsymbol{\mu}_{N+1})^\top = \frac{1}{N+1}\sum_{j=1}^{N+1}(\boldsymbol{x}_j - \boldsymbol{\mu}_N)(\boldsymbol{x}_j - \boldsymbol{\mu}_N)^\top = \frac{N}{N+1}\Sigma_N + \frac{1}{N+1}(\boldsymbol{x}_{N+1} - \boldsymbol{\mu}_N)(\boldsymbol{x}_{N+1} - \boldsymbol{\mu}_N)^\top$ reduces to $\frac{N}{N+1}\Sigma_N$, hence the top d eigenvectors of Σ_N and Σ_{N+1} are the same, and so $U_{N+1} = U_N$.

   In the case of KPCA, recall from Mercer's theorem that given a kernel κ that satisfies some suitable conditions, there exists an embedding ϕ such that $\kappa(\boldsymbol{x}, \boldsymbol{y}) = \phi(\boldsymbol{x})^\top\phi(\boldsymbol{y})$. We can use this embedding to define the mean embedded vector $\bar{\boldsymbol{\phi}}_N = \frac{1}{N}\sum\phi(\boldsymbol{x}_j)$ and the embedded data matrix $\Phi = [\phi(\boldsymbol{x}_1) - \bar{\boldsymbol{\phi}}_N, \ldots, \phi(\boldsymbol{x}_N) - \bar{\boldsymbol{\phi}}_N]$. It follows from (4.23) of the GPCA book that the low-dimensional coordinates of a new point $\boldsymbol{x}$ can be computed as

$$\boldsymbol{y} = W^\top\Phi^\top(\phi(\boldsymbol{x}) - \bar{\boldsymbol{\phi}}_N) = W^\top\tilde{\kappa}_{\boldsymbol{x}},$$

   where $W \in \mathbb{R}^{N\times d}$ is a matrix whose ith column, $\boldsymbol{w}_i$, is the eigenvector of the centered kernel matrix $\tilde{\mathcal{K}} = \Phi^\top\Phi$ associated with its ith largest eigenvalue, λ_i, and normalized so that $\|\boldsymbol{w}_i\| = \lambda_i^{-2}$, and the vector $\tilde{\kappa}_{\boldsymbol{x}}$ is defined as

$$\tilde{\kappa}_{\boldsymbol{x}} = \Phi^\top(\phi(\boldsymbol{x}) - \bar{\boldsymbol{\phi}}_N) = [\tilde{\kappa}(\boldsymbol{x}_1, \boldsymbol{x}), \tilde{\kappa}(\boldsymbol{x}_2, \boldsymbol{x}), \ldots, \tilde{\kappa}(\boldsymbol{x}_N, \boldsymbol{x})]^\top \in \mathbb{R}^N,$$

   where $\tilde{\kappa}$ is the centered kernel

$$\tilde{\kappa}(\boldsymbol{x}, \boldsymbol{y}) = (\phi(\boldsymbol{x}) - \bar{\boldsymbol{\phi}}_N)^\top(\phi(\boldsymbol{y}) - \bar{\boldsymbol{\phi}}_N) = \kappa(\boldsymbol{x}, \boldsymbol{y}) - \frac{1}{N}\sum_{j=1}^{N}\kappa(\boldsymbol{x}, \boldsymbol{x}_j) - \frac{1}{N}\sum_{i=1}^{N}\kappa(\boldsymbol{x}_i, \boldsymbol{y}) + \frac{1}{N^2}\sum_{i=1}^{N}\sum_{j=1}^{N}\kappa(\boldsymbol{x}_i, \boldsymbol{y}_j).$$

   Therefore, in the case of KPCA we can compute the vector $\boldsymbol{y}$ directly from the kernel matrix for N points. Alternatively, if we were to compute $\boldsymbol{y}$ from the kernel matrix for $N+1$ points, the low-dimensional coordinate for the $(N+1)$st point would not be the same, unless $\phi(\boldsymbol{x}_{N+1}) = \bar{\boldsymbol{\phi}}_N$ in which case $\bar{\boldsymbol{\phi}}_{N+1} = \bar{\boldsymbol{\phi}}_N$ and $(\phi(\boldsymbol{x}_{N+1}) - \bar{\boldsymbol{\phi}}_N)(\phi(\boldsymbol{x}_{N+1}) - \bar{\boldsymbol{\phi}}_N)^\top$ does not affect the top d eigenvectors of the embedded covariance matrix.

   Now, in the case of LLE, recall that each data point is expressed approximately as an affine combination of its K-NN, i.e., $\boldsymbol{x}_j \approx \sum_{i=1}^N \boldsymbol{x}_i c_{ij}$, where $\sum_{j=1}^N c_{ij} = 1$ and $c_{ij} = 0$ if $\boldsymbol{x}_i$ is not a K-NN of $\boldsymbol{x}_j$. Recall also that the calculation of the coefficients c_{ij} can be done locally, i.e., it depends on points $\boldsymbol{x}_j$ and its K-NN, $\boldsymbol{x}_{j_1}, \boldsymbol{x}_{j_2}, \ldots, \boldsymbol{x}_{j_K}$. Then, the low-dimensional coordinates are obtained so that $\boldsymbol{y}_j \approx \sum_{i=1}^N \boldsymbol{y}_i c_{ij}$. Therefore, given a new point $\boldsymbol{x}$, a simple method for obtaining its low-dimensional representation is to identify its K-NN,

find its coefficients $c_{\cdot,N+1}$, and then define its low-dimensional representation as $\boldsymbol{y} = \sum_{i=1}^{N} \boldsymbol{y}_i c_{i,N+1}$. Suppose now that we apply LLE to the $N+1$ points. Then, if the K-NN of the first N points do not change, then the coefficients c_{ij} do not change for $(i,j) \in N \times N$. The only new coefficients are those for the new point $c_{\cdot,N+1}$. Assume further that the point $\boldsymbol{x}_{N+1}$ can be written as an exact linear combination of its K-NN, i,e,m there are $c_{\cdot,N+1}$ such that $\boldsymbol{x}_{N+1} = \sum c_{i,N+1} \boldsymbol{x}_i$. Then the reconstruction error is not affected by the new data point, and hence $\boldsymbol{y} = \sum_{i=1}^{N} \boldsymbol{y}_i c_{i,N+1}$.

2. **K-Subspaces.** Consider the objective function of the K-subspaces algorithm:

$$f(\{\boldsymbol{\mu}_i\}_{i=1}^n, \{U_i\}_{i=1}^n) = \sum_{j=1}^{N} \min_{i=1,\dots,n} \|(I - U_i U_i^\top)(\boldsymbol{x}_j - \boldsymbol{\mu}_i)\|^2. \tag{1}$$

Let $\{\boldsymbol{\mu}_i^{(k)}\}_{i=1}^n, \{U_i^{(k)}\}_{i=1}^n$ be the estimates of the subspace parameters at the kth iteration of the K-subspaces algorithm. Show that the iterations of K-subspaces are such that

$$f(\{\boldsymbol{\mu}_i^{(k+1)}\}_{i=1}^n, \{U_i^{(k+1)}\}_{i=1}^n) \leq f(\{\boldsymbol{\mu}_i^{(k)}\}_{i=1}^n, \{U_i^{(k)}\}_{i=1}^n). \tag{2}$$

ANSWER: Let S_i denote the ith subspace, $d(\boldsymbol{x}_j, S_i) = \|(I - U_i U_i^\top)(\boldsymbol{x}_j, \boldsymbol{\mu}_i)\|$ denote the distance from point $\boldsymbol{x}_j$ to subspace S_i, and $S = \{S_i\}_{i=1}^n$ denote the collection of all subspaces. Then the objective function can be rewritten as $f(S) = \sum_{j=1}^{N} \min_{i=1,\dots,n} d(\boldsymbol{x}_j, S_i)^2$. Therefore, our goal is to prove that $f(S^{(k+1)}) \leq f(S^{(k)})$, where $S^{(k)}$ is the estimate of all subspaces at the kth iteration of K-subspaces. Now, recall that the kth iteration of K-subspaces consists of two steps: (1) finding the optimal subspaces $S^{(k+1)}$ given the current assignments of points to subspaces $w_{ij}^{(k)}$ and (2) finding the optimal assignments $w_{ij}^{(k+1)}$ given the subspaces $S^{(k+1)}$, that is

$$S^{(k+1)} = \operatorname*{argmin}_{S} \sum_{j=1}^{N} \sum_{i=1}^{n} w_{ij}^{(k)} d(\boldsymbol{x}_j, S_i)^2 \quad \text{and} \quad w_{ij}^{(k+1)} = \begin{cases} 1 & i = \operatorname*{argmin}_{\ell=1,\dots,n} d(\boldsymbol{x}_j, S_\ell^{(k+1)})^2 \\ 0 & \text{else.} \end{cases}$$

Now,

$$\begin{aligned} f(S^{(k+1)}) &= \sum_{j=1}^{N} \min_{i=1,\dots,n} d(\boldsymbol{x}_j, S_i^{(k+1)})^2 = \sum_{j=1}^{N} \sum_{i=1}^{n} w_{ij}^{(k+1)} d(\boldsymbol{x}_j, S_i^{(k+1)})^2 \qquad \text{(by definition of } w_{ij}^{(k+1)}\text{)} \\ &\leq \sum_{j=1}^{N} \sum_{i=1}^{n} w_{ij}^{(k)} d(\boldsymbol{x}_j, S_i^{(k+1)})^2 \qquad \text{(because } w_{ij}^{(k+1)} \text{ are the best assignments of } \boldsymbol{x}_j \text{ to } S_i^{(k+1)}\text{)} \\ &\leq \sum_{j=1}^{N} \sum_{i=1}^{n} w_{ij}^{(k)} d(\boldsymbol{x}_j, S_i^{(k)})^2 \quad \text{(because } S_i^{(k+1)} \text{ are the optimal subspaces given the assignments } w_{ij}^{(k)}\text{)} \\ &= f(S^{(k)}), \end{aligned}$$

which proofs the claim.

3. **Low-Rank Subspace Clustering.** Let $X = [\boldsymbol{x}_1, \dots, \boldsymbol{x}_N] \in \mathbb{R}^{D \times N}$ be a data matrix whose columns are drawn from a union of n subspaces. Let $X = U \Sigma V^\top$ and $X = U_1 \Sigma_1 V_1^\top$ be, respectively, the full and compact SVDs of X, with V partitioned as $[V_1, V_2]$, where $V_1 \in \mathbb{R}^{N \times r}$, $V_2 \in \mathbb{R}^{N \times (N-r)}$ and $\operatorname{rank}(X) = r$. Let us express each data point as a linear combination of all data points, i.e., for all j, $\boldsymbol{x}_j = \sum_{i=1}^{N} \boldsymbol{x}_i c_{ij}$, or equivalently $\boldsymbol{x}_j = X \boldsymbol{c}_j$, where $\boldsymbol{c}_j \in \mathbb{R}^N$. Let us now search for a matrix of coefficients $C = [\boldsymbol{c}_1, \dots, \boldsymbol{c}_N] \in \mathbb{R}^{N \times N}$ that solves the following optimization problem:

$$\min_{C} \ \|C\|_* + \frac{\lambda}{2} \|C\|_F^2 \ \text{ s.t. } \ X = XC \ \text{ and } \ C = C^\top, \tag{3}$$

where $\lambda > 0$ is a parameter. Prove that $C^* = V_1 V_1^\top$.

Hint: We showed in class that the solutions to $X = XC$ are of the form $C = V_1 V_1^\top + V_2 A$, for $A \in \mathbb{R}^{(N-r) \times N}$.

ANSWER: Since the solutions to $X = XC$ are of the form $C = V_1 V_1^\top + V_2 A$ for some A, and C must also be symmetric, we have that $V_1 V_1^\top + V_2 A = V_1 V_1^\top + A^\top V_2^\top \iff V_2 A = A^\top V_2^\top$. Therefore, we must have that $A = \Sigma V_2^\top$ for some matrix $\Sigma \in \mathbb{R}^{(N-r)\times(N-r)}$. This implies that

$$C = \begin{bmatrix} V_1 & V_2 \end{bmatrix} \begin{bmatrix} I & 0 \\ 0 & \Sigma \end{bmatrix} \begin{bmatrix} V_1 & V_2 \end{bmatrix}^\top$$

and so $\|C\|_* + \frac{\lambda}{2}\|C\|_F^2 = r + \|\Sigma\|_* + \frac{\lambda}{2}(r + \|\Sigma\|_F^2)$. Therefore, the optimization problem in (3) thus reduces to $\min_\Sigma \|\Sigma\|_* + \frac{\lambda}{2}\|\Sigma\|_F^2$, whose optimal solution is $\Sigma = 0$. Therefore, $C = V_1 V_1^\top$ as claimed.