HOMEWORK SET 5

CSCI5525 Machine Learning: Analysis and Methods (Fall 2024)

Due 11:59 pm, Dec 23 2024

Instruction Your writeup, either typeset or scanned, should be a single PDF file. For problems requiring coding, organize all codes for all problems into **ONE** Jupyter notebook file (i.e., `.ipynb` file) with cell execution outputs. Your submission to Gradescope should include the single PDF and the one notebook file—**please DO NOT zip them**! Please assign page(s) to each question to help reduce the TA's navigation time. If your notebook submission does not display properly on Gradescope due to a large file error, please try to remove images and/or figures from your cell outputs and re-upload it. No late submission will be accepted. For each problem, you should acknowledge your collaborators—**including AI tools**, if any.

About the use of AI tools You are strongly encouraged to use AI tools—they are becoming our workspace friends, such as ChatGPT (https://chat.openai.com/), Claude (https://claude.ai/chats), and Github Copilot (https://github.com/features/copilot), to help you when trying to solve problems. It takes a bit of practice to ask the right and effective questions/prompts to these tools; we highly recommend that you go through this popular free short course **ChatGPT Prompt Engineering for Developers** offered by https://learn.deeplearning.ai/ to get started.

If you use any AI tools for any of the problems, you should include screenshots of your prompting questions and their answers in your writeup. The answers provided by such AI tools often contain factual errors and reasoning gaps. **So, if you only submit an AI answer with such bugs for any problem, you will obtain a zero score for that problem**. You obtain the scores only when you explain the bugs and also correct them in your own writing. You can also choose not to use any of these AI tools, in which case we will grade based on the efforts you have made.

Reminder about notations We will use small letters (e.g., u) for scalars, small boldface letters (e.g., $\boldsymbol{a}$) for vectors, and capital boldface letters (e.g., $\boldsymbol{A}$) for matrices. For a matrix $\boldsymbol{A}$, $\boldsymbol{a}^i$ (supscripting) means its i-th row as a *row vector*, and $\boldsymbol{a}_j$ (subscripting) means the j-the column as a column vector, and a_{ij} means its (i,j)-th element. $\mathbb{R}$ is the set of real numbers. $\mathbb{R}^n$ is the space of n-dimensional real vectors, and similarly $\mathbb{R}^{m\times n}$ is the space of $m \times n$ real matrices. The dotted equal sign $\doteq$ means defining.

Problem 1 (Dimension reduction; 7.5/15)

(a) In our geometric view of PCA, we try to find a best-fitting low-dimensional subspace to our dataset. Mathematically, let $\boldsymbol{X} \in \mathbb{R}^{N\times d}$ be our data matrix, and $\overline{\boldsymbol{X}}$ the centered version. PCA tries to solve the following problem

$$\min_{\boldsymbol{U}\in\mathbb{R}^{d\times r}:\boldsymbol{U}^\intercal\boldsymbol{U}=\boldsymbol{I}} \quad \frac{1}{N}\sum\nolimits_{i\in[N]} \|\overline{\boldsymbol{x}_i} - \boldsymbol{U}\boldsymbol{U}^\intercal\overline{\boldsymbol{x}_i}\|_2^2\,, \tag{1}$$

where r is our target dimension for the subspace. We know that the optimal solution is the $\boldsymbol{U}$ that collects the top r right singular vectors of $\overline{\boldsymbol{X}}$ (or equivalently, the top r eigenvectors of $\frac{1}{N}\overline{\boldsymbol{X}}^\intercal\overline{\boldsymbol{X}}$).

Instead of performing the centering step separately, we can also try to fit an r-dimensional affine subspace directly. Since any affine subspace can be represented as $\{\boldsymbol{U}\boldsymbol{z}+\boldsymbol{v} : \boldsymbol{z}\in\mathbb{R}^r\}$ for some orthonormal $\boldsymbol{U}\in\mathbb{R}^{d\times r}$ and some $\boldsymbol{v}$, we can formulate the learning problem as

$$\min_{\boldsymbol{U}\in\mathbb{R}^{d\times r}:\boldsymbol{U}^\intercal\boldsymbol{U}=\boldsymbol{I},\ \boldsymbol{z}_i\in\mathbb{R}^r\ \forall i,\boldsymbol{v}\in\mathbb{R}^d} \quad \frac{1}{N}\sum\nolimits_{i\in[N]} \|\boldsymbol{x}_i - \boldsymbol{v} - \boldsymbol{U}\boldsymbol{z}_i\|_2^2\,. \tag{2}$$

(i) Prove that for any fixed $\boldsymbol{U}$, an optimal solution for Eq. (2) is $\boldsymbol{v} = \frac{1}{N}\sum_{i\in[N]} \boldsymbol{x}_i$ and $\boldsymbol{z}_i = \boldsymbol{U}^\intercal(\boldsymbol{x}_i - \boldsymbol{v})$, i.e., solving Eq. (2) is equivalent to centering as data preprocessing and then solving Eq. (1). (1/15)

(ii) Consider $d = 50$, $N = 200$, and $r = 10$. Fix a random seed, and generate data as follows: (1) generate a subspace basis $\boldsymbol{A}_0 \in \mathbb{R}^{d\times r}$ with iid $N(0, 1/r)$ entries; (2) generate a coefficient matrix $\boldsymbol{Z}_0 \in \mathbb{R}^{r\times N}$ with iid $N(0, 1)$ entries; (3) generate the data as $\boldsymbol{X} = (\boldsymbol{A}_0\boldsymbol{Z}_0 + \boldsymbol{N})^\intercal$, where $\boldsymbol{N} \in \mathbb{R}^{d\times N}$ contains iid $N(0, 0.05)$ entries. **Implement PCA (including centering, but no variance normalization) on $\boldsymbol{X}$ to obtain your estimation of the subspace basis, call it $\boldsymbol{A}_1 \in \mathbb{R}^{d\times r}$.** To compare the two subspaces represented by $\boldsymbol{A}_0$ and $\boldsymbol{A}_1$, a reasonable metric is

$$d(\mathrm{span}(\boldsymbol{A}_1), \mathrm{span}(\boldsymbol{A}_0)) = \|\boldsymbol{A}_1\boldsymbol{A}_1^\dagger - \boldsymbol{A}_0\boldsymbol{A}_0^\dagger\|_F / \|\boldsymbol{A}_0\boldsymbol{A}_0^\dagger\|_F, \tag{3}$$

where for any matrix $\boldsymbol{M}$, $\boldsymbol{M}^\dagger$ denotes its pseudoinverse (https://en.wikipedia.org/wiki/Moore%E2%80%93Penrose_inverse). Here, $\boldsymbol{A}_1\boldsymbol{A}_1^\dagger$ and $\boldsymbol{A}_0\boldsymbol{A}_0^\dagger$ are the projectors on the subspaces $\boldsymbol{A}_1$ and $\boldsymbol{A}_0$ span, respectively, and $\|\boldsymbol{A}_0\boldsymbol{A}_0^\dagger\|_F$ performs proper normalization of the subspace distance. **Report your $d(\mathrm{span}(\boldsymbol{A}_1), \mathrm{span}(\boldsymbol{A}_0))$; ideally, it should be close to 0.** (1/15)

(iii) Linear autoencoders are also known to perform PCA. Consider the following formulation of a linear autoencoder:

$$\min_{\boldsymbol{A}\in\mathbb{R}^{d\times r}} \frac{1}{N}\sum\nolimits_{i\in[N]} \|\overline{\boldsymbol{x}_i} - \boldsymbol{A}\boldsymbol{A}^\intercal\overline{\boldsymbol{x}_i}\|_2^2, \tag{4}$$

where $\boldsymbol{A}^\intercal$ denotes the linear encoder and $\boldsymbol{A}$ denotes the linear decoder, and our encoder and decoder are symmetric with respect to each other (i.e., the transpose of each other). **Implement the gradient descent with line search to solve Eq. (4) with the same data $\boldsymbol{X}$ and $\overline{\boldsymbol{X}}$ you generate in (ii), and obtain an estimate of the subspace basis $\boldsymbol{A}_2$. Report your $d(\mathrm{span}(\boldsymbol{A}_2), \mathrm{span}(\boldsymbol{A}_0))$; is it close to 0?** (1/15)

(iv) Now keep your $\boldsymbol{A}_0$ and $\boldsymbol{Z}_0$ in (ii), but change to noise model to Laplace noise (https://en.wikipedia.org/wiki/Laplace_distribution), i.e., $\boldsymbol{X}' = (\boldsymbol{A}_0\boldsymbol{Z}_0 + \boldsymbol{L})^\intercal$, where $\boldsymbol{L}$ contains iid $\mathrm{Laplace}(0, 0.15)$ entries. Due to the long-tailed nature of Laplace noise, the noise magnitudes on certain coordinates can be substantially larger than the rest. In this case, it is appropriate to switch the loss in Eq. (2) to a robust loss, say the ℓ_1 loss, yielding:

$$\min\nolimits_{\boldsymbol{U}\in\mathbb{R}^{d\times r}:\boldsymbol{U}^\intercal\boldsymbol{U}=\boldsymbol{I},\ \boldsymbol{z}_i\in\mathbb{R}^r\ \forall i, \boldsymbol{v}\in\mathbb{R}^d} \quad \frac{1}{N}\sum\nolimits_{i\in[N]} \|\boldsymbol{x}_i - \boldsymbol{v} - \boldsymbol{U}\boldsymbol{z}_i\|_1. \tag{5}$$

However, unlike for Eq. (2) that we can eliminate $\boldsymbol{z}_i$'s and $\boldsymbol{v}$ (as proved in (i)), here, we have to solve the three groups of variables altogether. To solve Eq. (5), we can ignore the $\boldsymbol{U}^\intercal\boldsymbol{U} = \boldsymbol{I}$, and consider alternatingly minimizing over the two groups of variables: $\boldsymbol{U}$ vs. $\boldsymbol{z}_i's, \boldsymbol{v}$—when we fix one group, the objective function is convex with respect to the other group. So, we can implement the alternating minimization algorithm (see, e.g., https://www.mit.edu/~rakhlin/6.883/lectures/lecture07.pdf for more information), and solve the two convex subproblems with say, the CVXPY package (https://www.cvxpy.org/; we have used it in our earlier HW sets). **Implement the suggested algorithm, or whatever numerical algorithm you like to solve Eq. (5) to yield your subspace estimate $\boldsymbol{A}_3$, and report your $d(\mathrm{span}(\boldsymbol{A}_3), \mathrm{span}(\boldsymbol{A}_0))$. For comparison, also apply the plain PCA on $\boldsymbol{X}'$ to produce another subspace estimate $\boldsymbol{A}_4$; report your $d(\mathrm{span}(\boldsymbol{A}_4), \mathrm{span}(\boldsymbol{A}_0))$. Do what you observe?** (1.5/15)

(b) The famous Johnson–Lindenstrauss (J-L) lemma states: for any $\varepsilon \in (0,1)$ and any N set $S = \{\boldsymbol{x}_i\}_{i\in[N]} \subset \mathbb{R}^d$, there exists a matrix $\boldsymbol{A} \in \mathbb{R}^{d'\times d}$ for every $d' \geq C\log N/\varepsilon^2$ ($C > 0$ is a universal constant) so that

$$(1-\varepsilon)\left\|\boldsymbol{u}-\boldsymbol{v}\right\|_2^2 \leq \left\|\boldsymbol{A}\boldsymbol{u}-\boldsymbol{A}\boldsymbol{v}\right\|_2^2 \leq (1+\varepsilon)\left\|\boldsymbol{u}-\boldsymbol{v}\right\|_2^2 \ \forall \boldsymbol{u},\boldsymbol{v}\in S, \tag{6}$$

$$(1-\varepsilon)\left\|\boldsymbol{u}+\boldsymbol{v}\right\|_2^2 \leq \left\|\boldsymbol{A}\boldsymbol{u}+\boldsymbol{A}\boldsymbol{v}\right\|_2^2 \leq (1+\varepsilon)\left\|\boldsymbol{u}+\boldsymbol{v}\right\|_2^2 \ \forall \boldsymbol{u},\boldsymbol{v}\in S, \tag{7}$$

where the second line is not typically stated in the J-L lemma, but can be proved easily using a very similar argument to that for the first line. In particular, a random draw iid Gaussian matrix $\frac{1}{\sqrt{d'}}\boldsymbol{G}$ satisfies the property with very high probability.

Now, assume that all points inside S have unit norm, i.e., $\left\|\boldsymbol{x}_i\right\|_2 = 1\ \forall i \in [N]$. Show that for any $d' \geq C\log N/\varepsilon^2$, we can also construct a matrix $\boldsymbol{B}$ so that

$$\left|\langle \boldsymbol{B}\boldsymbol{u}, \boldsymbol{B}\boldsymbol{v}\rangle - \langle \boldsymbol{u},\boldsymbol{v}\rangle\right| \leq \varepsilon\ \forall \boldsymbol{u},\boldsymbol{v}\in S. \tag{8}$$

(**Hint: consider the identity** $4\langle \boldsymbol{x},\boldsymbol{x}'\rangle = (\left\|\boldsymbol{x}+\boldsymbol{x}'\right\|_2^2 - \left\|\boldsymbol{x}-\boldsymbol{x}'\right\|_2^2)$ **that holds for arbitrary** $\boldsymbol{x}$ **and** $\boldsymbol{x}'$ **in** $\mathbb{R}^d$**.**) (1/15)

(c) In the ISOMAP method for nonlinear dimension reduction, we need to turn a Euclidean distance matrix into a similarity matrix. Let $\boldsymbol{X} \in \mathbb{R}^{N\times d}$ be the data matrix (i.e., we have N points in $\mathbb{R}^d$), and let $\boldsymbol{X}^* = \boldsymbol{X} - \frac{1}{N}\boldsymbol{1}\boldsymbol{1}^\intercal\boldsymbol{X}$ be the centered version of $\boldsymbol{X}$ ($\boldsymbol{1}$'s are all-one vectors of appropriate dimensions). Moreover, let $\boldsymbol{D} \in \mathbb{R}^{N\times N}$ denote the squared Euclidean distance matrix, i.e., $d_{ij} = \left\|\boldsymbol{x}_i - \boldsymbol{x}_j\right\|_2^2$, and write $\boldsymbol{G} = \boldsymbol{X}^\intercal\boldsymbol{X}$. Our goal is to write $\boldsymbol{G}^* \doteq (\boldsymbol{X}^*)^\intercal\boldsymbol{X}^*$ as a function of $\boldsymbol{D}$.

(i) Can you represent $\boldsymbol{G}^*$ as a function of $\boldsymbol{G}$? (0.5/15)

(ii) Combine (i) and the fact that $g_{ij} = \frac{1}{2}\left(\left\|\boldsymbol{x}_i\right\|_2^2 + \left\|\boldsymbol{x}_i\right\|_2^2 - d_{ij}\right), \forall i,j$ to show that

$$g_{ij}^* = -\frac{1}{2}\left(d_{ij} - \frac{1}{N}\boldsymbol{d}^i\boldsymbol{1} - \frac{1}{N}\boldsymbol{1}^\intercal\boldsymbol{d}_j + \frac{1}{N^2}\boldsymbol{1}^\intercal\boldsymbol{D}\boldsymbol{1}\right). \tag{9}$$

(1/15)

(iii) Show that $\boldsymbol{G}^* = -\frac{1}{2}\left(\boldsymbol{I} - \frac{1}{N}\boldsymbol{1}\boldsymbol{1}^\intercal\right)\boldsymbol{D}\left(\boldsymbol{I} - \frac{1}{N}\boldsymbol{1}\boldsymbol{1}^\intercal\right)$. (0.5/15)

Problem 2 (Clustering; 6/15)

(a) The classical k-means algorithm can be viewed as an alternating minimization algorithm to solve

$$\min_{\boldsymbol{\mu}_1,\ldots,\boldsymbol{\mu}_K,\, \boldsymbol{M}\in\{0,1\}^{N\times K}, \boldsymbol{M}\boldsymbol{1}=\boldsymbol{1}} \frac{1}{N}\sum\nolimits_{i\in[N]}\sum\nolimits_{k\in[K]} m_{ik}d\left(\boldsymbol{x}_i,\boldsymbol{\mu}_k\right). \tag{10}$$

Here, $\boldsymbol{M} \in \{0,1\}^{N\times K}$ is the cluster-assignment matrix, and $\forall i,k$, $m_{ik} = 1$ means $\boldsymbol{x}_i$ is assigned to cluster k. The constraint $\boldsymbol{M}\boldsymbol{1} = \boldsymbol{1}$ ensures that each point is only assigned to one cluster. Consider the k-medoids variant of k-means, which requires the cluster centroids $\mu_1,\ldots,\mu_K$ come from the dataset $S = \{\boldsymbol{x}_i\}_{i\in[N]}$, not from outside. **Change the formulation in Eq. (10) to reflect this, and also derive an alternating minimization algorithm to solve the k-medoids problem.** (1/15)

(b) In spectral clustering, the graph is represented by a weight/similarity matrix $\boldsymbol{W} \in \mathbb{R}^{N \times N}$ (N points are assumed), whose entries are large when the connection between the corresponding nodes are strong. A critical quantity is the graph Laplacian:

$$\boldsymbol{L} = \boldsymbol{D} - \boldsymbol{W}, \quad \boldsymbol{D} \doteq \mathrm{diag}(\boldsymbol{W}\boldsymbol{1}) : \text{degree matrix}. \tag{11}$$

(i) A crucial identity we use to derive the final learning objective in spectral clustering is

$$\forall \boldsymbol{v},\ \boldsymbol{v}^\intercal \boldsymbol{L} \boldsymbol{v} = \frac{1}{2} \sum\nolimits_{i \in [N], j \in [N]} w_{ij} (v_i - v_j)^2. \tag{12}$$

Show it! $(1/15)$

(ii) Assume that all entries of $\boldsymbol{W}$ are nonnegative, i.e., $\boldsymbol{W} \geq 0$. Prove that $\boldsymbol{L}$ is positive semidefinite, and also that 0 is an eigenvalue of $\boldsymbol{L}$. $(1/15)$

(iii) Suppose there are exactly $K \leq N$ connected subgraphs. It can be shown that $\boldsymbol{L}$ has exactly K zero eigenvalues and there are K eigenvectors exactly indicate the cluster memberships of all data points. If we obtain such eigenvectors, we are done. But, when we have K equal eigenvalues, all vectors from the corresponding K-dimensional subspace are valid eigenvectors, so our eigensolver may not return the ones we like. Let $\boldsymbol{H}^* \in \mathbb{R}^{N \times K}$ be the ideal eigenvectors, and $\widehat{\boldsymbol{H}}$ the one we actually obtain. It must hold that $\boldsymbol{H}^* = \widehat{\boldsymbol{H}} \boldsymbol{R}$ for an unknown rotation matrix $\boldsymbol{R} \in \mathbb{R}^{K \times K}$, with $\boldsymbol{R}^\intercal \boldsymbol{R} = \boldsymbol{Q}\boldsymbol{R}^\intercal = \boldsymbol{I}$. **Show that for any two vectors $\boldsymbol{u}, \boldsymbol{v} \in \mathbb{R}^K$, $\|(\boldsymbol{u} - \boldsymbol{v})\boldsymbol{R}\|_2 = \|\boldsymbol{u} - \boldsymbol{v}\|_2$, and thereby explain why the subsequent k-means step with ℓ_2 distance on the rows of $\widehat{\boldsymbol{H}}$ makes sense.** $(1/15)$

(c) In the mean-shift algorithm for clustering, the underlying density is estimated via kernel density estimation that typically takes the form

$$\widehat{p}(\boldsymbol{x}) = \frac{1}{N} \sum\nolimits_{i \in [N]} K(\|\boldsymbol{x} - \boldsymbol{x}_i\|_2), \tag{13}$$

where K is a smoothing kernel that satisfies $\int_{\boldsymbol{x}} K(\|\boldsymbol{x} - \boldsymbol{x}_i\|_2)\ d\boldsymbol{x} = 1$ for all $i \in [N]$ so that $\widehat{p}(\boldsymbol{x})$ is a valid density function. Consider the kernel

$$K(\|\boldsymbol{x} - \boldsymbol{x}'\|_2) = c_d \max\left(0, 1 - \|\boldsymbol{x} - \boldsymbol{x}'\|_2^2\right), \tag{14}$$

where c_d is a universal constant to ensure $\int_x c_d \max\left(0, 1 - \|\boldsymbol{x} - \boldsymbol{x}'\|_2^2\right)\ d\boldsymbol{x} = 1$. **Derive the key update formula for mean-shift with this kernel**. $(1/15)$

(d) In $\mathbb{R}^d$, consider mixture modeling with K multivariate Bernoulli components, i.e.,

$$p(\boldsymbol{x}; \boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_K) = \sum\nolimits_{k \in [K]} w_k \prod\nolimits_{j \in [d]} \mu_{kj}^{x_j} (1 - \mu_{kj})^{1 - x_j}, \tag{15}$$

i.e., for each mixture component, the d coordinates are assumed to be independent for simplicity. **Derive an EM style algorithm for fitting this mixture model to a dataset $S = \{\boldsymbol{x}_i\}_{i \in [N]}$ using the principle of maximum likelihood estimation**. $(1/15)$

Problem 3 (Generative modeling; $1.5/15$) Consider a transformation $\boldsymbol{x} = f(\boldsymbol{z})$ and its inverse $\boldsymbol{z} = g(\boldsymbol{x})$.

(a) By differentiating both sides of $\boldsymbol{x} = f \circ g(\boldsymbol{x})$ with respect to $\boldsymbol{x}$, show that $\boldsymbol{J}_f(g(\boldsymbol{x}))\boldsymbol{J}_g(\boldsymbol{x}) = \boldsymbol{I}$, where $\boldsymbol{J}$ denotes Jacobian matrix (refer to our notes on calculus review); $(0.5/15)$

(b) Given the density function of $\boldsymbol{z}$ as $p_{\boldsymbol{z}}(\boldsymbol{z})$, the change-of-variable formula reads

$$p_{\boldsymbol{x}}(\boldsymbol{x}) = p_{\boldsymbol{z}}(g(\boldsymbol{x})) \left|\det \boldsymbol{J}_g(\boldsymbol{x})\right|. \tag{16}$$

Show that we can also write it as

$$p_{\boldsymbol{x}}(\boldsymbol{x}) = p_{\boldsymbol{z}}(g(\boldsymbol{x})) \left|\det \boldsymbol{J}_f(\boldsymbol{z})\right|^{-1}. \tag{17}$$

(**Hint: the determinant of the product of two square matrices is the product of their determinants.**) $(0.5/15)$

(c) Assume that the entries inside $\boldsymbol{z} \in \mathbb{R}^d$ are iid, and each follows a half-normal distribution (https://en.wikipedia.org/wiki/Half-normal_distribution), i.e., $\forall j \in [d]$, $z_j \geq 0$ and $p(z_i) = \frac{2}{\sqrt{2\pi}} e^{-z_j^2/2}$. Let $\boldsymbol{x} = \boldsymbol{z}^2$, where the square is taken elementwise. Derive the probability density function for $\boldsymbol{x}$. $(0.5/15)$