[redacted]

4 Bias-Variance for Ridge Regression (24 points)

Consider the scalar data-generation model:

$$Y = xw^* + Z$$

where x denotes the scalar input feature, Y denotes the scalar noisy measurement, $Z \sim \mathcal{N}(0,1)$ is standard unit-variance zero-mean Gaussian noise, and w^* denotes the true generating parameter that we would like to estimate.

We are given a set of n training samples $\{x_i, y_i\}_{i=1}^n$ that are generated by the above model with i.i.d. Z_i and distinct x_i. Our goal is to fit a linear model and get an estimate $\widehat{w}$ for the true parameter w^*. For all parts, assume that x_i's are given and fixed (not random).

For a given training set $\{x_i, y_i\}_{i=1}^n$, the ridge-regression estimate for w^* is defined by

$$\widehat{w}_\lambda = \arg\min_{w \in \mathbb{R}} \lambda w^2 + \sum_{i=1}^n (y_i - x_i w)^2 \qquad \text{with } \lambda \geq 0.$$

For the rest of the problem, assume that this has been solved and written in the form:

$$\widehat{w}_\lambda = \frac{S_{xy}}{s_x^2 + \lambda} \tag{2}$$

(a) (8 pts) **Compute the squared-bias of the ridge estimate $\widehat{w}_\lambda$ defined as follows**

$$\text{Bias}^2(\widehat{w}_\lambda) = (\mathbb{E}[\widehat{w}_\lambda] - w^*)^2. \tag{3}$$

It is fine if your answer depends on w^* or s_x, but it should not depend directly or indirectly on the realizations of the random Z noise. (So, no S_{xy} allowed.)

Hint: First compute the expectation of the estimate $\widehat{w}_\lambda$ over the noises Z in the observation.

Solution: To compute the expectation, we note that 1) the ridge-estimate is random because the observation Y is random, and 2) that $E[Y_i] = E[x_i w^* + Z_i] = x_i w^*$. Using these two facts and the linearity of expectation, we have

$$\begin{aligned} E[\widehat{w}_\lambda] = E\left[\frac{\sum_{i=1}^n x_i Y_i}{\sum_{i=1}^n x_i^2 + \lambda}\right] &= \frac{1}{\sum_{i=1}^n x_i^2 + \lambda}\left[\sum_{i=1}^n E[x_i Y_i]\right] \\ &= \frac{1}{\sum_{i=1}^n x_i^2 + \lambda}\left[\sum_{i=1}^n x_i E[Y_i]\right] \\ &= \frac{1}{\sum_{i=1}^n x_i^2 + \lambda}\left[\sum_{i=1}^n x_i^2 w^*\right] \\ &= w^*\left(\frac{\sum_{i=1}^n x_i^2}{\sum_{i=1}^n x_i^2 + \lambda}\right). \end{aligned}$$

We now compute the squared-bias as follows:

$$\begin{aligned} (E[\widehat{w}_\lambda - w^*])^2 &= \left(w^*\left(\frac{\sum_{i=1}^n x_i^2}{\sum_{i=1}^n x_i^2 + \lambda}\right) - w^*\right)^2 \\ &= (w^*)^2\frac{\lambda^2}{(\sum_{i=1}^n x_i^2 + \lambda)^2} \\ &= (w^*)^2\frac{\lambda^2}{(s_x^2 + \lambda)^2}. \end{aligned}$$

(b) (8 pts) **Compute the variance of the estimate $\widehat{w}_\lambda$ which is defined as**

$$\text{Var}(\widehat{w}_\lambda) = \mathbb{E}[(\widehat{w}_\lambda - \mathbb{E}[\widehat{w}_\lambda])^2]. \tag{4}$$

Hint: It might be useful to write $\widehat{w}_\lambda = \mathbb{E}[\widehat{w}_\lambda] + R$ for some random variable R.

Solution: We have

$$\begin{aligned}\widehat{w}_\lambda = \frac{\sum_{i=1}^n x_i Y_i}{\lambda + \sum_{i=1}^n x_i^2} &= \frac{\sum_{i=1}^n x_i^2 w^* + x_i Z_i}{\lambda + \sum_{i=1}^n x_i^2}\\ &= E[\widehat{w}_\lambda] + \underbrace{\frac{1}{\lambda + \sum_{i=1}^n x_i^2}\sum_{i=1}^n x_i Z_i}_{R},\end{aligned}$$

where R denotes the random variable defined in the hint.

Thus, we have

$$\begin{aligned}E(\widehat{w}_\lambda - E[\widehat{w}_\lambda])^2 &= \frac{1}{(\lambda + \sum_{i=1}^n x_i^2)^2} E(\sum_{i=1}^n x_i Z_i)^2\\ &= \frac{1}{(\lambda + \sum_{i=1}^n x_i^2)^2}\left(\sum_{i=1}^n x_i^2 E(Z_i^2) + \sum_{i \neq j} x_i x_j E[Z_i Z_j]\right)\\ &= \frac{\sum_{i=1}^n x_i^2}{(\lambda + \sum_{i=1}^n x_i^2)^2}\\ &= \frac{s_x^2}{(\lambda + s_x^2)^2}.\end{aligned}$$

since $E(Z_i^2) = 1$ and $E(Z_i Z_j) = 0$ for $i \neq j$ as Z_i is independent of Z_j.

(c) (8 pts) **Describe how the squared-bias and variance of the estimate $\widehat{w}_\lambda$ change as we change the value of λ. What happens as $\lambda \to 0$? $\lambda \to \infty$? Is the bias increasing or decreasing? Is the variance increasing or decreasing? In what sense is there a bias/variance tradeoff?**

Solution: We have

$$\begin{aligned}\text{Bias}^2(\widehat{w}_\lambda) &= (w^*)^2\frac{\lambda^2}{(\sum_{i=1}^n x_i^2+\lambda)^2} = (w^*)^2\frac{1}{(s_x^2/\lambda+1)^2}\\ \text{Var}(\widehat{w}_\lambda) &= \frac{\sum_{i=1}^n x_i^2}{(\lambda+\sum_{i=1}^n x_i^2)^2} = \frac{s_x^2}{(\lambda+s_x^2)^2}\end{aligned}$$

and thus clearly squared-bias increases with increase in λ and takes value 0 at 0. In fact, $\lambda = 0$ corresponds to the OLS case and you may recall that OLS for linear models is unbiased. (Students are not expected to make this observation in the exam, but it is stated here as a take away message.) Furthermore, for $\lambda \to \infty$, we have that bias-squared $\to (w^*)^2$. We can directly observe this fact, since $\lambda \to \infty$ implies that the ridge estimate would be 0 and hence the bias would be w^* and consequently the bias-squared would be $(w^*)^2$.

For the variance, we see that increasing λ reduces variance. This is intuitively correct too, because larger penalty forces the weight to shrink towards zero thereby reducing its scale and hence the variance too!

Thus, we see that a larger penalty in ridge-regression increases the squared-bias for the estimate and reduces the variance, and thus we observe a trade-off.

5 Hospital (25 points)

You work at hospital A. Your hospital has collected patient data to build a model to predict who is likely to get sepsis (a bad outcome). Specifically, the data set contains the feature matrix $\mathbf{X} \in \mathbb{R}^{n\times d}$, and associated real number labels $\mathbf{y} \in \mathbb{R}^n$, where n is the number of patients you are learning from and d is the number of features describing each patient. You plan to fit a linear regression model $\widehat{y} = \mathbf{w}^\top\mathbf{x}$ that will enable you to predict a label for future, unseen patients (using their feature vectors).

However, your hospital has only started collecting data a short time ago. Consequently the model you fit is not likely to be particularly accurate. Hospital B has exactly the same set up as your hospital (i.e., their patients are drawn from the same distribution as yours and they have the same measurement tools). For privacy reasons, Hospital B will not share their data. However, they tell you that they have trained a linear model on their own sepsis-relevant data: ($\mathbf{X}_B$ and $\mathbf{y}_B$) and are willing to share their learned model $\widehat{y} = \widehat{\mathbf{w}}_B^\top\mathbf{x}$ with you. In particular, Hospital B shares their entire Gaussian posterior distribution on $\mathbf{w}$ with you: $\mathcal{N}(\widehat{\mathbf{w}}_B, \mathbf{\Psi})$.

(a) (10 pts) Assume that we use the posterior from Hospital B as our own prior distribution for $\mathbf{w} \sim \mathcal{N}(\widehat{\mathbf{w}}_B, \mathbf{\Psi})$. Suppose that our Hospital A model is given by $\mathbf{y} = \mathbf{X}\mathbf{w} + \boldsymbol{\epsilon}$, where the noise, $\boldsymbol{\epsilon}$, has an assumed distribution $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathbf{I})$. **Derive the MAP estimate $\widehat{\mathbf{w}}$ for $\mathbf{w}$ using Hospital A's data $\mathbf{X}, \mathbf{y}$ and the prior information from Hospital B.**

HINT: Recall that traditional ridge regression could be derived from a MAP perspective, where the parameter $\mathbf{w}$ *has a zero mean Gaussian prior distribution with a scaled identity covariance. How could you use reparameterization (i.e. change of variables) for the problem here?*

Solution:

The overall point of the Hospital problem was to highlight the equivalence between a prior distribution on parameter(s) and "pseudo" data. Stating a prior can in general be reformulated as pretending to have seen some made up data, often referred to as pseudo-data. This point was highlighted in homework. In general, the tighter the prior distribution (e.g. the less variance), the more sure we are that it is "correct". Thus it also corresponds to having seen more "pseudo" data.

In part (a) the question is really just asking you to crank through the MAP estimate. But the pedagogical insight was to appreciate that in the hospital setting, the "pseudo" data actually corresponded to real data and helped us get around a privacy problem! One could derive it from "scratch" as done in lecture, with the specific prior given in the question, as follows (but there is a much easier solution we go over next):

From first principles solution:

Let D be the dataset, i.e. $\mathbf{X}$ and $\mathbf{y}$. According to the Bayes' Rule and ignoring the terms not related to $\mathbf{w}$, we have:

$$\begin{aligned}\log p(\mathbf{w}|D) &\propto \log p(D|\mathbf{w}) + \log p(\mathbf{w}) \\ &\propto -(\mathbf{X}\mathbf{w}-\mathbf{y})^\top(\mathbf{X}\mathbf{w}-\mathbf{y}) - (\mathbf{w}-\mathbf{w}_B)^\top\mathbf{\Psi}_B^{-1}(\mathbf{w}-\mathbf{w}_B) \\ &\propto (-\mathbf{w}^\top\mathbf{X}^\top\mathbf{X}\mathbf{w} + 2\mathbf{y}^\top\mathbf{X}\mathbf{w}) - (\mathbf{w}^\top\mathbf{\Psi}_B^{-1}\mathbf{w} - 2\mathbf{w}_B{}^\top\mathbf{\Psi}_B^{-1}\mathbf{w}) \\ &\propto -\mathbf{w}^\top(\mathbf{X}^\top\mathbf{X} + \mathbf{\Psi}_B^{-1})\mathbf{w} + 2(\mathbf{y}^\top\mathbf{X} + \mathbf{w}_B{}^\top\mathbf{\Psi}_B^{-1})\mathbf{w}\end{aligned} \tag{5}$$

Setting the derivative of $\log p(\mathbf{w}|D)$ to zero and solve the equation, we get:
$\hat{\mathbf{w}} = (\mathbf{X}^\top\mathbf{X} + \mathbf{\Psi}_B^{-1})^{-1}(\mathbf{X}^\top\mathbf{y} + \mathbf{\Psi}_B^{-1}\mathbf{w}_B)$

However, an easier way to solve this question was to simply do a change of variables on w such that its prior became zero-mean and we could just read off the solution derived in lecture for MAP with arbitrary prior variance. In particular let $\mathbf{v} = \mathbf{w} - \mathbf{w}_b$ (i.e. $\mathbf{w} = \mathbf{v} + \mathbf{w} =$), then we have

$$\begin{aligned}\log p(\mathbf{w}|D) &\propto \log p(D|\mathbf{w}) + \log p(\mathbf{w}) \\ &= \log N(\mathbf{y}; \mathbf{X}(\mathbf{v}+\mathbf{w}_B), \mathbf{I}) + \log N(\mathbf{v}; 0, \Psi)\end{aligned} \tag{6}$$

From which we read off that $\hat{w} = w_B + (X^TX + \Psi^{-1})^{-1}X^T(y - Xw_b)$, appealing to the formula we derived in class and appearing in the notes.

A third solution would have been to use the change of variables

$$\mathbf{w} = \Psi^{1/2}\mathbf{v} + \mathbf{w}_B$$

to reduce to standard ridge with $\lambda = 1$ and read the solution off directly as done in lecture–see the lecture notes only without the $\mathbf{w}_B$ offset term.

(b) (15 pts) Now, for simplicity, consider $d = 1$ so that the w is a scalar parameter. Suppose that instead of giving you their posterior distribution, Hospital B only gave you their mean $\widehat{w}_B$. How can you use this information to help fit your model? **Describe in detail how you should use your own hospital's patient data and combine it with the mean $\widehat{w}_B$ from Hospital B in a procedure to find your own $\widehat{w}$ for predicting sepsis in Hospital A.**

Hint 1: You might want to consider introducing an appropriate hyperparameter and doing what you usually do with hyperparameters.

Hint 2: What does the λ hyperparameter in ridge-regression correspond to from a probabilistic perspective?

Solution:

This problem was asking students to remember that an unknown variance term in a prior corresponds to a hyperparameter. After all λ in traditional ridge regression corresponded to the reciprocal of the prior variance. This suggests that we should introduce a hyperparameter ψ into the problem for the unknown variance. Once we have done this, we can do what we always do with hyperparameters: use cross-validation to pick an appropriate value for them. Putting these ideas together, we get the solution:

(a) Divide the data $\mathbf{X}$ and $\mathbf{y}$ from Hospital A into k folds. (For example, we could even have $k = n$ here.)

(b) For different potential values of ψ do the following:

   i. Repeat over the k folds:

      A. Let the training data $\mathbf{X}_{train}$ be all of $\mathbf{X}$ except the j-th fold. Similarly, let the training data $\mathbf{y}_{train}$ be all of $\mathbf{y}$ except the j-th fold.

      B. Use the modified ridge regression of Part (a) using the current potential value for the hyperparameter ψ to fit a candidate $\widehat{w}_\psi$ using $\mathbf{X}_{train}$ and $\mathbf{y}_{train}$.

      C. Evaluate the validation error on the validation data representing the j-th fold of $\mathbf{X}$ and $\mathbf{y}$.

   ii. Average together the validation error (squared errors) for the different fold to get an average validation error estimate for this particular value for the hyperparameter ψ.

(c) Choose the hyperparameter value ψ with the lowest average validation error.

(d) Retrain the model using the chosen value for ψ on the entire training data $\mathbf{X}$ and $\mathbf{y}$ to get our final $\widehat{w}$ for hospital A.

We also would accept other ways of using a hyperparameter. For example, realizing that the effect of the hyperparameter in the scalar case is simply to "shrink" the OLS estimate towards $\widehat{w}_B$, it would also be reasonable to directly define the amount of interpolation between the two as the hyperparameter. The important thing is that the validation data for evaluating hyperparameters be chosen in a way that is not cross-contaminated with data being used to set the parameters themselves. So, it is still vital that the OLS estimate be done with different data and cross-validation be used.

Given that the context here is data poor, it does make sense to use k-fold cross-validation instead of simply splitting the data into a training and validation set once. However on the exam, no points were deducted for just using a simple validation approach.

6 Ridge regression vs. PCA (24 points)

Assume we are given n training data points $(\mathbf{x}_i, y_i)$. We collect the target values into $\mathbf{y} \in \mathbb{R}^n$, and the inputs into the matrix $\mathbf{X} \in \mathbb{R}^{n\times d}$ where the rows are the $d-$dimensional feature vectors $\mathbf{x}_i^\top$ corresponding to each training point. Furthermore, assume that $\frac{1}{n}\sum_{i=1}^n \mathbf{x_i} = \mathbf{0}$, $n > d$ and $\mathbf{X}$ has rank d.

In this problem we want to compare two procedures: The first is ridge regression with hyperparameter λ, while the second is applying ordinary least squares after using PCA to reduce the feature dimension from d to k (we give this latter approach the short-hand name k-PCA-OLS where k is the hyperparameter).

Notation: The singular value decomposition of $\mathbf{X}$ reads $\mathbf{X} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top$ where $\mathbf{U} \in \mathbb{R}^{n\times n}$, $\mathbf{\Sigma} \in \mathbb{R}^{n\times d}$ and $\mathbf{V} \in \mathbb{R}^{d\times d}$. We denote by $\mathbf{u}_i$ the n-dimensional column vectors of $\mathbf{U}$ and by $\mathbf{v}_i$ the $d-$dimensional column vectors of $\mathbf{V}$. Furthermore the diagonal entries $\sigma_i = \Sigma_{i,i}$ of $\mathbf{\Sigma}$ satisfy $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_d > 0$. For notational convenience, assume that $\sigma_i = 0$ for $i > d$.

(a) (6 pts) It turns out that the ridge regression optimizer (with $\lambda > 0$) in the $\mathbf{V}$-transformed coordinates

$$\widehat{\mathbf{w}}_{\text{ridge}} = \arg\min_{\mathbf{w}} \|\mathbf{X}\mathbf{V}\mathbf{w} - \mathbf{y}\|_2^2 + \lambda\|\mathbf{w}\|_2^2$$

has the following expression:

$$\widehat{\mathbf{w}}_{\text{ridge}} = \text{diag}(\frac{\sigma_i}{\lambda + \sigma_i^2})\mathbf{U}^\top\mathbf{y}. \tag{7}$$

Use $\widehat{y}_{test} = \mathbf{x}_{test}^\top \mathbf{V}\widehat{\mathbf{w}}_{\text{ridge}}$ to denote the resulting prediction for a hypothetical $\mathbf{x}_{test}$. Using (7) and the appropriate scalar $\{\beta_i\}$, this can be written as:

$$\widehat{y}_{test} = \mathbf{x}_{test}^\top \sum_{i=1}^d \mathbf{v}_i\beta_i\mathbf{u}_i^\top\mathbf{y}. \tag{8}$$

What are the $\beta_i \in \mathbb{R}$ for this to correspond to (7) from ridge regression?

Solution:

The resulting prediction for ridge reads

$$\hat{\mathbf{y}}_{\text{ridge}} = \mathbf{x}^\top\mathbf{V}\,\text{diag}\,(\frac{\sigma_i}{\lambda + \sigma_i^2})\mathbf{U}^\top\mathbf{y}$$
$$= \mathbf{x}^\top \sum_{i=1}^d \frac{\sigma_i}{\lambda + \sigma_i^2}\mathbf{v}_i\mathbf{u}_i^\top\mathbf{y}$$

Therefore we have $\beta_i = \frac{\sigma_i}{\lambda + \sigma_i^2}$ for $i = 1, \ldots, d$.

(b) (12 pts) Suppose that we do k-PCA-OLS — i.e. ordinary least squares on the reduced k-dimensional feature space obtained by projecting the raw feature vectors onto the $k < d$ principal components of the covariance matrix $\mathbf{X}^\top\mathbf{X}$. Use $\widehat{y}_{test}$ to denote the resulting prediction for a hypothetical $\mathbf{x}_{test}$,

It turns out that the learned k-PCA-OLS predictor can be written as:

$$\widehat{y}_{test} = \mathbf{x}_{test}^\top \sum_{i=1}^{d} \mathbf{v}_i \beta_i \mathbf{u}_i^\top \mathbf{y}. \tag{9}$$

Give the $\beta_i \in \mathbb{R}$ coefficients for k-PCA-OLS. Show work.

Hint 1: some of these β_i will be zero. Also, if you want to use the compact form of the SVD, feel free to do so if that speeds up your derivation.

Hint 2: some inspiration may be possible by looking at the next part for an implicit clue as to what the answer might be.

Solution: The OLS on the k-PCA-reduced features reads

$$\min_{\mathbf{w}} \|\mathbf{X}\mathbf{V}_k\mathbf{w} - \mathbf{y}\|_2^2$$

where the columns of $\mathbf{V}_k$ consist of the first k eigenvectors of $\mathbf{X}$.

In the following we use the compact form SVD, that is note that one can write

$$\begin{aligned}\mathbf{X} &= \mathbf{U}\mathbf{\Sigma}\mathbf{V} \\ &= \mathbf{U}_d\mathbf{\Sigma}_d\mathbf{V}\end{aligned}$$

where $\mathbf{\Sigma}_d = \mathrm{diag}(\sigma_i)$ for $i = 1, \ldots, d$ and $\mathbf{U}_d$ are the first d columns of $\mathbf{U}$. In general we use the notation $\mathbf{\Sigma}_k = \mathrm{diag}(\sigma_i)$ for $i = 1, \ldots, k$.

Apply OLS on the new matrix $\mathbf{X}\mathbf{V}_k$ to obtain

$$\begin{aligned}\widehat{\mathbf{w}}_{\mathrm{PCA}} &= [(\mathbf{X}\mathbf{V}_k)^\top(\mathbf{X}\mathbf{V}_k)]^{-1}(\mathbf{X}\mathbf{V}_k)^\top\mathbf{y} \\ &= [\mathbf{V}_k^\top\mathbf{V}\mathbf{\Sigma}_d^2\mathbf{V}^\top\mathbf{V}_k]^{-1}\mathbf{V}_k^\top\mathbf{X}^\top\mathbf{y} \\ &= \mathbf{\Sigma}_k^{-1}\mathbf{U}_k^\top\mathbf{y} = \widetilde{\mathbf{\Sigma}}_k^{-1}\mathbf{U}^\top\mathbf{y}\end{aligned}$$

where $\widetilde{\mathbf{\Sigma}}_k = \begin{pmatrix}\mathbf{\Sigma}_k & 0\end{pmatrix}$

The resulting prediction for PCA reads (note that you need to project it first!)

$$\begin{aligned}\widehat{\mathbf{y}}_{\mathrm{PCA}} &= \mathbf{x}^\top\mathbf{V}_k\widehat{\mathbf{w}}_{\mathrm{PCA}} \\ &= \mathbf{x}^\top\mathbf{V}_k\mathbf{\Sigma}_k^{-1}\mathbf{U}_k^\top\mathbf{y} \\ &= \mathbf{x}^\top\sum_{i=1}^{k}\frac{1}{\sigma_i}\mathbf{v}_i\mathbf{u}_i^\top\mathbf{y}\end{aligned}$$

and hence $\beta_i = \frac{1}{\sigma_i}$ if $i \leq k$ and $\beta_i = 0$ for $i = k+1, \ldots, d$.

(c) (6 pts) For the following part, $d = 5$. The following $\boldsymbol{\beta} := (\beta_1, \ldots, \beta_5)$ (written out to two significant figures) are the results of OLS (i.e. what we would get from ridge regression in the limit $\lambda \to 0$), λ-ridge-regression, and k-PCA-OLS for some $\mathbf{X}, \mathbf{y}$ (identical for each method) and $\lambda = 1, k = 3$. **Write down which procedure was used for each of the three sub-parts below.**

We hope this helps you intuitively see the connection between these three methods.

Hint: It is not necessary to find the singular values of $\mathbf{X}$ *explicitly, or to do any numerical computations at all.*

(i) $\boldsymbol{\beta} = (0.01, 0.1, 0.5, 0.1, 0.01)$

(ii) $\boldsymbol{\beta} = (0.01, 0.1, 1, 0, 0)$

(iii) $\boldsymbol{\beta} = (0.01, 0.1, 1, 10, 100)$

Solution: Ridge, 3-PCA-OLS, OLS.

Reasoning: The prediction for OLS is the same as for PCA with $k = d$.

$$\hat{\mathbf{y}}_{OLS} = \mathbf{x}^\top \sum_{i=1}^{d} \frac{1}{\sigma_i} \mathbf{v}_i \mathbf{u}_i^\top \mathbf{y}$$

Putting all pieces together, we can thus see that PCA does "hard shrinkage" or "hard cutoff" (i.e. sets to zero) of the last $k+1, \ldots, d$ coefficients β_i, whereas ridge regression does "soft shrinkage" (i.e. shrinks towards zero) of the coefficients.

7 Kernel PCA (24 points)

In lectures, discussion, and homework, we learned how to use PCA to do dimensionality reduction by projecting the data to a subspace that captures most of the variability. This works well for data that is roughly Gaussian shaped, but many real-world high dimensional datasets have underlying low-dimensional structure that is not well captured by linear subspaces. However, when we lift the raw data into a higher-dimensional feature space by means of a nonlinear transformation, the underlying low-dimensional structure once again can manifest as an approximate subspace. Linear dimensionality reduction can then proceed. As we have seen in class so far, kernels are an alternate way to deal with these kinds of nonlinear patterns without having to explicitly deal with the augmented feature space. This problem asks you to discover how to apply the "kernel trick" to PCA.

Let $\mathbf{X} \in \mathbb{R}^{n\times\ell}$ be the data matrix, where n is the number of samples and ℓ is the dimension of the raw data. Namely, the data matrix contains the data points $\mathbf{x}_j \in \mathbb{R}^\ell$ as rows

$$\mathbf{X} = \begin{pmatrix} \mathbf{x}_1^\top \\ \mathbf{x}_2^\top \\ \vdots \\ \mathbf{x}_n^\top \end{pmatrix} \in \mathbb{R}^{n\times\ell}. \tag{10}$$

(a) (5 pts) **Compute $\mathbf{X}\mathbf{X}^\top$ in terms of the singular value decomposition $\mathbf{X} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top$ where $\mathbf{U} \in \mathbb{R}^{n\times n}$, $\mathbf{\Sigma} \in \mathbb{R}^{n\times\ell}$ and $\mathbf{V} \in \mathbb{R}^{\ell\times\ell}$.** Notice that $\mathbf{X}\mathbf{X}^\top$ is the matrix of pairwise Euclidean inner products for the data points. **How would you get $\mathbf{U}$ if you only had access to $\mathbf{X}\mathbf{X}^\top$?**

Solution: By plugging in the compact SVD decomposition $\mathbf{X} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top$ and using $\mathbf{U}^\top\mathbf{U} = \mathbf{I}$ we get

$$\mathbf{X}^\top\mathbf{X} = \mathbf{V}\mathbf{\Sigma}\mathbf{U}^\top\mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top = \mathbf{V}\mathbf{\Sigma}^2\mathbf{V}^\top.$$

Similarly with $\mathbf{V}^\top\mathbf{V} = \mathbf{I}$ we get

$$\mathbf{X}\mathbf{X}^\top = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top\mathbf{V}\mathbf{\Sigma}\mathbf{U}^\top = \mathbf{U}\mathbf{\Sigma}^2\mathbf{U}^\top.$$

Notice from the last line that $\mathbf{U}$ are the eigenvectors of $\mathbf{X}\mathbf{X}^\top$ with eigenvalues $\sigma_1^2, \sigma_2^2, \ldots, \sigma_\ell^2$ where $\sigma_1, \sigma_2, \ldots, \sigma_d$ are the singular values of $\mathbf{X}$ and can therefore be computed by performing an eigendecomposition of $\mathbf{X}\mathbf{X}^\top$.

(b) (7 pts) Given a new test point $\mathbf{x}_{test} \in \mathbb{R}^{\ell}$, one central use of PCA is to compute the projection of $\mathbf{x}_{test}$ onto the subspace spanned by the k top singular vectors $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_k$.

Express the scalar projection $z_j = \mathbf{v}_j^\top \mathbf{x}_{test}$ onto the j-th principal component as a function of the inner products

$$\mathbf{X}\mathbf{x}_{test} = \begin{pmatrix} \langle \mathbf{x}_1, \mathbf{x}_{test} \rangle \\ \vdots \\ \langle \mathbf{x}_n, \mathbf{x}_{test} \rangle \end{pmatrix}. \tag{11}$$

Assume that all diagonal entries of $\mathbf{\Sigma}$ are nonzero and non-increasing, that is $\sigma_1 \geq \sigma_2 \geq \cdots > 0$.

Hint: Express $\mathbf{V}^\top$ in terms of the singular values $\mathbf{\Sigma}$, the left singular vectors $\mathbf{U}$ and the data matrix $\mathbf{X}$. If you want to use the compact form of the SVD, feel free to do so.

Solution: By multiplying the compact SVD $\mathbf{X} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top$ on both sides with $\mathbf{U}^\top$, we get $\mathbf{U}^\top\mathbf{X} = \mathbf{\Sigma}\mathbf{V}^\top$ and multiplying both sides of the new equation with $\mathbf{\Sigma}^{-1}$, we obtain

$$\mathbf{V}^\top = \mathbf{\Sigma}^{-1}\mathbf{U}^\top\mathbf{X}.$$

Therefore we get

$$z_j = \mathbf{v}_j^\top \mathbf{x}_{test} = \frac{1}{\sigma_j}\mathbf{u}_j^\top \mathbf{X}\mathbf{x}_{test}$$

.

(c) (12 pts) How would you define kernelized PCA for a general kernel function $k(\mathbf{x}_i, \mathbf{x}_j)$ (to replace the Euclidean inner product $\langle \mathbf{x}_i, \mathbf{x}_j \rangle$)? For example, the RBF kernel $k(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{\delta^2}\right)$.

Describe this in terms of a procedure which takes as inputs the training data points $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n \in \mathbb{R}^\ell$ and the new test point $\mathbf{x}_{test} \in \mathbb{R}^\ell$, and outputs the analog of the previous part's z_j coordinate in the kernelized PCA setting. You should include how to compute U from the data, as well as how to compute the analog of $\mathbf{X}\mathbf{x}_{test}$ from the previous part.

Invoking the SVD or computing eigenvalues/eigenvectors is fine in your procedure, as long as it is clear what matrix is having its SVD or eigenvalues/eigenvectors computed. The kernel $k(\cdot, \cdot)$ can be used as a black-box function in your procedure as long as it is clear what arguments it is being given.

Solution: For kernelizing PCA, we replace inner products $\langle \mathbf{x}_i, \mathbf{x}_j \rangle$ with $k(\mathbf{x}_i, \mathbf{x}_j)$ and $\langle \mathbf{x}_i, \mathbf{x}_{test} \rangle$ with $k(\mathbf{x}_i, \mathbf{x}_{test})$, the procedure is then:

(a) Obtain the vectors $\mathbf{u}_j$ as eigenvectors from the eigendecomposition of the kernelized counterpart of the Gram matrix: $\mathbf{K} \in \mathbb{R}^{n \times n}$ with $\mathbf{K}_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$. The eigenvalues should be sorted in decreasing order. They are all non-negative real numbers because of the properties of kernels — the K matrix must be positive semi-definite.

(b) Kernelize the inner products $z_j = \frac{1}{\sigma_j}\mathbf{u}_j^\top \mathbf{X}\mathbf{x}_{test}$ from the previous part by using:

$$z_j = \frac{1}{\sigma_j}\mathbf{u}_j^\top \begin{pmatrix} k(\mathbf{x}_1, \mathbf{x}_{test}) \\ k(\mathbf{x}_2, \mathbf{x}_{test}) \\ \vdots \\ k(\mathbf{x}_n, \mathbf{x}_{test}) \end{pmatrix}, \tag{12}$$

where the σ_j are the square roots of the eigenvalues for the martix K above generated by using the kernel on all pairs of training points. Because these are non-negative real numbers, the square root is well defined.

8 Multiple Choice Questions (14 points)

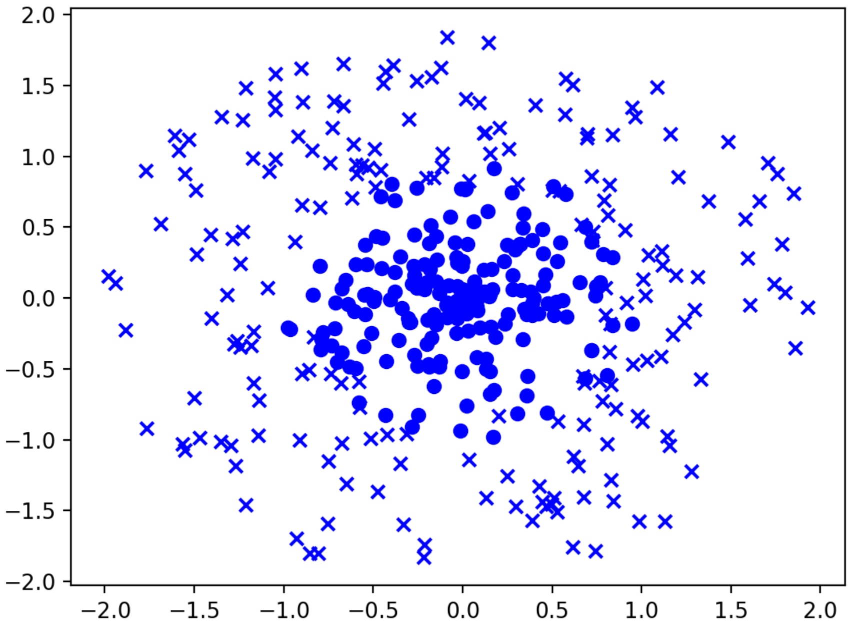
For these questions, select **all** the answers which are correct. You will get full credit for selecting all the right answers. On some questions, real-valued partial credit will be assigned. You will be graded on your **best seven of nine, so feel free to skip up to two of them.**

(a) Let $\mathbf{X} \in \mathbb{R}^{n \times d}$ with $n \geq d$. Suppose $\mathbf{X} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top$ is the singular value decomposition of $\mathbf{X}$ where $\sigma_i = \Sigma_{i,i}$ are the diagonal entries of $\mathbf{\Sigma}$ and satisfy $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_d$ while $\mathbf{u}_i$ and $\mathbf{v}_i$ are the ith columns of $\mathbf{U}$ and $\mathbf{V}$ respectively. **Which of the following is the rank k approximation to X that is best in the Froebenius norm.** That is, which low rank approximation, $\mathbf{X}_k$, for $\mathbf{X}$ yields the lowest value for $||\mathbf{X} - \mathbf{X}_k||_F^2$?

○ $\sum_{i=1}^{k} \sigma_i \mathbf{u}_i \mathbf{v}_{n-i}^\top$ ○ $\sum_{i=1}^{k} \sigma_i \mathbf{u}_i \mathbf{v}_i^\top$ ○ $\sum_{i=d-k+1}^{d} \sigma_i \mathbf{u}_i \mathbf{v}_i^\top$ ○ $\sum_{i=1}^{k} \sigma_i \mathbf{u}_{n-i} \mathbf{v}_i^\top$

Solution: The solution comes from the Eckhart-Young theorem which states we should use the singular vector expansion from 1 to k, $\sum_{i=1}^{k} \sigma_i \mathbf{u}_i \mathbf{v}_i^\top$

(b) Consider a simple dataset of points $(x_i, y_i) \in \mathbb{R}^2$, each associated with a label b_i which is -1 or $+1$. The dataset was generated by sampling data points with label -1 from a disk of radius 1.0 (shown as filled circles in the figure) and data points with label $+1$ from a ring with inner radius 0.8 and outer radius 2.0 (shown as crosses in the figure). **Which set of polynomial features would be best for performing linear regression, assuming at least as much data as shown in the figure?**



○ $1, x_i$

○ $1, x_i, y_i$

○ $1, x_i, y_i, x_i^2, x_i y_i, y_i^2$

○ $1, x_i, y_i, x_i^2, x_i y_i, y_i^2, x_i^3, y_i^3, x_i^2 y_i, x_i y_i^2$

Solution: Because the decision boundary was created by circles, and because there was enough data to justify modelling circles, we should use precisely order two polynomial features: $1, x_i^2, x_i y_i, y_i^2$. (Aside: If we had not specified that the points were created by circles, it could have been reasonable also add in possibly up to cubic polynomials, but this was considered incorrect here. Even if we had specified that the boundaries were created by circles, but had given only say two data points, then even a second order polynomial would not have made sense.)

(c) **Which of the following is a valid kernel function** for vectors of the same length, $\mathbf{x}$ and $\mathbf{y}$?

○ $k(\mathbf{x},\mathbf{y}) = \mathbf{x}^\top \mathbf{y}$

○ $k(\mathbf{x},\mathbf{y}) = e^{-\frac{1}{2}||\mathbf{x}-\mathbf{y}||_2^2}$

○ $k(\mathbf{x},\mathbf{y}) = (1+\mathbf{x}^\top \mathbf{y})^p$ for some degree p

○ $k(\mathbf{x},\mathbf{y}) = k_1(\mathbf{x},\mathbf{y}) - k_2(\mathbf{x},\mathbf{y})$ for valid kernels k_1 and k_2.

Solution: All except $k(\mathbf{x},\mathbf{y}) = k_1(x,y) - k_2(x,y)$ for valid kernels k_1 and k_2. This is not a valid kernel because valid kernel functions are not in general closed under subtraction, rather they are closed only under positive, linear combinations.

(d) During training of your model, both independent variables in the matrix $\mathbf{X} \in \mathbb{R}^{n \times d}$ and dependent target variables $\mathbf{y} \in \mathbb{R}^n$ are corrupted by noise. At test time, the data points you are computing predictions for, $\mathbf{x}_{test}$, are noiseless. **Which method(s) should you use to estimate the value of $\hat{\mathbf{w}}$ from the training data in order to make the most accurate predictions $\mathbf{y}_{test}$ from the noiseless test input data, $\mathbf{X}_{test}$?** Assume that you make predictions using $\mathbf{y}_{test} = \mathbf{X}_{test}\hat{\mathbf{w}}$.

○ OLS

○ Ridge regression

○ Weighted Least Squares

○ TLS

Solution: TLS since the test data has no noise while the training data does. This means that we are looking for the true relationship (as opposed to the best predictor for the training data) and total least squares is the one that does this.

Because TLS was shown to be equivalent to ridge-regression with a negative regularizer, points were not taken off for marking that as well.

(e) Assume you have n input data points, each with d high quality features ($\mathbf{X} \in \mathbb{R}^{n \times d}$) and associated labels ($\mathbf{y} \in \mathbb{R}^n$). Suppose that $d \gg n$ and that you want to learn a linear predictor. **Which of these approaches would help you to avoid overfitting?**

○ Preprocess $\mathbf{X}$ using $k \ll n$ random projections

○ Preprocess $\mathbf{X}$ using PCA with $k \ll n$ components.

○ Preprocess $\mathbf{X}$ using PCA with n components.

○ Add polynomial features

○ Use a kernel approach

○ Add a ridge penalty to OLS

○ Do weighted least squares

Solution: The goal here is simple dimensionality reduction for overfitting avoidance. As the earlier problem in the exam showed, ridge regression can be viewed as a softer form of k-PCA-OLS where the different dimensions are downweighted softly rather than as a strict truncation. So both PCA and ridge are clearly valid approaches to reduce overfitting.

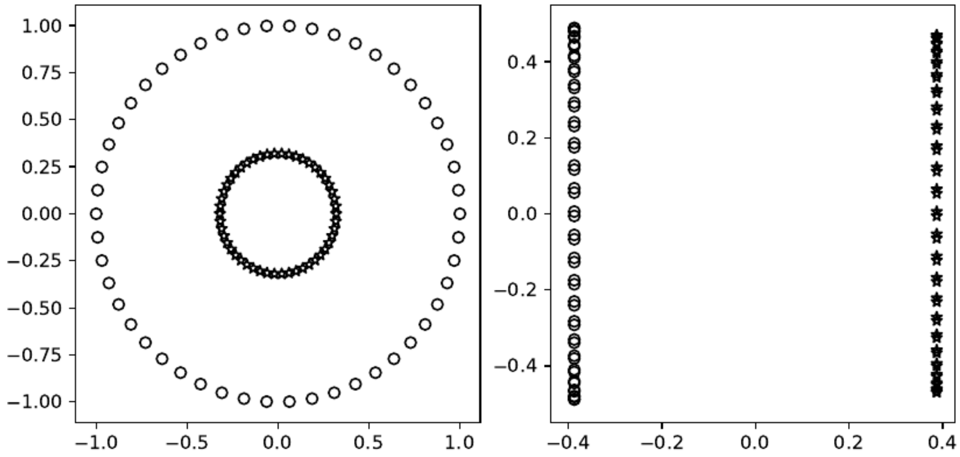
However, because the problem said that these are high-quality features, what you know about random projection also applies and so they too can be useful for dimensionality reduction.

The other answers are mostly wrong. If you use n components of PCA, there are still too many parameters relative to the data points. And so some overfitting will still occur.

Adding polynomial features makes the overfitting issue worse and not better, while weighing samples doesn't help us in any way.

We did not penalize for also marking kernel approaches since you know from lecture that the right kernel can also regularize because the kernel approach serves the same purpose as choosing features and priors together.

(f) **Which methods could yield a transformation to go from the two-dimensional data on the left to the two-dimensional data on the right?**



○ Random projections

○ PCA

○ Use of a kernel

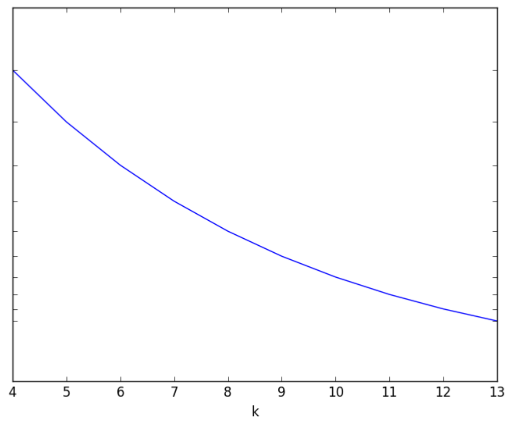
○ Adding polynomial features

Solution:

The plot here was literally obtained by doing RBF kernel-PCA on the data and choosing the two dominant components. PCA is clearly being used, but also something that allows us to get nonlinear relationships. Either the right kernel or polynomial features would suffice since circles are involved.

Random projections would not help here since they are linear.

(g) Your friend is training a machine learning model to predict disease severity based on k different health indicators. She generates the following plot, where the value of k is on the x axis.



Which of these might the y axis represent?

○ Training Error

○ Validation Error

○ Bias

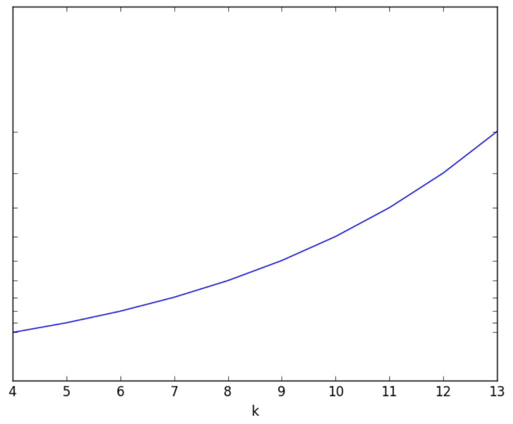
○ Variance

Solution: As k increases, the number of features in the ML model increases.

Full credit was given to anyone who responded **Training Error, Bias, Validation Error** or **Training Error, Bias**.

It is most important to recognize that training error and bias decrease with the number of features in the ML model. The reason that validation error is a correct answer is because it may be the case that the behavior changes after $k = 13$. Given this graph, it is not possible to know for sure.

(h) Your friend is training a machine learning model to predict disease severity based on k different health indicators. She generates the following plot, where the value of k is on the x axis.



Which of these might the y axis represent?

○ Training Error

○ Validation Error

○ Bias

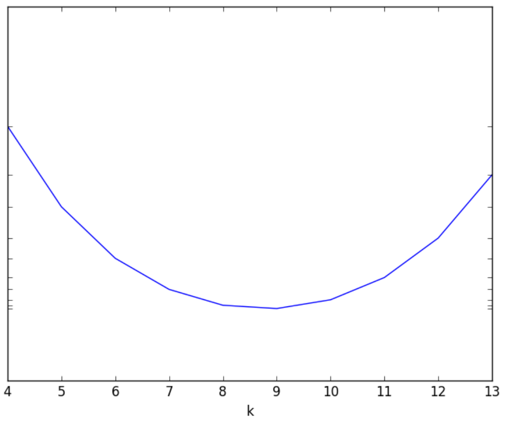
○ Variance

Solution: As k increases, the number of features in the ML model increases.

Full credit was given to anyone who responded **Variance, Validation Error** or **Variance**.

It is most important to recognize that variance increases with the number of features in the ML model. The reason that validation error is a correct answer is because it may be the case that the behavior changes before $k = 4$. Given this graph, it is not possible to know for sure.

(i) Your friend is training a machine learning model to predict disease severity based on k different health indicators. She generates the following plot, where the value of k is on the x axis.



Which of these might the y axis represent?

○ Training Error

○ Validation Error

○ Bias

○ Variance

Solution: As k increases, the number of features in the ML model increases.

Full credit was given to anyone who responded **Validation Error**.

[redacted]