ECE 6254, Spring 2022

Homework # 3

Due Sunday, February 18, at 11:59pm EST.

Suggested reading:

- *Elements of Statistical Learning* (by Hastie, Tibshirani, and Friedman): Section 3.2 (pages 44–56) discusses least squares regression; Section 3.4 (pages 61–79) covers ridge regression and the Lasso.
- *Learning from Data* (by Abu-Mostafa, Magdon-Ismail, Lin): Section 3.2 (pages 82–88) contains an alternative introduction to linear regression.

Problems:

1. In this problem, we will consider a simple learning scenario where $x \in \mathbb{R}$ and $y \in \mathbb{R}$ is given by $y = x^2$. In other words, $h^\star(x) = x^2$. Assume that the input variable x is an independent sample from a normal distribution with zero mean and variance 1. Now suppose that we are given two independent observations of input output pairs, i.e., our data set is given by $\mathcal{D} = \{(x_1, x_1^2), (x_2, x_2^2)\}$.

   (a) Suppose our hypothesis set consists of horizontal lines, i.e., $\mathcal{H} : h(x) = b$. We will fit the line by setting $h_{\mathcal{D}}(x) = \frac{y_1+y_2}{2} = \frac{x_1^2+x_2^2}{2}$. For this case, analytically derive the average hypothesis

   $$\bar{h}(x) = \mathbb{E}_{\mathcal{D}}[h_{\mathcal{D}}(x)] \tag{1}$$

   (b) Next, analytically compute the bias, i.e.,

   $$\mathbb{E}_X \left[\left(\bar{h}(X) - h^\star(X) \right)^2 \right] \tag{2}$$

   (c) Now, analytically compute the variance

   $$\mathbb{E}_X \left[\mathbb{E}_{\mathcal{D}} \left[\left(h_{\mathcal{D}}(X) - \bar{h}(X) \right)^2 \right] \right] \tag{3}$$

2. Suppose that we are given a dataset $\mathcal{D} = \{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_n, y_n)\}$ where each pair $(\boldsymbol{x}_i, y_i)$ is an independent realization of a random vector X in $\mathbb{R}^d$ and a random variable Y in $\mathbb{R}$ satisfying

   $$Y = X^\top \boldsymbol{\beta}^\star + N,$$

   where N represents noise that is independent of X and satisfies $\mathbb{E}[N] = 0$ and $\mathrm{var}(N) = \sigma^2$. In this scenario, $\boldsymbol{\beta}^\star \in \mathbb{R}^d$ represents the parameters of the function

   $$h^\star(\boldsymbol{x}) = \boldsymbol{x}^{\mathrm{T}} \boldsymbol{\beta}^\star$$

   that would minimize the expected squared error.

(a) Compute the "noise variance" term in the bias-variance decomposition, i.e., compute $R(h^\star) = \mathbb{E}[(Y - h^\star(X))^2]$?

(b) Now consider the function $h_{\mathcal{D}}$ formed from the least squares regression estimate. Specifically, if $\boldsymbol{X}$ is the $n \times d$ matrix whose i^{th} row is $\boldsymbol{x}_i^T$ and $\boldsymbol{y} \in \mathbb{R}^n$ contains $y_1, \ldots, y_n$, then

$$h_{\mathcal{D}}(\boldsymbol{x}) = \boldsymbol{x}^\top (\boldsymbol{X}^\top \boldsymbol{X})^{-1} \boldsymbol{X}^\top \boldsymbol{y}.$$

Compute the "average hypothesis"

$$\bar{h}(\boldsymbol{x}) = \mathbb{E}_{\mathcal{D}}[h_{\mathcal{D}}(\boldsymbol{x})] = \mathbb{E}_{\mathcal{D}}[\boldsymbol{x}^\top (\boldsymbol{X}^\top \boldsymbol{X})^{-1} \boldsymbol{X}^\top \boldsymbol{y}].$$

To do this, it is helpful to express $\boldsymbol{y}$ using $\boldsymbol{y} = \boldsymbol{X}\beta^\star + \boldsymbol{n}$, where $\boldsymbol{n} \in \mathbb{R}^n$. When taking the expectation with respect to $\mathcal{D}$, you should be thinking of both $\boldsymbol{X}$ and $\boldsymbol{n}$ as being random. (Hint: Remember that $\boldsymbol{X}$ and $\boldsymbol{n}$ are independent.)

(c) Using the result of the previous problem, compute the bias $\mathbb{E}_X[(\bar{h}(X) - h^\star(X))^2]$.

(d) Next we wish to compute the variance

$$\mathbb{E}_X \left[\mathbb{E}_{\mathcal{D}}[(h_{\mathcal{D}}(X) - \bar{h}(X))^2] \right].$$

First show that

$$\mathbb{E}_{\mathcal{D}}[(h_{\mathcal{D}}(X) - \bar{h}(X))^2] = \sigma^2 X^\top \mathbb{E}_{\mathcal{D}} \left[(\boldsymbol{X}^\top \boldsymbol{X})^{-1} \right] X.$$

Computing $\mathbb{E}_{\mathcal{D}} \left[(\boldsymbol{X}^\top \boldsymbol{X})^{-1} \right]$ is not straightforward, but note that $\boldsymbol{X}^\top \boldsymbol{X} = \sum_{i=1}^n \boldsymbol{x}_i \boldsymbol{x}_i^T$ is simply a scaled version of what looks like an empirical estimate of the matrix $\boldsymbol{C}_X = \mathbb{E}_X[XX^\top]$. Using the approximation $\boldsymbol{X}^\top \boldsymbol{X} \approx n\boldsymbol{C}_X$, compute an approximation for the variance. Your answer should only involve the dimensions d and n and the variance σ^2. (Hint: Write this in terms of the trace of a matrix and recall properties of the trace.)

(e) From the discussion in class, we have that

$$\mathbb{E}\left[R(h_{\mathcal{D}})\right] = \text{noise variance} + \text{bias} + \text{variance},$$

thus, from the problems above we can form a simple approximate estimate of the expected risk. This represents the expected error when applying our least squares regression estimate to a *new* sample $(\boldsymbol{x}, y)$ not seen in the training data. In this part we will contrast this with the expected error on the training data itself. Specifically, compute

$$\mathbb{E}\left[\widehat{R}_n(h_{\mathcal{D}})\right] = \mathbb{E}\left[\frac{1}{n}\|\boldsymbol{y} - \boldsymbol{X}(\boldsymbol{X}^\top \boldsymbol{X})^{-1} \boldsymbol{X}^\top \boldsymbol{y}\|_2^2\right].$$

Here, the expectation is with respect to the data, i.e., both $\boldsymbol{X}$ and $\boldsymbol{y}$ (or equivalently, $\boldsymbol{X}$ and $\boldsymbol{n}$). Your final answer should only involve the dimensions d and n and the variance σ^2. (Hint: One can show (you don't need to, but may want to try!) that for any matrix $\boldsymbol{P}$ that satisfies $\boldsymbol{P}^2 = \boldsymbol{P}$, $\text{trace}(\boldsymbol{P}) = \text{rank}(\boldsymbol{P})$. An implication of this fact is that the trace of a matrix that projects onto a subspace is equal to the dimension of the subspace.)

(f) Make a representative sketch of your approximation of $\mathbb{E}\left[R(h_{\mathcal{D}})\right]$ and what you computed for $\mathbb{E}\left[\widehat{R}_n(h_{\mathcal{D}})\right]$ in the previous problem as a function of the number of data points n. On the vertical axis, label the value σ^2.

3. Modern datasets often contain data points corrupted by *heteroscedastic* noise, meaning some of the collected data points are noisier than others. One scenario where this phenomena may occur is using sensor arrays of various quality to collect data; each sensor array produces a d-dimensional measurement, but sensor arrays composed of higher quality sensors may result in less noisy measurements compared to sensor arrays composed of cheaper sensors.

   If we know our measurements contain heteroscedastic noise, we should be able to take the fact that the noise level varies across samples into account when performing PCA. One way to combat this phenomena is by using *weighted PCA*. Given a set of data points $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n \in \mathbb{R}^d$, we want to solve

$$\underset{\boldsymbol{\mu}, \boldsymbol{X}, \{\boldsymbol{z}_i\}}{\text{minimize}} \sum_{i=1}^{n} \alpha_i \|\boldsymbol{x}_i - \boldsymbol{\mu} - \boldsymbol{X}\boldsymbol{z}_i\|_2^2 \qquad \text{subject to} \qquad \boldsymbol{X}^\top \boldsymbol{X} = \boldsymbol{I}, \tag{4}$$

   where $\{\alpha_i\}$ are a set of scalar weights that we get to choose. In this problem, we will derive the weighted PCA solution following the lecture notes.

   (a) Keeping $\boldsymbol{\mu}$ and $\boldsymbol{X}$ fixed, derive a closed form solution for $\{\widehat{\boldsymbol{z}}_i\}$.

   (b) Using the above expression for $\widehat{\boldsymbol{z}}_i$, find an expression for $\widehat{\boldsymbol{\mu}}$.

   (c) Now, assuming without loss of generality that $\boldsymbol{\mu} = \boldsymbol{0}$, derive an expression for $\widehat{\boldsymbol{X}}$.

   (d) Describe, briefly, the what each of two various weighting schemes is doing, and discuss any disadvantages:

   - **Binary weights:** For each data point $\boldsymbol{x}_i$, we pick $\alpha_i \in \{0, 1\}$.
   - **Inverse noise standard deviation weights:** If the variance of the noise corresponding to data point $\boldsymbol{x}_i$ is σ_i^2 (e.g., the noise follows distribution Normal$(0, \sigma_i^2 \boldsymbol{I})$), then we pick $\alpha_i = \frac{1}{\sigma_i}$.

4. In class we discussed ridge regression and the LASSO. Another form of regularized least squares regression involves the so-called *elastic-net* regularizer, which corresponds to the optimization problem

$$\min_{\boldsymbol{\theta}} \ \|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\theta}\|_2^2 + \lambda \left(\alpha \|\boldsymbol{\theta}\|_2^2 + (1-\alpha)\|\boldsymbol{\theta}\|_1\right),$$

   where both λ and α are scalar parameters set by the user. The elastic-net regularizer can be viewed as a compromise between the ℓ_2 and ℓ_1 penalties, being prone to both selecting variables (like the LASSO) and shrinking together the coefficients of correlated predictors (like ridge regression). Argue that another way to view the elastic-net optimization problem is as a LASSO optimization problem with an augmented version of $\boldsymbol{y}$ and $\boldsymbol{X}$. Specifically, show that you can write this in the form

$$\min_{\boldsymbol{\theta}} \ \|\widetilde{\boldsymbol{y}} - \widetilde{\boldsymbol{X}}\boldsymbol{\theta}\|_2^2 + \lambda \|\boldsymbol{\theta}\|_1.$$

5. In this problem we consider the scenario described in class where x is drawn uniformly on $[-1, 1]$ and $y = \sin(\pi x)$ and we are again given $n = 2$ training samples. Here we will consider an alternative approach to fitting a line to the data based on Tikhonov regularization. Specifically, we let

$$\boldsymbol{y} = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} \qquad \boldsymbol{X} = \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \end{bmatrix} \qquad \boldsymbol{\theta} = \begin{bmatrix} b \\ a \end{bmatrix}.$$

We will then consider Tikhonov regularized least squares estimators of the form

$$\widehat{\boldsymbol{\theta}} = (\boldsymbol{X}^\top \boldsymbol{X} + \boldsymbol{\Gamma}^\top \boldsymbol{\Gamma})^{-1} \boldsymbol{X}^\top \boldsymbol{y}. \tag{5}$$

(a) How should we set $\boldsymbol{\Gamma}$ to reduce this estimator to fitting a constant function (i.e., finding an $h(x)$ of the form $h(x) = b$)? (Hint: For the purposes of this problem, it is sufficient to set $\boldsymbol{\Gamma}$ in a way that just makes $a \approx 0$. To make $a = 0$ exactly requires setting $\boldsymbol{\Gamma}$ in a way that makes the matrix $\boldsymbol{X}^\top \boldsymbol{X} + \boldsymbol{\Gamma}^\top \boldsymbol{\Gamma}$ singular — but note that this does not mean that the regularized least-squares optimization problem cannot be solved, you must just use a different formula than the one given in (5).)

(b) How should we set $\boldsymbol{\Gamma}$ to reduce this estimator to fitting a line of the form $h(x) = ax + b$ that passes through the observed data points (x_1, y_1) and (x_2, y_2)?

(c) Use the same approach as in the previous problem to numerically estimate the bias and variance for (at least approximations of) both of these estimators, and confirm that your estimates correspond to the numbers I provided in class.

(d) Play around and see if you can find a matrix $\boldsymbol{\Gamma}$ that results in a smaller risk than either of the two approaches we discussed in class. Report the $\boldsymbol{\Gamma}$ that gives you the best results. (You can restrict your search to diagonal $\boldsymbol{\Gamma}$ to simplify this.)

6. In this problem we will compare the performance of traditional least squares, ridge regression, and the LASSO on a real-world dataset. We will use the "California Housing Prices" dataset which contains the median sale price of owner occupied homes in about 20,640 different neighborhoods in California, along with 8 features for each home that might be relevant. These features include factors such as measures of income, age of the house, number of rooms/bedrooms, etc. To get this dataset in Python, simply type

```
from sklearn.datasets import fetch_california_housing
california = fetch_california_housing()
X = california.data
y = california.target
```

In order to judge the quality of each approach, you should split the dataset into a training set and a testing set. The training set should consist of 1,000 observations, and you can use the remaining observations for testing.

Before training any of these algorithms, it is a good idea to "standardize" the data. By this, I mean that you should take each feature (i.e., each column of the matrix `X`) and subtract off its mean and divide by the standard deviation to make it zero mean and unit variance. Otherwise, the regularized methods will implicitly be placing bigger penalties on using features which

just happen to be scaled to have small variance. You should determine how to "standardize" your training data by appropriately shifting/scaling each feature *using only the training data*, and then apply this transformation to both the training data and the testing data so that your learned function can readily be applied to the test set.

For all parts of the problem below, I would like you to submit your code.

(a) First, I would like you to evaluate the performance of least squares. You should implement this yourself using the equation we derived in class. Report the performance of your algorithm in terms of mean-squared error on the test set, i.e.,

$$\frac{1}{n_{\text{test}}} \|\boldsymbol{y}_{\text{test}} - \boldsymbol{X}_{\text{test}} \widehat{\boldsymbol{\theta}}\|_2^2.$$

(b) Next, using the formula derived in class, implement your own version of ridge regression. You will need to set the free parameter λ. You should do this using the training data in whatever manner you like (e.g., via a holdout set) – but you should *not* allow the testing dataset to influence your choice of λ. Report the value of λ selected and the performance of your algorithm in terms of mean-squared error on the test set.

(c) Finally, I would like you to evaluate the performance of the LASSO. You do not need to implement this yourself. Instead, you can use scikit-learn's built in solver via

```
from sklearn import linear_model
reg = linear_model.Lasso(alpha = ???)
reg.fit(Xtrain,ytrain)
reg.predict(Xtest)
```

Above, `alpha` corresponds to the λ parameter from the lecture notes. As in part (b), you will need to do something principled to choose a good value for this parameter. Report the value of `alpha` used in your code, the performance of your algorithm in terms of mean-squared error, and the number of nonzeros in $\boldsymbol{\theta}$. (You can get $\boldsymbol{\theta}$ via `reg.coef_`.)