

# Homework #1

CSE 546: Machine Learning

Prof. Kevin Jamieson

Due: 10/18/18 11:59 PM

## 1 Gaussians

Recall that for any vector  $u \in \mathbb{R}^n$  we have  $\|u\|_2^2 = u^T u = \sum_{i=1}^n u_i^2$  and  $\|u\|_1 = \sum_{i=1}^n |u_i|$ . For a matrix  $A \in \mathbb{R}^{n \times n}$  we denote  $|A|$  as the determinant of  $A$ . A multivariate Gaussian with mean  $\mu \in \mathbb{R}^n$  and covariance  $\Sigma \in \mathbb{R}^{n \times n}$  has a probability density function  $p(x|\mu, \Sigma) = \frac{1}{\sqrt{(2\pi)^n |\Sigma|}} \exp(-\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu))$  which we denote as  $\mathcal{N}(\mu, \Sigma)$ .

1. [4 points] Let

- $\mu_1 = \begin{bmatrix} 1 \\ 2 \end{bmatrix}$  and  $\Sigma_1 = \begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix}$
- $\mu_2 = \begin{bmatrix} -1 \\ 1 \end{bmatrix}$  and  $\Sigma_2 = \begin{bmatrix} 2 & -1.8 \\ -1.8 & 2 \end{bmatrix}$
- $\mu_3 = \begin{bmatrix} 2 \\ -2 \end{bmatrix}$  and  $\Sigma_3 = \begin{bmatrix} 3 & 1 \\ 1 & 2 \end{bmatrix}$

For each  $i = 1, 2, 3$  on a separate plot:

- Draw  $n = 100$  points  $X_{i,1}, \dots, X_{i,n} \sim \mathcal{N}(\mu_i, \Sigma_i)$  and plot the points as a scatter plot with each point as a triangle marker (Hint: use `numpy.random.randn` to generate a mean-zero independent Gaussian vector, then use the properties of Gaussians to generate  $X$ ).
- Compute the sample mean and covariance matrices  $\hat{\mu}_i = \frac{1}{n} \sum_{j=1}^n X_{i,j}$  and  $\hat{\Sigma}_i = \frac{1}{n-1} \sum_{j=1}^n (X_{i,j} - \hat{\mu}_i)^2$ . Compute the eigenvectors of  $\hat{\Sigma}_i$ . Plot the eigenvectors as line segments originating from  $\hat{\mu}_i$  and have magnitude equal to the square root of their corresponding eigenvalues.
- If  $(u_{i,1}, \lambda_{i,1})$  and  $(u_{i,2}, \lambda_{i,2})$  are the eigenvector-eigenvalue pairs of the sample covariance matrix with  $\lambda_{i,1} \geq \lambda_{i,2}$  and  $\|u_{i,1}\|_2 = \|u_{i,2}\|_2 = 1$ , for  $j = 1, \dots, n$  let  $\tilde{X}_{i,j} = \begin{bmatrix} \frac{1}{\sqrt{\lambda_{i,1}}} u_{i,1}^T (X_{i,j} - \hat{\mu}_i) \\ \frac{1}{\sqrt{\lambda_{i,2}}} u_{i,2}^T (X_{i,j} - \hat{\mu}_i) \end{bmatrix}$ . Plot these new points as a scatter plot with each point as a circle marker.

For each plot, make sure the limits of the plot are square around the origin (e.g.,  $[-c, c] \times [-c, c]$  for some  $c > 0$ ).

## 2 MLE and Bias Variance Tradeoff

Recall that for any vector  $u \in \mathbb{R}^n$  we have  $\|u\|_2^2 = u^T u = \sum_{i=1}^n u_i^2$  and  $\|u\|_1 = \sum_{i=1}^n |u_i|$ . Unless otherwise specified, if  $P$  is a probability distribution and  $x_1, \dots, x_n \sim P$  then it can be assumed each  $x_i$  is drawn iid from  $P$ .

2. [1 points] Let  $x_1, \dots, x_n \sim \text{uniform}(0, \theta)$  for some  $\theta$ . What is the Maximum likelihood estimate for  $\theta$ ?

3. [2 points] Let  $(x_1, y_1), \dots, (x_n, y_n)$  be drawn at random from some population where each  $x_i \in \mathbb{R}^d$ ,  $y_i \in \mathbb{R}$ , and let  $\hat{w} = \arg \min_w \sum_{i=1}^n (y_i - w^T x_i)^2$ . Suppose we have some test data  $(\tilde{x}_1, \tilde{y}_1), \dots, (\tilde{x}_m, \tilde{y}_m)$  drawn at random

from the population as the training data. If  $R_{tr}(w) = \frac{1}{n} \sum_{i=1}^n (y_i - w^T x_i)^2$  and  $R_{te}(w) = \frac{1}{m} \sum_{i=1}^m (\tilde{y}_i - w^T \tilde{x}_i)^2$ . Prove that

$$\mathbb{E}[R_{tr}(\hat{w})] \leq \mathbb{E}[R_{te}(\hat{w})]$$

where the expectations are over all that is random in each expression. Do not assume any model for  $y_i$  given  $x_i$  (e.g., linear plus Gaussian noise). [This is exercise 2.9 from HTF, originally from Andrew Ng.]

4. [8 points] Let random vector  $X \in \mathbb{R}^d$  and random variable  $Y \in \mathbb{R}$  have a joint distribution  $P_{XY}(X, Y)$ . Assume  $\mathbb{E}[X] = 0$  and define  $\Sigma = \text{Cov}(X) = \mathbb{E}[(X - \mathbb{E}[X])(X - \mathbb{E}[X])^T]$  with eigenvalues  $\alpha_1 \geq \alpha_2 \geq \dots \geq \alpha_d$  and orthonormal eigenvectors  $v_1, \dots, v_d$  such that  $\Sigma = \sum_{i=1}^d \alpha_i v_i v_i^T$ . For  $(X, Y) \sim P_{XY}$  assume that  $Y = X^T w + \epsilon$  for  $\epsilon \sim \mathcal{N}(0, \sigma^2)$  such that  $\mathbb{E}_{Y|X}[Y|X = x] = x^T w$ . Let  $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^n$  where each  $(x_i, y_i) \sim P_{XY}$ . For some  $\lambda > 0$  let

$$\hat{w} = \arg \min_w \sum_{i=1}^n (y_i - x_i^T w)^2 + \lambda \|w\|_2^2$$

If  $\mathbf{X} = [x_1, \dots, x_n]^T$ ,  $\mathbf{y} = [y_1, \dots, y_n]^T$ ,  $\epsilon = [\epsilon_1, \dots, \epsilon_n]^T$  then it can be shown that

$$\hat{w} = (\mathbf{X}^T \mathbf{X} + \lambda I)^{-1} \mathbf{X}^T \mathbf{y}. \quad (1)$$

Note the notational difference between a random  $X$  of  $(X, Y) \sim P_{XY}$  and the  $n \times d$  matrix  $\mathbf{X}$  where each row is drawn from  $P_X$ . Realizing that  $\mathbf{X}^T \mathbf{X} = \sum_{i=1}^n x_i x_i^T$ , by the law of large numbers we have  $\frac{1}{n} \mathbf{X}^T \mathbf{X} \rightarrow \Sigma$  as  $n \rightarrow \infty$ . In your analysis assume  $n$  is large and make use of the approximation  $\mathbf{X}^T \mathbf{X} = n\Sigma$ . Justify all answers.

- Show Equation (??).
- Show that  $\hat{w}$  of Equation ?? can also be written as

$$\hat{w} = w - \lambda(\mathbf{X}^T \mathbf{X} + \lambda I)^{-1} w + (\mathbf{X}^T \mathbf{X} + \lambda I)^{-1} \mathbf{X}^T \epsilon$$

- For general  $\hat{f}_{\mathcal{D}}(x)$  and  $\eta(x) = \mathbb{E}_{Y|X}[Y|X = x]$ , we showed in class that the bias variance decomposition is stated as

$$\mathbb{E}_{XY, \mathcal{D}}[(Y - \hat{f}_{\mathcal{D}}(X))^2] = \mathbb{E}_X \left[ \mathbb{E}_{Y|X, \mathcal{D}}[(Y - \hat{f}_{\mathcal{D}}(X))^2 | X = x] \right]$$

where

$$\mathbb{E}_{Y|X, \mathcal{D}}[(Y - \hat{f}_{\mathcal{D}}(X))^2 | X = x] = \underbrace{\mathbb{E}_{Y|X}[(Y - \eta(x))^2 | X = x]}_{\text{Irreducible error}} + \underbrace{(\eta(x) - \mathbb{E}_{\mathcal{D}}[\hat{f}_{\mathcal{D}}(x)])^2}_{\text{Bias-squared}} + \underbrace{\mathbb{E}_{\mathcal{D}}[(\mathbb{E}_{\mathcal{D}}[\hat{f}_{\mathcal{D}}(x)] - \hat{f}_{\mathcal{D}}(x))^2]}_{\text{Variance}}.$$

In what follows, use our particular problem setting with  $\hat{f}_{\mathcal{D}}(x) = \hat{w}^T x$ .

Irreducible error: What is  $\mathbb{E}_X \left[ \mathbb{E}_{Y|X}[(Y - \eta(x))^2 | X = x] \right]$ ?

- Bias-squared: Use the approximation  $\mathbf{X}^T \mathbf{X} = n\Sigma$  to show that

$$\mathbb{E}_X [(\eta(X) - \mathbb{E}_{\mathcal{D}}[\hat{f}_{\mathcal{D}}(X)])^2] = \sum_{i=1}^d \frac{\lambda^2 (w_i^T v_i)^2 \alpha_i}{(n\alpha_i + \lambda)^2} \leq \max_{j=1, \dots, d} \frac{\lambda^2 \alpha_j \|w\|_2^2}{(n\alpha_j + \lambda)^2}$$

- Variance: Use the approximation  $\mathbf{X}^T \mathbf{X} = n\Sigma$  to show that

$$\mathbb{E}_X [\mathbb{E}_{\mathcal{D}}[(\mathbb{E}_{\mathcal{D}}[\hat{f}_{\mathcal{D}}(X)] - \hat{f}_{\mathcal{D}}(X))^2]] = \sum_{i=1}^d \frac{\sigma^2 \alpha_i^2 n}{(\alpha_i n + \lambda)^2} \leq \frac{d\sigma^2 \alpha_1^2 n}{(\alpha_1 n + \lambda)^2}$$

f. Assume  $\Sigma = \alpha_1 I$  for some  $\alpha_1 > 0$ . Show that for the approximation  $\mathbf{X}^T \mathbf{X} = n\Sigma$  we have

$$\mathbb{E}_{XY, \mathcal{D}}[(Y - \hat{f}_{\mathcal{D}}(X))^2] = \sigma^2 + \frac{\lambda^2 \alpha_1 \|w\|_2^2}{(\alpha_1 n + \lambda)^2} + \frac{d\sigma^2 \alpha_1^2 n}{(\alpha_1 n + \lambda)^2}$$

What is the  $\lambda^*$  that minimizes this expression? In a sentence each describe how varying each parameter (e.g.,  $\|w\|_2$ ,  $d$ ,  $\sigma^2$ ) affects the size of  $\lambda^*$  and if this makes intuitive sense. Plug this  $\lambda^*$  back into the expression and comment on how this result compares to the  $\lambda = 0$  solution. It may be helpful to use  $\frac{1}{2}(a+b) \leq \max\{a, b\} \leq a+b$  for any  $a, b > 0$  to simplify the expression.

g. Assume that  $\alpha_1 > \alpha_2 = \alpha_3 = \dots = \alpha_d$  and furthermore, that  $w/\|w\|_2 = v_1$ . Show that for the approximation  $\mathbf{X}^T \mathbf{X} = n\Sigma$  we have

$$\mathbb{E}_{XY, \mathcal{D}}[(Y - \hat{f}_{\mathcal{D}}(X))^2] = \sigma^2 + \frac{\lambda^2 \alpha_1 \|w\|_2^2}{(\alpha_1 n + \lambda)^2} + \frac{\sigma^2 n \alpha_1^2}{(\alpha_1 n + \lambda)^2} + \frac{\sigma^2 n \alpha_2^2 (d-1)}{(\alpha_2 n + \lambda)^2}$$

It can be shown that  $\lambda^* = \frac{\sigma^2 + \sigma^2 (d-1) \alpha_1 / \alpha_2}{\|w\|_2^2}$  approximately minimizes this expression. In a sentence each describe if this makes intuitive sense, comparing to the solution of the last problem.

h. As  $\lambda$  increases, how does the bias and variance terms behave?

### 3 Programming: Ridge Regression on MNIST

5. *[10 points]* In this problem we will implement a least squares classifier for the MNIST data set. The task is to classify handwritten images of numbers between 0 to 9.

You are **NOT** allowed to use any of the prebuilt classifiers in `sklearn`. Feel free to use any method from `numpy` or `scipy`. Remember: if you are inverting a matrix in your code, you are probably doing something wrong (Hint: look at `scipy.linalg.solve`).

Get the data from <https://pypi.python.org/pypi/python-mnist>.

Load the data as follows:

```
from mnist import MNIST

def load_dataset():
    mndata = MNIST('./data/')
    X_train, labels_train = map(np.array, mndata.load_training())
    X_test, labels_test = map(np.array, mndata.load_testing())
    X_train = X_train/255.0
    X_test = X_test/255.0
```

You can visualize a single example by reshaping it to its original  $28 \times 28$  image shape.

a. In this problem we will choose a linear classifier to minimize the least squares objective:

$$\widehat{W} = \operatorname{argmin}_{W \in \mathbb{R}^{d \times k}} \sum_{i=0}^n \|W^T x_i - y_i\|_2^2 + \lambda \|W\|_F^2$$

We adopt the notation where we have  $n$  data points in our training objective and each data point  $x_i \in \mathbb{R}^d$ .  $k$  denotes the number of classes which is in this case equal to 10. Note that  $\|W\|_F$  corresponds to the Frobenius norm of  $W$ , i.e.  $\|\operatorname{vec}(W)\|_2^2$ .

Derive a closed form for  $\widehat{W}$ .

- b. As a first step we need to choose the vectors  $y_i \in \mathbb{R}^k$  by converting the original labels (which are in  $\{0, \dots, 9\}$ ) to vectors. We will use the one-hot encoding of the labels, i.e. the original label  $j \in \{0, \dots, 9\}$  is mapped to the standard basis vector  $e_j$ . To classify a point  $x_i$  we will use the rule  $\arg \max_{j=0, \dots, 9} \widehat{W}^T x_i$ .
- c. Code up a function called `train` that returns  $\widehat{W}$  that takes as input  $X \in \mathbb{R}^{n \times d}$ ,  $y \in \{0, 1\}^{n \times k}$ , and  $\lambda > 0$ . Code up a function called `predict` that takes as input  $W \in \mathbb{R}^{d \times k}$ ,  $X' \in \mathbb{R}^{m \times d}$  and returns an  $m$ -length vector with the  $i$ th entry equal to  $\arg \max_{j=0, \dots, 9} W^T x'_i$  where  $x'_i$  is a column vector representing the  $i$ th example from  $X'$ .

Train  $\widehat{W}$  on the MNIST training data with  $\lambda = 10^{-4}$  and make label predictions on the test data. What is the training and testing classification accuracy (they should both be about 85%)?

- d. We just fit a classifier that was linear in the pixel intensities to the MNIST data. For classification of digits the raw pixel values are very, very bad features: it's pretty hard to separate digits with linear functions in pixel space. The standard solution to this is to come up with some transform  $h: \mathbb{R}^d \rightarrow \mathbb{R}^p$  of the original pixel values such that the transformed points are (more easily) linearly separable. In this problem, you'll use the feature transform:

$$h(x) = \cos(Gx + b).$$

where  $G \in \mathbb{R}^{p \times d}$ ,  $b \in \mathbb{R}^p$ , and the cosine function is applied elementwise. We'll choose  $G$  to be a *random* matrix, with each entry sampled i.i.d. with mean  $\mu = 0$  and variance  $\sigma^2 = 0.1$ , and  $b$  to be a random vector sampled i.i.d. from the uniform distribution on  $[0, 2\pi]$ . The big question is: *how do we choose  $p$* ? Cross-validation, of course!

Randomly partition your training set into proportions 80/20 to use as a new training set and validation set, respectively. Using the `train` function you wrote above, train a  $\widehat{W}^p$  for different values of  $p$  and plot the classification training error and validation error on a single plot with  $p$  on the  $x$ -axis. Be careful, your computer may run out of memory and slow to a crawl if  $p$  is too large ( $p \leq 6000$  should fit into 4 GB of memory). You can use the same value of  $\lambda$  as above but feel free to study the effect of using different values of  $\lambda$  and  $\sigma^2$  for fun.

- e. Instead of reporting just the classification test error, which is an unbiased estimate of the *true* error, we would like to report a *confidence interval* around the test error that contains the true error. For any  $\delta \in (0, 1)$ , it follows from Hoeffding's inequality that if  $X_i$  for all  $i = 1, \dots, m$  are i.i.d. random variables with  $X_i \in [a, b]$  and  $\mathbb{E}[X_i] = \mu$ , then with probability at least  $1 - \delta$

$$\mathbb{P} \left( \left| \left( \frac{1}{m} \sum_{i=1}^m X_i \right) - \mu \right| \geq \sqrt{\frac{\log(2/\delta)}{2m}} \right) \leq \delta$$

We will use the above equation to construct a confidence interval around our true classification error since the test error is just the average of indicator variables taking values in 0 or 1 corresponding to the  $i$ th test example being classified correctly or not, respectively, where an error happens with probability  $\mu$ , the *true* classification error.

Let  $\widehat{p}$  be the value of  $p$  that approximately minimizes the validation error on the plot you just made and use  $\widehat{W}^{\widehat{p}}$  to compute the classification test accuracy, which we will denote as  $E_{test}$ . Use Hoeffding's inequality, above, to compute a confidence interval that contains  $\mathbb{E}[E_{test}]$  (i.e., the *true* error) with probability at least 0.95 (i.e.,  $\delta = 0.05$ ). Report  $E_{test}$  and the confidence interval.