INFR10086 Machine Learning (MLG) Semester 2, 2025/6

Exercises 3

Lecturer: Hao Tang

Exercise 1. Given a data set $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$ whose elements $\mathbf{x}_i \in \mathbb{R}^d$ and $y_i \in \mathbb{R}$, show that the mean-squared error

$$L = \sum_{i=1}^{n} (y_i - \mathbf{w}^\top \mathbf{x}_i)^2 \tag{1}$$

is convex in $\mathbf{w}$.

Since the square is convex and we already know that the affine transform and nonnegative sum of convex functions preserve convexity, L is convex.

Exercise 2. Given a data set $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$ whose elements $\mathbf{x}_i \in \mathbb{R}^d$ and $y_i \in \mathbb{R}$, show that the optimal solution for the mean-squared error

$$L = \sum_{i=1}^{n} (y_i - \mathbf{w}^\top \mathbf{x}_i)^2 \tag{2}$$

is

$$\mathbf{w} = (X^\top X)^{-1} X^\top \mathbf{y}, \tag{3}$$

where

$$X = \begin{bmatrix} \mathbf{x}_1^\top \\ \vdots \\ \mathbf{x}_n^\top \end{bmatrix} \qquad \mathbf{y} = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} \tag{4}$$

We start by expanding the loss function into

$$L = \sum_{i=1}^{n} \left[y_i^2 - 2 y_i \mathbf{w}^\top \mathbf{x}_i + (\mathbf{w}^\top \mathbf{x}_i)^2 \right]. \tag{5}$$

Given that the function is convex, the optimal solution would be the point where the gradient is

zero. The gradient of the loss is

$$\nabla_{\mathbf{w}} L = \sum_{i=1}^{n} \Big[-2y_i \mathbf{x}_i + 2(\mathbf{w}^\top \mathbf{x}_i)\mathbf{x}_i \Big] \tag{6}$$

$$= -2 \begin{bmatrix} \mathbf{x}_1 & \cdots & \mathbf{x}_n \end{bmatrix} \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} + 2 \begin{bmatrix} \mathbf{x}_1 & \cdots & \mathbf{x}_n \end{bmatrix} \begin{bmatrix} \mathbf{w}^\top \mathbf{x}_1 \\ \vdots \\ \mathbf{w}^\top \mathbf{x}_n \end{bmatrix} \tag{7}$$

$$= -2 \begin{bmatrix} \mathbf{x}_1 & \cdots & \mathbf{x}_n \end{bmatrix} \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} + 2 \begin{bmatrix} \mathbf{x}_1 & \cdots & \mathbf{x}_n \end{bmatrix} \begin{bmatrix} \mathbf{x}_1^\top \\ \vdots \\ \mathbf{x}_n^\top \end{bmatrix} \mathbf{w} \tag{8}$$

$$= -2X^\top \mathbf{y} + 2X^\top X \mathbf{w}. \tag{9}$$

Solving $\nabla_{\mathbf{w}} L = 0$ gives $\mathbf{w} = (X^\top X)^{-1} X^\top \mathbf{y}$.

Exercise 3. In this question, we are going to extend the previous result to predicting a vector instead of just a value. Given a data set $\{(\mathbf{x}_1, \mathbf{y}_1), \ldots, (\mathbf{x}_n, \mathbf{y}_n)\}$ whose elements $\mathbf{x}_i \in \mathbb{R}^p$ and $\mathbf{y}_i \in \mathbb{R}^q$, show that the optimal solution for the mean-squared error

$$L = \sum_{i=1}^{n} \|\mathbf{y}_i - W\mathbf{x}_i\|^2 \tag{10}$$

is

$$W = Y^\top X (X^\top X)^{-1}, \tag{11}$$

where

$$X = \begin{bmatrix} \mathbf{x}_1^\top \\ \vdots \\ \mathbf{x}_n^\top \end{bmatrix} \qquad \mathbf{y} = \begin{bmatrix} \mathbf{y}_1^\top \\ \vdots \\ \mathbf{y}_n^\top \end{bmatrix} \tag{12}$$

Following the same strategy, we expand the loss function into

$$L = \sum_{i=1}^{n} [\|\mathbf{y}_i\|^2 - 2\mathbf{y}_i^\top W \mathbf{x}_i + \|W\mathbf{x}_i\|^2]. \tag{13}$$

The gradient is

$$\nabla_W L = \sum_{i=1}^{n} [-2\mathbf{y}_i \mathbf{x}_i^\top + 2W\mathbf{x}_i \mathbf{x}_i^\top], \tag{14}$$

and solving $\nabla_W L = 0$ gives

$$W = \left(\sum_{i=1}^{n} \mathbf{y}_i \mathbf{x}_i^\top \right) \left(\sum_{i=1}^{n} \mathbf{x}_i \mathbf{x}_i^\top \right)^{-1} = Y^\top X (X^\top X)^{-1}. \tag{15}$$

Exercise 4. In this question, we are going to train a simple regression model on MNIST to generate an image from a label. MNIST is a labeled data set $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$, where $\mathbf{x}_i \in \mathbb{R}^{784}$ is a 28×28 image and $y_i \in \{0, \ldots, 9\}$ is a label. Our goal is to train a function $f(y) = W\mathbf{1}_y$, where $W \in \mathbb{R}^{784\times 10}$ and $\mathbf{1}_y \in \mathbb{R}^{10}$ is a one-hot vector where the y-th coordinate is 1 and is otherwise 0.

1. Write a python script that trains a model $f(y) = W\mathbf{1}_y + \mathbf{b}$ by minimizing the mean-squared error using stochastic gradient descent. You can modify the training script from Tutorial 2.

2. Given the trained f, visualize the rows of W. Each row of W has 784 dimensions. Reshape each row to 28×28 and plot them with `matplotlib.imshow`. Note that $W \in \mathbb{R}^{784\times 10}$, so you should see 10 images.

3. Write a script that computes the mean of each individual digits, and visualize them with `matplotlib.imshow`. Compare them with those from the rows of W. Do they look nearly identical?

4. Given a data set $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$, show that the optimal W to the mean-squared error

$$L = \sum_{i=1}^{n} \|\mathbf{x}_i - (W\mathbf{1}_{y_i} + \mathbf{b})\|^2 \tag{16}$$

   consists of the means of each digit and the optimal $\mathbf{b}$ is the mean of all digits. In other words, the k-th row of the optimal W has the form

$$\mathbf{w}_k = \frac{\sum_{i=1}^{n} \mathbb{1}_{y_i=k}\mathbf{x}_i}{\sum_{i=1}^{n} \mathbb{1}_{y_i=k}}. \tag{17}$$

   This problem is known as regression to the mean.

5. Since we standardize the images already before training, confirm that the $\mathbf{b}$ after training is nearly a 0 vector by showing that its norm is close to 0.

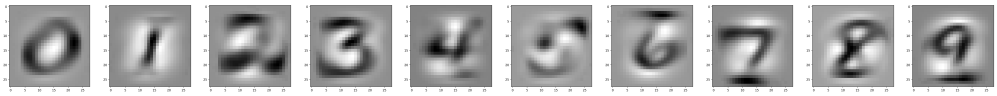
1. It is the exact same script but with a few changes. The loss function and its gradient are listed below. Note that the vector that gets multiplied with W is the one-hot vector of y, and this reflects in the code as indexing the y-th row of W, i.e., `W[y]`.

```
def l2_loss(W, b, x, y):
    r = W[y] + b - x
    return numpy.dot(r, r)

def grad_l2_loss(W, b, x, y):
    gW = numpy.zeros_like(W)
    gW[y] = 2 * (W[y] - x)
    gb = 2 * (b - x)
    return gW, gb
```

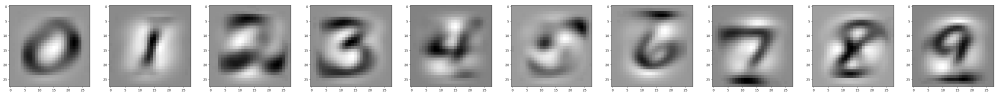
The loop for SGD needs to use the functions defined above.

```
for i in indices:
    xi = train_images[i].reshape(784)
    xi = (xi - mean) / numpy.sqrt(var_ + 1e-8)

    yi = train_labels[i]

    loss = l2_loss(weight, bias, xi, yi)
    gW, gb = grad_l2_loss(weight, bias, xi, yi)

    weight = weight - eta * gW
    bias = bias - eta * gb
```

2. The rows are visualized below.



3. The individual means are visualized below.



4. We follow the strategy again and expand the loss function to

$$L = \sum_{i=1}^{n} [\|\mathbf{x}_i\|^2 - 2x_i^\top (W\mathbf{1}_{y_i} + b) + \|W(\mathbf{1}_{y_i} + b)\|^2]. \tag{18}$$

Next, we find the gradient to be

$$\nabla_W L = \sum_{i=1}^{n} [-2\mathbf{x}_i \mathbf{1}_{y_i}^\top + 2W(\mathbf{1}_{y_i} + b)\mathbf{1}_{y_i}^\top] \tag{19}$$

$$= -2\sum_{i=1}^{n} \mathbf{x}_i \mathbf{1}_{y_i}^\top + 2W \sum_{i=1}^{n} \mathbf{1}_{y_i} \mathbf{1}_{y_i}^\top. \tag{20}$$

Solving $\nabla_W L = 0$ gives us

$$W = \left(\sum_{i=1}^{n} \mathbf{x}_i \mathbf{1}_{y_i}^\top \right) \left(\sum_{i=1}^{n} \mathbf{1}_{y_i} \mathbf{1}_{y_i}^\top \right)^{-1}. \tag{21}$$

Note that the matrix $\sum_{i=1}^{n} \mathbf{1}_{y_i} \mathbf{1}_{y_i}^\top$ is a diagonal matrix C, where $c_{kk} = \sum_{i=1}^{n} \mathbb{1}_{y_i=k}$. Similarly, the matrix $S = \sum_{i=1}^{n} \mathbf{x}_i \mathbf{1}_{y_i}^\top$ accumulates the images of the same digit, i.e., the k-th row of S is $\sum_{i=1}^{n} \mathbb{1}_{y_i=k} \mathbf{x}_i$. It is now clear that the k-th row of W is

$$\mathbf{w}_k = \frac{\sum_{i=1}^{n} \mathbb{1}_{y_i=k}}{\sum_{i=1}^{n} \mathbb{1}_{y_i=k} \mathbf{x}_i}. \tag{22}$$

5. The norm of the bias should theoretically be 0, and empirically, I got a norm of 0.176 after running SGD with step size 5×10^{-5} for 20 epochs.