ECE421: Introduction to Machine Learning — Fall 2024

Worksheet 3: Gradient Descent, Multi-class Logistic Regression, Regularized Regression Model

Q0 (Matrix Decomposition Primer) The singular value decomposition is frequently used as part of unsupervised learning algorithms. In particular, the SVD forms the basis of an algorithm called principal components analysis (PCA), which reduces data dimensions such that the reduced data still contains the maximum amount of variability given the number of reduced dimensions. The reduction axes picked by PCA directly correspond to the singular vectors identified by the singular value decomposition. In the following questions, we review matrix decomposition.
[**NOTE:** Your TAs won't review **Q0** during the tutorial session. Please review your linear algebra notes/references if you are struggling with the following questions.]

0.a (Eigen-based Decomposition) For a square diagonalizable matrix $A \in \mathbb{R}^{n\times n}$ the eigendecomposition is given by $A = Q\Lambda Q^{-1}$, where $Q \in \mathbb{R}^{n\times n}$ is a square matrix that contains the eigenvectors $\underline{q}_i \in \mathbb{R}^n$ as columns, and Λ is a diagonal matrix[1] whose entries correspond to the eigenvalues of the respective eigenvectors from Q. In particular $\Lambda_{ii} = \lambda_i$ is the eigenvalue associated with eigenvector $\underline{q}_i$. [2]

Compute the characteristic polynomial of $A = \begin{bmatrix} 4 & 0 & -2 \\ 1 & 3 & -2 \\ 1 & 2 & -1 \end{bmatrix}$ and determine its eigenvalues and eigenvectors. Then, find the eigendecomposition of A.

0.b (Singular Value Decomposition) The Singular Value Decomposition (SVD) generalizes the concepts from the eigendecomposition to general matrices $A \in \mathbb{R}^{m\times n}$ by decomposing it as $A = U\Sigma V^\top$, where, $U \in \mathbb{R}^{m\times m}$ is an orthogonal matrix[3], $\Sigma \in \mathbb{R}^{m\times n}$ is a diagonal matrix, and $V \in \mathbb{R}^{n\times n}$ is an orthogonal matrix.

Derive the singular value decomposition of $A = \begin{bmatrix} 3 & 2 & 2 \\ 2 & 3 & -2 \end{bmatrix}$ with following steps.[4]

- **0.b.i** Compute $A^\top A$, yielding a square symmetric matrix.
- **0.b.ii** Find the eigenvalues for $A^\top A$.
- **0.b.iii** Find the eigenvectors for $A^\top A$.
- **0.b.iv** Normalize the eigenvectors of $A^\top A$ to get V.
- **0.b.v** Find U using the normalized eigenvectors of V where $\underline{u}_i = \frac{1}{\sigma_i} A\underline{v}_i$.

Q1 (Regularized Linear Regression) Many machine learning algorithms use ℓ_p norms to either

- measure the distance between points in a high-dimensional data space (*e.g.*, k-nearest neighbor classification); or
- bound the magnitude of a vector to a specific value (*e.g.*, regularization in linear regression).

In this question, you will train a regularized linear regression model with an ℓ_2 regularization penalty.

You are given a dataset $\mathcal{D} = \{(\underline{x}_n, y_n)\}_{n=1}^N$, where $\underline{x}_n \in \mathbb{R}^{d+1}, d \geq 1$, and $y_n \in \{+1, -1\}$. We would like to train a regularized linear regression model, where the mean squared loss is augmented with an ℓ_2 regularization

[1] A matrix M is called diagonal if all the entries outside of its main diagonal are 0. The main diagonal is the top-left to bottom-right diagonal.

[2] A non-zero vector $\underline{v} \in \mathbb{R}^n$ is called an eigenvector of a square diagonalizable matrix $A \in \mathbb{R}^{n\times n}$ if, for a scalar $\lambda \in \mathbb{R}$ it satisfies $A\underline{v} = \lambda\underline{v}$. Intuitively speaking, an eigenvector $\underline{v}$ is a vector which, under the transformation applied by A, is only scaled in its magnitude. In particular, such vectors $\underline{v}$ stay on their own span and are only elongated or shrinked. The degree of change in magnitude inflicted by A can be summarized in a single scalar λ, which is the eigenvalue corresponding to $\underline{v}$. The set of all eigenvalue-eigenvector combinations can be computed by solving for the characteristic polynomial yielded by $det(A - \lambda I) = 0$.

[3] A square matrix M is called orthogonal if $MM^\top = M^\top M = I$.

[4] A nice visualization of SVD can be found here: https://youtu.be/vSczTbgc8Rc?si=iIYcflhGdcdYV2bv

penalty $\|\underline{w}\|_2^2$ on the weight parameter $\underline{w} \in \mathbb{R}^{d+1}$, *i.e.*,

$$E_{\text{in}}(\underline{w}) = \frac{1}{2N}\sum_{n=1}^{N}\left(\underline{w}^\top \underline{x}_n - y_n\right)^2 + \frac{\lambda}{2}\|\underline{w}\|_2^2 = \frac{1}{2N}\|X\underline{w} - \underline{y}\|_2^2 + \frac{\lambda}{2}\|\underline{w}\|_2^2.$$

where $\lambda > 0$ is a hyperparameter controlling the penalty, and the data matrix X and target vector $\underline{y}$ are defined as $X = \begin{bmatrix} \underline{x}_1^\top \\ \vdots \\ \underline{x}_N^\top \end{bmatrix} \in \mathbb{R}^{N\times(d+1)}$, and $\underline{y} = \begin{bmatrix} y_1 \\ \vdots \\ y_N \end{bmatrix} \in \mathbb{R}^N$.

1.a Prove that all eigenvalues of $X^\top X$ are non-negative.
[**HINT:** An $n \times n$ real matrix M is is said to be **positive-semidefinite** if $\underline{z}^\top M \underline{z} \geq 0$, for any non-zero $\underline{z} \in \mathbb{R}^n$. One can show that $M \in \mathbb{R}^{n\times n}$ is positive-semidefinite **if and only if** all of its eigenvalues are non-negative. So, to prove that all eigenvalues of $X^\top X$ are non-negative, it suffices to prove that $\underline{z}^\top X^\top X \underline{z} \geq 0$ for any non-zero vector $\underline{z}$.]

Answer. It suffices to prove that $\underline{z}^\top X^\top X \underline{z} \geq 0$ for any non-zero vector $\underline{z}$. Observe that

$$\underline{z}^\top X^\top X \underline{z} = \|X\underline{z}\|_2^2 \geq 0.$$

1.b Demonstrate that matrix $X^\top X + \lambda N I$, where I is the identity matrix, is invertible by proving that none of its eigenvalues are zero.
[**HINT:** Assume $\underline{v}$ is an eigenvector of $X^\top X$ with corresponding eigenvalue μ. In **1.a**, you proved that $\mu \geq 0$. Now, show that $\underline{v}$ is also an eigenvector of $X^\top X + \lambda N I$ and find its corresponding eigenvalue.]

Answer. Assume an arbitrary eigenvector of $X^\top X$, $\underline{v}$, and its corresponding eigenvalue μ. Therefore, $X^\top X\underline{v} = \mu\underline{v}$. By **1.a**, $\mu \geq 0$. Observe that $(X^\top X + \lambda N I)\underline{v} = X^\top X\underline{v} + \lambda N I\underline{v} = \mu\underline{v} + N\underline{v} = (\mu + N)\underline{v}$. Hence, $\underline{v}$ is the eigenvector of $(X^\top X + \lambda N I)$ and its corresponding eigenvalue is $(\mu + N) > 0$. Thus, all the eigenvalues of $(X^\top X + \lambda N I)$ are non-zero.

1.c Using the invertibility of matrix $X^\top X + \lambda N I$, find an analytical solution for $\underline{w}^\star = \arg\min_{\underline{w}} E_{\text{in}}(\underline{w})$.
Congratulations! You have trained a regularized linear regression model with an ℓ_2 regularization penalty.

Answer. As $E_{\text{in}}(\underline{w})$ is a convex function, $\nabla_{\underline{w}} E_{\text{in}}(\underline{w}^\star) = 0$. Note that

$$\begin{aligned}
\nabla_{\underline{w}} E_{\text{in}}(\underline{w}) &= \nabla_{\underline{w}}\left(\frac{1}{2N}(X\underline{w} - \underline{y})^\top (X\underline{w} - \underline{y}) + \frac{\lambda}{2}\underline{w}^\top \underline{w}\right) = \\
&= \frac{1}{2N}\nabla_{\underline{w}}\left((X\underline{w} - \underline{y})^\top (X\underline{w} - \underline{y})\right) + \frac{\lambda}{2}\nabla_{\underline{w}}\underline{w}^\top \underline{w} \\
&= \frac{1}{2N}\nabla_{\underline{w}}\left((\underline{w}^\top X^\top - \underline{y}^\top)(X\underline{w} - \underline{y})\right) + \lambda I\underline{w} \\
&= \frac{1}{2N}\nabla_{\underline{w}}\left(\underline{w}^\top X^\top X\underline{w} - 2\underline{w}^\top X^\top \underline{y} + \underline{y}^\top \underline{y}\right) + \lambda I\underline{w} \\
&= \frac{1}{2N}\left(2X^\top X\underline{w} - 2X^\top \underline{y}\right) + \lambda I\underline{w} \\
&= \frac{1}{N}\left((X^\top X + N\lambda I)\underline{w} - X^\top \underline{y}\right).
\end{aligned}$$

Therefore, $\nabla_{\underline{w}} E_{\text{in}}(\underline{w}^\star) = \frac{1}{N}\left((X^\top X + N\lambda I)\underline{w}^\star - X^\top \underline{y}\right) = 0 \Rightarrow \underline{w}^\star = (X^\top X + N\lambda I)^{-1} X^\top \underline{y}$, as by **1.b**, $(X^\top X + N\lambda I)$ is invertible.

Q2 (Hinge Loss and Stochastic Gradient Descent) Consider the same *Rotten Tomatoes* reviews in the previous worksheet:

> **Panos Kotzathanasis** (Asian Movie Plus): "Perfect Blue" is an artistic and technical masterpiece; however, what is of outmost importance is the fact that Satoshi Kon never deteriorate from the high

standards he set here, in the first project that was entirely his own.

Derek Smith (Cinematic Reflections): [An] nime thriller [that] often plays as an examination of identity and celebrity, but ultimately gets so lost in its own complex structure that it doesn't end up saying much at all.

Rotten Tomatoes has classified these reviews as "positive" and "negative," respectively.

In this assignment, you will create a simple text classification system that can perform this task automatically. similar to the previous worksheet, we'll warm up with the following set of four mini-reviews, each labeled positive $(+1)$ or negative (-1):

1. $\underline{x}_1$: not good; label: (-1)
2. $\underline{x}_2$: pretty bad; label: (-1)
3. $\underline{x}_3$: good plot; label: $(+1)$
4. $\underline{x}_4$: pretty scenery; label: $(+1)$

Each review $\underline{x}$ is mapped onto a feature vector $\phi(\underline{x})$, which maps each word to the number of occurrences of that word in the review and adds a 1 to account for bias. For example, the second review maps to the (sparse) feature vector $\phi(\underline{x}_2) = \{\textbf{extra added 1} : 1, \textbf{pretty} : 1, \textbf{bad} : 1\}$. The hinge loss for a single datapoint is defined as

$$L_{\text{hinge}}(\underline{x}, y, \underline{w}) = \max\{0, 1 - y\underline{w}^\top \phi(\underline{x})\},$$

where $\underline{x}$ is the review text, y is the correct label, $\underline{w}$ is the weight vector.

Suppose we run **stochastic gradient descent** once for each of the 4 samples in the order given above, updating the weights according to

$$\underline{w} \leftarrow \underline{w} - \epsilon \nabla_{\underline{w}} L_{\text{hinge}}(\underline{x}, y, \underline{w}).$$

After the updates, what are the bias and the weights of the six words ("pretty", "good", "bad", "plot", "not", "scenery") that appear in the above reviews?

- Use $\epsilon = 0.1$ as the step size.
- Initialize $\underline{w} = [0, 0, 0, 0, 0, 0, 0]$.
- The gradient $\nabla_{\underline{w}} L_{\text{hinge}}(\underline{x}, y, \underline{w}) = \underline{0}$ when margin is exactly 1 (*i.e.*, when $y\underline{w}^\top \phi(\underline{x}) = 1$).

Present your answer as a weight vector that contains a numerical value for each of the tokens in the reviews ("extra added 1", "pretty", "good", "bad", "plot", "not", "scenery").

Answer. First, let's formulate the gradient of the loss function.

$$\nabla_{\underline{w}} L_{\text{hinge}}(\underline{x}, y, \underline{w}) = \begin{cases} -\phi(\underline{x})y, & \text{if } \underline{w}^\top \phi(\underline{x}) y < 1, \\ \underline{0}, & \text{if } \underline{w}^\top \phi(\underline{x}) y \geq 1. \end{cases}$$

Step 1 (datapoint $\underline{x}_1$): Observe that $\underline{w}^\top \phi(\underline{x}_1) y_1 = 0$ and $y_1 = -1$. Thus, $\nabla_{\underline{w}} L_{\text{hinge}}(\underline{x}_1, y_1, \underline{w}) = \phi(\underline{x}_1)$. Therefore,

$$\underline{w} \leftarrow \underline{w} - 0.1\phi(\underline{x}_1).$$

Hence, $\underline{w} = [-0.1, 0, -0.1, 0, 0, -0.1, 0]$.

Step 2 (datapoint $\underline{x}_2$): Observe that $\underline{w}^\top \phi(\underline{x}_2) y_2 = 0.1$ and $y_1 = -1$. Thus, $\nabla_{\underline{w}} L_{\text{hinge}}(\underline{x}_2, y_2, \underline{w}) = \phi(\underline{x}_2)$. Therefore,

$$\underline{w} \leftarrow \underline{w} - 0.1\phi(\underline{x}_2).$$

Hence, $\underline{w} = [-0.2, -0.1, -0.1, -0.1, 0, -0.1, 0]$.

Step 3 (datapoint $\underline{x}_3$): Observe that $\underline{w}^\top \phi(\underline{x}_3) y_3 = -0.3$ and $y_3 = 1$. Thus, $\nabla_{\underline{w}} L_{\text{hinge}}(\underline{x}_3, y_3, \underline{w}) = -\phi(\underline{x}_3)$.

$$\underline{w} \leftarrow \underline{w} + 0.1\phi(\underline{x}_3).$$

Hence, $\underline{w} = [-0.1, -0.1, 0, -0.1, 0.1, -0.1, 0]$.

Step 4 (datapoint $\underline{x}_4$): Observe that $\underline{w}^\top \phi(\underline{x}_4) y_4 = -0.2$ and $y_4 = 1$. Thus, $\nabla_{\underline{w}} L_{\text{hinge}}(\underline{x}_4, y_4, \underline{w}) = -\phi(\underline{x}_4)$.

$$\underline{w} \leftarrow \underline{w} + 0.1\phi(\underline{x}_4).$$

Hence, $\underline{w} = [0, 0, 0, -0.1, 0.1, -0.1, 0.1]$.

Q3 (Problem 2, Midterm 2019, Multi-class softmax regression model) Suppose we use a multi-class softmax regression model to classify input data vectors $\underline{x} \in \mathbb{R}^{d+1}$ with two possible class labels $y \in \{1, 2\}$. Let $\underline{w}(1)$ and $\underline{w}(2)$ be the weight vectors for class 1 and 2, respectively. For any input $\underline{x}$, we hypothesize that the probability of $\underline{x}$ belonging to class $i \in \{1, 2\}$ is

$$\mathbb{P}^{\mathsf{SM}}(y = i \mid \underline{x}) = \frac{\exp(\underline{w}(i)^\top \underline{x})}{\exp(\underline{w}(1)^\top \underline{x}) + \exp(\underline{w}(2)^\top \underline{x})}.$$

For a training example $(\underline{x}_n, y_n)$, let the loss function be defined as

$$e_n^{\mathsf{SM}}(\underline{w}(1), \underline{w}(2)) = -\log(\mathbb{P}^{\mathsf{SM}}(y_n \mid \underline{x}_n)) = -\log\left[\frac{\exp(\underline{w}(y_n)^\top \underline{x}_n)}{\exp(\underline{w}(1)^\top \underline{x}_n) + \exp(\underline{w}(2)^\top \underline{x}_n)}\right].$$

3.a Find the gradients of e_n^{SM} with respect to $\underline{w}(1)$ and $\underline{w}(2)$. (Note that you should always consider two possible values of y_n).

Answer. If $y_n = 1$,

$$\begin{aligned}
\nabla_{\underline{w}(1)} e_n^{\mathsf{SM}}(\underline{w}(1), \underline{w}(2)) &= -\nabla_{\underline{w}(1)} \log(\mathbb{P}^{\mathsf{SM}}(y_n \mid \underline{x}_n)) = -\nabla_{\underline{w}(1)} \log\left[\frac{\exp(\underline{w}(1)^\top \underline{x}_n)}{\exp(\underline{w}(1)^\top \underline{x}_n) + \exp(\underline{w}(2)^\top \underline{x}_n)}\right] \\
&= -\nabla_{\underline{w}(1)} \left(\underline{w}(1)^\top \underline{x}_n - \log\left(\exp(\underline{w}(1)^\top \underline{x}_n) + \exp(\underline{w}(2)^\top \underline{x}_n)\right)\right) \\
&= -\underline{x}_n + \frac{\nabla_{\underline{w}(1)} \left(\exp(\underline{w}(1)^\top \underline{x}_n) + \exp(\underline{w}(2)^\top \underline{x}_n)\right)}{\exp(\underline{w}(1)^\top \underline{x}_n) + \exp(\underline{w}(2)^\top \underline{x}_n)} \\
&= -\underline{x}_n + \frac{\exp(\underline{w}(1)^\top \underline{x}_n)}{\exp(\underline{w}(1)^\top \underline{x}_n) + \exp(\underline{w}(2)^\top \underline{x}_n)} \underline{x}_n \\
&= \frac{-\exp(\underline{w}(2)^\top \underline{x}_n)}{\exp(\underline{w}(1)^\top \underline{x}_n) + \exp(\underline{w}(2)^\top \underline{x}_n)} \underline{x}_n.
\end{aligned}$$

If $y_n = 2$,

$$\begin{aligned}
\nabla_{\underline{w}(1)} e_n^{\mathsf{SM}}(\underline{w}(1), \underline{w}(2)) &= -\nabla_{\underline{w}(1)} \log(\mathbb{P}^{\mathsf{SM}}(y_n \mid \underline{x}_n)) = -\nabla_{\underline{w}(1)} \log\left[\frac{\exp(\underline{w}(2)^\top \underline{x}_n)}{\exp(\underline{w}(1)^\top \underline{x}_n) + \exp(\underline{w}(2)^\top \underline{x}_n)}\right] \\
&= -\nabla_{\underline{w}(1)} \left(\underline{w}(2)^\top \underline{x}_n - \log\left(\exp(\underline{w}(1)^\top \underline{x}_n) + \exp(\underline{w}(2)^\top \underline{x}_n)\right)\right) \\
&= \frac{\nabla_{\underline{w}(1)} \left(\exp(\underline{w}(1)^\top \underline{x}_n) + \exp(\underline{w}(2)^\top \underline{x}_n)\right)}{\exp(\underline{w}(1)^\top \underline{x}_n) + \exp(\underline{w}(2)^\top \underline{x}_n)} \\
&= \frac{\exp(\underline{w}(1)^\top \underline{x}_n)}{\exp(\underline{w}(1)^\top \underline{x}_n) + \exp(\underline{w}(2)^\top \underline{x}_n)} \underline{x}_n.
\end{aligned}$$

Similarly, we can derive $\nabla_{\underline{w}(2)} e_n^{\mathsf{SM}}(\underline{w}(1), \underline{w}(2))$ as

$$\nabla_{\underline{w}(2)} e_n^{\mathsf{SM}}(\underline{w}(1), \underline{w}(2)) = \begin{cases} \frac{\exp(\underline{w}(2)^\top \underline{x}_n)}{\exp(\underline{w}(1)^\top \underline{x}_n) + \exp(\underline{w}(2)^\top \underline{x}_n)} \underline{x}_n, & \text{if } y_n = 1, \\ \frac{-\exp(\underline{w}(1)^\top \underline{x}_n)}{\exp(\underline{w}(1)^\top \underline{x}_n) + \exp(\underline{w}(2)^\top \underline{x}_n)} \underline{x}_n, & \text{if } y_n = 2. \end{cases}$$

3.b Suppose instead of the multi-class softmax regression model, we use a binary logistic regression model. For an input $\underline{x} \in \mathbb{R}^{d+1}$, we hypothesize that the probability of x belonging to class 1 is

$$\mathbb{P}^{\mathsf{LR}}(y = 1 \mid \underline{x}) = \frac{\exp(\underline{w}^\top \underline{x})}{1 + \exp(\underline{w}^\top \underline{x})}.$$

Therefore, $\underline{x}$ belongs to class 2 with probability $\mathbb{P}^{\mathsf{LR}}(y = 2 \mid \underline{x}) = 1 - \mathbb{P}^{\mathsf{LR}}(y = 1 \mid \underline{x})$. For an example $(\underline{x}_n, y_n)$ we define the loss function as

$$e_n^{\mathsf{LR}}(\underline{w}) = -\log(\mathbb{P}^{\mathsf{LR}}(y_n \mid \underline{x}_n)).$$

Find a relationship between $\underline{w}(1), \underline{w}(2)$ and $\underline{w}$, so that we have

$$\mathbb{P}^{\mathsf{SM}}(y = 1 \mid \underline{x}) = \mathbb{P}^{\mathsf{LR}}(y = 1 \mid \underline{x}), \quad \mathbb{P}^{\mathsf{SM}}(y = 2 \mid \underline{x}) = 1 - \mathbb{P}^{\mathsf{LR}}(y = 1 \mid \underline{x}).$$

Answer. Let $\underline{w} = \underline{w}(1) - \underline{w}(2)$. Observe that with such $\underline{w}$,

$$\begin{aligned}\mathbb{P}^{\mathsf{LR}}(y = 1 \mid \underline{x}) &= \frac{\exp((\underline{w}(1) - \underline{w}(2))^\top \underline{x})}{1 + \exp((\underline{w}(1) - \underline{w}(2))^\top \underline{x})} \\ &= \frac{\exp(\underline{w}(1)^\top \underline{x})}{\exp(\underline{w}(2)^\top \underline{x}) + \exp(\underline{w}(1)^\top \underline{x})} = \mathbb{P}^{\mathsf{SM}}(y = 1 \mid \underline{x}),\end{aligned}$$

and

$$\begin{aligned}\mathbb{P}^{\mathsf{LR}}(y = 2 \mid \underline{x}) &= 1 - \frac{\exp((\underline{w}(1) - \underline{w}(2))^\top \underline{x})}{1 + \exp((\underline{w}(1) - \underline{w}(2))^\top \underline{x})} = \frac{1}{1 + \exp((\underline{w}(1) - \underline{w}(2))^\top \underline{x})} \\ &= \frac{\exp(\underline{w}(2)^\top \underline{x})}{\exp(\underline{w}(2)^\top \underline{x}) + \exp(\underline{w}(1)^\top \underline{x})} = \mathbb{P}^{\mathsf{SM}}(y = 2 \mid \underline{x}).\end{aligned}$$

3.c Given $\underline{w}(1)$, $\underline{w}(2)$ and $\underline{w}$ as described in (b), we apply SGD to separately train the softmax regression model and binary logistic regression model, with constant learning rates ϵ^{SM} and ϵ^{LR}, respectively. For both models, all weights are initialized to zero, and we use the same random seed so that in each iteration of SGD the same random training example is selected. Find a relationship between ϵ^{SM} and ϵ^{LR}, so that $e_n^{\mathsf{SM}}(\underline{w}(1), \underline{w}(2))$ and $e_n^{\mathsf{LR}}(\underline{w})$ are identical in all iterations of SGD.

Answer. Let $\underline{w}_k(1)$ and $\underline{w}_k(2)$ denote the multi-class softmax regression model weight vectors after the k_{th} iteration of SGD with, and $\underline{w}_k$ denote the binary logistic regression model weight vector after the k_{th} iteration of SGD. Note that $\underline{w}_0 = \underline{w}_k(1) - \underline{w}_k(2) = \underline{0}$. Thus, by the result from **3.b**, $e_n^{\mathsf{LR}}(\underline{w}_0) = e_n^{\mathsf{SM}}(\underline{w}_0(1), \underline{w}_0(2))$ for any n. To have $e_n^{\mathsf{LR}}(\underline{w}_k) = e_n^{\mathsf{SM}}(\underline{w}_k(1), \underline{w}_k(2))$ for any iteration $k \geq 0$, it is enough to set the learning rates ϵ^{SM} and ϵ^{LR} such that:

> "for any datapoint n and any weight vectors $\underline{w}_t$, $\underline{w}(1)_t$ and $\underline{w}(2)_t$, if $\underline{w}_t = \underline{w}(1)_t - \underline{w}(2)_t$ then $\underline{w}_{t+1} = \underline{w}(1)_{t+1} - \underline{w}(2)_{t+1}$."

Assume $\underline{w}_t = \underline{w}(1)_t - \underline{w}(2)_t$. By SGD update rule,

$$\begin{aligned}\underline{w}_{t+1} &= \underline{w}_t - \epsilon^{\mathsf{LR}} \nabla_{\underline{w}} e_n^{\mathsf{LR}}(\underline{w}_t), \\ \underline{w}(1)_{t+1} - \underline{w}(2)_{t+1} &= \underline{w}(1)_t - \epsilon^{\mathsf{SM}} \nabla_{\underline{w}(1)} e_n^{\mathsf{SM}}(\underline{w}(1)_t, \underline{w}(2)_t) - \underline{w}(2)_t + \epsilon^{\mathsf{SM}} \nabla_{\underline{w}(2)} e_n^{\mathsf{SM}}(\underline{w}(1)_t, \underline{w}(2)_t) \\ &= \underline{w}(1)_t - \underline{w}(2)_t - \epsilon^{\mathsf{SM}} \left(\nabla_{\underline{w}(1)} e_n^{\mathsf{SM}}(\underline{w}(1)_t, \underline{w}(2)_t) - \nabla_{\underline{w}(2)} e_n^{\mathsf{SM}}(\underline{w}(1)_t, \underline{w}(2)_t) \right) \\ &= \underline{w}_t - \epsilon^{\mathsf{SM}} \left(\nabla_{\underline{w}(1)} e_n^{\mathsf{SM}}(\underline{w}(1)_t, \underline{w}(2)_t) - \nabla_{\underline{w}(2)} e_n^{\mathsf{SM}}(\underline{w}(1)_t, \underline{w}(2)_t) \right).\end{aligned}$$

Therefore, to obtain $\underline{w}_{t+1} = \underline{w}(1)_{t+1} - \underline{w}(2)_{t+1}$, it suffices to have

$$\epsilon^{\mathsf{LR}} \nabla_{\underline{w}} e_n^{\mathsf{LR}}(\underline{w}_t) = \epsilon^{\mathsf{SM}} \left(\nabla_{\underline{w}(1)} e_n^{\mathsf{SM}}(\underline{w}(1)_t, \underline{w}(2)_t) - \nabla_{\underline{w}(2)} e_n^{\mathsf{SM}}(\underline{w}(1)_t, \underline{w}(2)_t) \right), \forall y_n \in \{1, 2\}. \quad (1)$$

Observe that if $y_n = 1$.

$$\nabla_{\underline{w}} e_n^{\mathsf{LR}}(\underline{w}_t) = \frac{-1}{1+\exp\left(\underline{w}_t^\top \underline{x}_n\right)}\underline{x}_n = \frac{-1}{1+\exp\left(\left(\underline{w}(1)_t - \underline{w}(2)_t\right)^\top \underline{x}_n\right)}\underline{x}_n = \frac{-\exp(\underline{w}(2)^\top \underline{x}_n)}{\exp(\underline{w}(1)^\top \underline{x}_n)+\exp(\underline{w}(2)^\top \underline{x}_n)}\underline{x}_n,$$

$$\nabla_{\underline{w}(1)} e_n^{\mathsf{SM}}(\underline{w}(1)_t, \underline{w}(2)_t) - \nabla_{\underline{w}(2)} e_n^{\mathsf{SM}}(\underline{w}(1)_t, \underline{w}(2)_t) = \frac{-2\exp(\underline{w}(2)^\top \underline{x}_n)}{\exp(\underline{w}(1)^\top \underline{x}_n)+\exp(\underline{w}(2)^\top \underline{x}_n)}\underline{x}_n.$$

If $y_n = 2$.

$$\nabla_{\underline{w}} e_n^{\mathsf{LR}}(\underline{w}_t) = \frac{1}{1+\exp\left(-\underline{w}_t^\top \underline{x}_n\right)}\underline{x}_n = \frac{1}{1+\exp\left(\left(\underline{w}(2)_t - \underline{w}(1)_t\right)^\top \underline{x}_n\right)}\underline{x}_n = \frac{\exp(\underline{w}(1)^\top \underline{x}_n)}{\exp(\underline{w}(1)^\top \underline{x}_n)+\exp(\underline{w}(2)^\top \underline{x}_n)}\underline{x}_n,$$

$$\nabla_{\underline{w}(1)} e_n^{\mathsf{SM}}(\underline{w}(1)_t, \underline{w}(2)_t) - \nabla_{\underline{w}(2)} e_n^{\mathsf{SM}}(\underline{w}(1)_t, \underline{w}(2)_t) = \frac{2\exp(\underline{w}(1)^\top \underline{x}_n)}{\exp(\underline{w}(1)^\top \underline{x}_n)+\exp(\underline{w}(2)^\top \underline{x}_n)}\underline{x}_n.$$

Thus, to satisfy (1), it suffices to have $\epsilon^{\mathsf{SM}} = \frac{\epsilon^{\mathsf{LR}}}{2}$.