Name: Seowoo Han
Andrew ID: seowooh

Machine Learning for Text Mining

Homework 4 – Template

1. Statement of Assurance

1. Did you receive any help whatsoever from anyone in solving this assignment? [✓ Yes] / No.

If you answered ‘yes’, give full details? (e.g.“Jane explained to me what is asked in Question 3.4").

Byeongju Han explained to me what is the loss function of SGD and gave some advice about how can we run PMF.

Insoo Kime explained to me what is asked in the whole question and gave an institution.

Jungkyung Lee gave some advice about PMF implementation.

2. Did you give any help whatsoever to anyone in solving this assignment? Yes / [✓ No].

If you answered ‘yes’, give full details? (e.g. “I pointed Joe to section 2.3 to help him with Question 2").

3. Did you find or come across code that implements any part of this assignment? [✓ Yes] / No.

If you answered ‘yes’, give full details? (e.g. book & page, URL & location within the page, etc)

https://paws.kettering.edu/~ktebeest/math305/newton.html

https://www.pyimagesearch.com/2016/10/17/stochastic-gradient-descent-sgd-with-python/

https://guide.freecodecamp.org/machine-learning/support-vector-machine/

2. Writeup (40 pts)

(1) [10 pts] Gradient

$$f(\boldsymbol{w}) = \frac{1}{2}\boldsymbol{w}^T\boldsymbol{w} + \frac{\lambda}{n}\sum_{i=1}^{n} max(1 - y_i\boldsymbol{w}^T x_i, 0)^2 \quad \cdots (1)$$

$$= \frac{1}{2}\|\boldsymbol{w}\|^2 + \frac{\lambda}{n}\sum_{i=1}^{n} max(1 - y_i\boldsymbol{w}^T x_i, 0)^2 \quad \cdots (2)$$

For any i, if $y_i\boldsymbol{w}^T x_i = 1$, it is not differential.

Let's divide the case for $max(1 - y_i\boldsymbol{w}^T x_i, 0)$.

$$max(1 - y_i\boldsymbol{w}^T x_i, 0) = \begin{cases} 1 - y_i\boldsymbol{w}^T x_i & ,1 - y_i\boldsymbol{w}^T x_i > 0 \\ 0 & ,1 - y_i\boldsymbol{w}^T x_i = 0 \\ 0 & ,1 - y_i\boldsymbol{w}^T x_i < 0 \end{cases}$$

if $y_i\boldsymbol{w}^T x_i < 1$, $f(\boldsymbol{w})$ can be differentiated.

$$\nabla f(\boldsymbol{w}) = \frac{d}{d\boldsymbol{w}}\left(\frac{1}{2}\|\boldsymbol{w}\|^2 + \frac{\lambda}{n}\sum_{i=1}^{n} \max(1 - y_i\boldsymbol{w}^T x_i, 0)^2\right) \quad \cdots (3)$$

$$= \boldsymbol{w} + \frac{d}{d\boldsymbol{w}}\left(\frac{\lambda}{n}\sum_{i=1}^{n}(1 - y_i\boldsymbol{w}^T x_i)^{\,2}\right) \quad \cdots (4)$$

Let's solve the problem in matrix form.

$$\frac{d(1 - y_i\boldsymbol{w}^T x_i)^2}{d\boldsymbol{w}} = \begin{bmatrix} \dfrac{d(1 - y_i\boldsymbol{w}^T x_i)^2}{dw_1} \\ \dfrac{d(1 - y_i\boldsymbol{w}^T x_i)^2}{dw_{\boldsymbol{2}}} \\ \vdots \\ \dfrac{d(1 - y_i\boldsymbol{w}^T x_i)^2}{dw_{\boldsymbol{n}}} \end{bmatrix} \quad \cdots (5)$$

$$= \begin{bmatrix} 2(1 - y_i\boldsymbol{w}^T x_i^1)(-y_i x_i^1) \\ 2(1 - y_i\boldsymbol{w}^T x_i^2)(-y_i x_i^2) \\ \vdots \\ 2(1 - y_i\boldsymbol{w}^T x_i^n)(-y_i x_i^n) \end{bmatrix} \quad \cdots (6)$$

We can rewrite equation (6) to equation(7) because $(y_i)^2 = 1$.

$$= \begin{bmatrix} 2(y_i - \boldsymbol{w}^T x_i^1)x_i^1 \\ 2(y_i - \boldsymbol{w}^T x_i^2)x_i^2 \\ \vdots \\ 2(y_i - \boldsymbol{w}^T x_i^n)x_i^n \end{bmatrix} \quad \cdots (7)$$

Now, we change equation (4) like below:

$$\boldsymbol{w} + \frac{d}{d\boldsymbol{w}}\left(\frac{\lambda}{n}\sum_{i=1}^{n}(1 - y_i\boldsymbol{w}^T x_i)^2\right) = \boldsymbol{w} + \frac{2\lambda}{n} X_{I,:}^T\left(X_{I,:}\boldsymbol{w} - \boldsymbol{y_I}\right) \quad \cdots (8)$$

Finally, we get the gradient of $f(\boldsymbol{w})$.

$$\nabla f(\boldsymbol{w}) = \boldsymbol{w} + \frac{2\lambda}{n} X_{I,:}^T(X_{I,:}\boldsymbol{w} - \boldsymbol{y_I}) \quad \cdots (9)$$

(2) **[10 pts]** Hessian

We get equation (9) from the question 2-(1).

$$\nabla f(\boldsymbol{w}) = \boldsymbol{w} + \frac{2\lambda}{n} X_{I,:}^T(X_{I,:}\boldsymbol{w} - \boldsymbol{y_I}) \quad \cdots (9)$$

We differentiate Equation 9 once more. We have to express Equation 10 simply.

$$\frac{d^2 f(\boldsymbol{w})}{d\boldsymbol{w}^{\boldsymbol{2}}} = \frac{d}{d\boldsymbol{w}}(\boldsymbol{w} + \frac{2\lambda}{n} X_{I,:}^T\left(X_{I,:}\boldsymbol{w} - \boldsymbol{y_I}\right)) \quad \cdots (10)$$

$$= \frac{d\boldsymbol{w}}{d\boldsymbol{w}} + \frac{2\lambda}{n}\frac{d}{d\boldsymbol{w}}(X_{I,:}^T\left(X_{I,:}\boldsymbol{w} - \boldsymbol{y_I}\right)) \quad \cdots (11)$$

Let's solve the problem in matrix form.

$$\frac{d\boldsymbol{w}}{d\boldsymbol{w}} = \begin{bmatrix} \dfrac{dw_1}{dw_1} & \dfrac{dw_1}{dw_2} & \cdots & \dfrac{dw_1}{dw_d} \\ \dfrac{dw_2}{dw_1} & \dfrac{dw_2}{dw_2} & \cdots & \dfrac{dw_2}{dw_d} \\ \vdots & \vdots & \ddots & \vdots \\ \dfrac{dw_d}{dw_1} & \cdots & \cdots & \dfrac{dw_d}{dw_d} \end{bmatrix} = I_d \quad \cdots (12)$$

$$\frac{d^2\xi}{d\boldsymbol{w}^2} = \frac{2\lambda}{n}\begin{bmatrix} \sum_{i\in I}\frac{(\boldsymbol{w}^T x_i - y_i)x_i^1}{dw_1} & \sum_{i\in I}\frac{(\boldsymbol{w}^T x_i - y_i)x_i^1}{dw_2} & \cdots & \sum_{i\in I}\frac{(\boldsymbol{w} x_i - y_i)x_i^1}{dw_d} \\ \sum_{i\in I}\frac{(\boldsymbol{w}^T x_i - y_i)x_i^2}{dw_1} & \sum_{i\in I}\frac{(\boldsymbol{w}^T x_i - y_i)x_i^2}{dw_2} & \cdots & \sum_{i\in I}\frac{(\boldsymbol{w}^T x_i - y_i)x_i^2}{dw_d} \\ \vdots & \vdots & \ddots & \vdots \\ \sum_{i\in I}\frac{(\boldsymbol{w}^T x_i - y_i)x_i^d}{dw_1} & \cdots & \cdots & \sum_{i\in I}\frac{(\boldsymbol{w}^T x_i - y_i)x_i^d}{dw_d} \end{bmatrix} \cdots (13)$$

$$= \frac{2\lambda}{n}\begin{bmatrix} \sum_{i\in I} x_i^1 x_i^1 & \sum_{i\in I} x_i^1 x_i^2 & \cdots & \sum_{i\in I} x_i^1 x_i^d \\ \sum_{i\in I} x_i^2 x_i^1 & \sum_{i\in I} x_i^2 x_i^2 & \cdots & \sum_{i\in I} x_i^2 x_i^d \\ \vdots & \vdots & \ddots & \vdots \\ \sum_{i\in I} x_i^d x_i^1 & \cdots & \cdots & \sum_{i\in I} x_i^d x_i^d \end{bmatrix} = \boldsymbol{X}^T\boldsymbol{D}\boldsymbol{X} \cdots (14)$$

Finally, we get the hessian of $f(\boldsymbol{w})$.

$$\nabla^2 f(\boldsymbol{w}) = \frac{d\boldsymbol{w}}{d\boldsymbol{w}} + \frac{d^2\xi}{d\boldsymbol{w}^2} = I_d + \frac{2\lambda}{n}\boldsymbol{X}^T\boldsymbol{D}\boldsymbol{X}$$

(3) **[10 pts]** Optimality

It has the following constraint.

$$y_i \boldsymbol{w}^T \boldsymbol{x}_i \geq \boldsymbol{1} - \ \xi_i,\ \ \xi_i \geq 0$$

We can think of this constraint as equation (15).

$$\xi_i \geq \max(1 - y_i \boldsymbol{w}^T x_i, 0)\ \cdots (15)$$

If there is a global optimum (minimum) of $\max(1 - y_i \boldsymbol{w}^T x_i, 0)$, then ξ_i must also be present. However, since equation (15) is the constraint, it is minimal when $\xi_i = \max(1 - y_i \boldsymbol{w}^T x_i, 0)$.

(4) **[10 pts]** Algorithm Pseudo Code

1) Mini-batch stochastic gradient method

```
m: mini-batch size
Initialize $\boldsymbol{w} := \boldsymbol{0}^{m-1}$, $b = 0$
for iteration $t \in [1, \cdots, \mathrm{T}]$:
    for $i \in [1, \cdots, m]$:
        set x, y data for mini-batch: $< \mathbf{x}^{[i]}, \mathrm{y}^{[i]} > \in \boldsymbol{D}$
    compute loss $\mathcal{L} := \frac{1}{m}\sum_{i=1}^{m} L(\hat{y}^{[i]}, \mathrm{y}^{[i]})$
    compute gradient $\Delta\boldsymbol{w} := -\boldsymbol{\nabla}_{\mathcal{L}}\boldsymbol{w}, \Delta b := -\frac{\partial\mathcal{L}}{\partial \boldsymbol{b}}$
    update parameters $\boldsymbol{w} := \boldsymbol{w} + \Delta\boldsymbol{w}$ , $b := b + \Delta b$
```

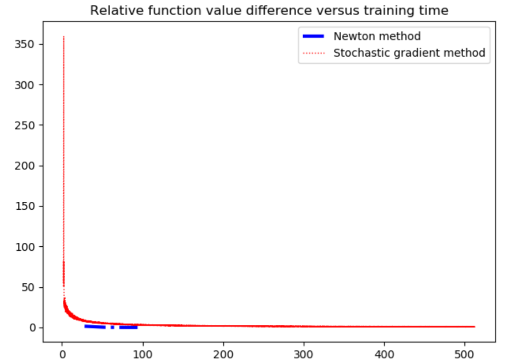
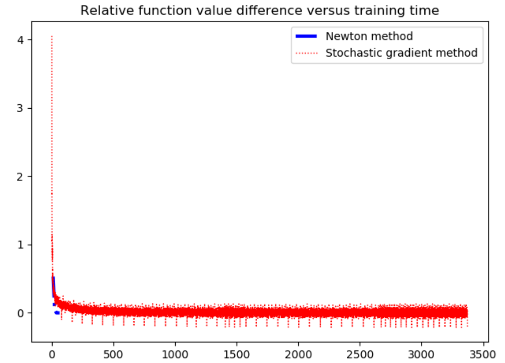
2) Newton method

```
thr: threshold to stop iterating
$\frac{df(x)}{dx} = f'(x)$.
for iteration $t \in [1, \cdots, \mathrm{T}]$:
    set $y_0 = f(x_0)$
set $y_p = f'(x_0)$
set $x_1 = x_0 - x_0/y_p$
if $|f(x_1)| <$ thr:
    halt
else:
    set $x_0 = x_1$
```

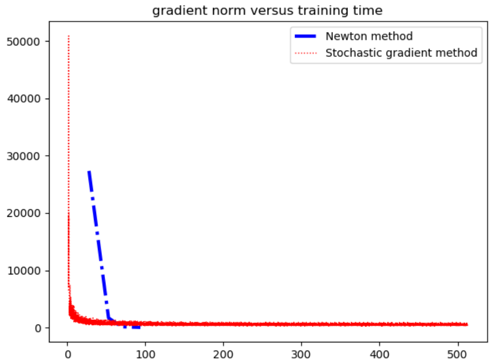
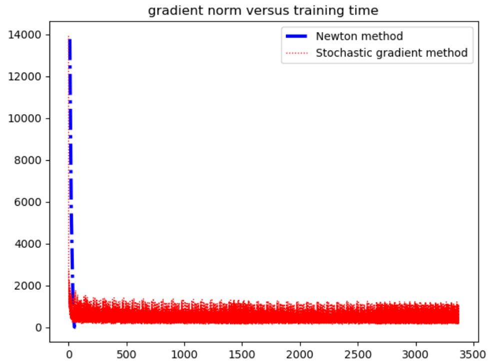
3. Experiments (20 pts)

Plot the figures for **both** of two datasets and **both** the approaches

(1) [5 pts] Relative function value difference versus training time

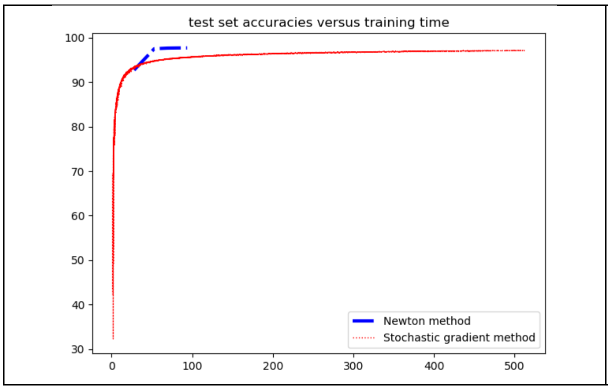
Realism	covtype
	

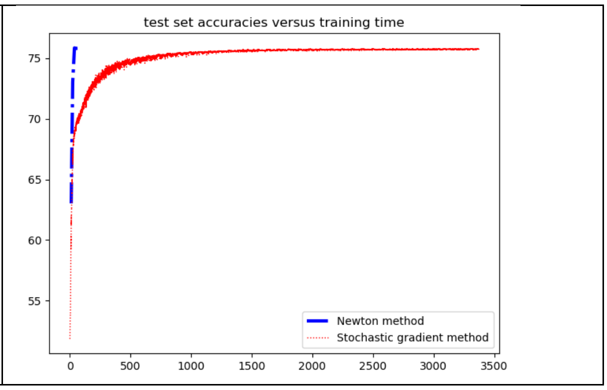
(2) [5 pts] Gradient norm versus training time

Realism	covtype
	

(3) [5 pts] Test set accuracies versus training time

Realism	covtype





(4) [5 pts] Discuss the difference between mini-batch SGD and Newton method in terms of the three types of figures

The reason why SGD's graph is drawn before newton is that newton calculates the relative function value, norm value, and accuracy after learning, but SGD implements calculation every minute in the mini-batch a ‘for’ statement. SGD tends to be jagged. The reason is that when we are learning that our algorithm is a mini-batch SGD, we turn the mini-batch to find the gradient. Unlike the Newton method, SGD can be described as going back, rather than immediately looking for an optimal value. Also, if the learning rate is too big, you may not find the optimal value. If it is too small, learning takes a long time. The newton method can find the optimal value without going through many iterations. Unlike SGD, when learning, the newton method takes less time to learn because it uses the entire data set. Both the SGD and the newton method can vary the type of optimum depending on how the initial value is set. Therefore, setting the initial value is essential for both methods.