Name: Changhyun, Lee
Andrew ID: changhyl

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? No.

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

2. Did you give any help whatsoever to anyone in solving this assignment? 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.

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

I referred the structure of the code below, such as kinds of function, skeleton. Then implemented my own algorithm and functions, and so forth.

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

2. Writeup (40 pts)

(1) [10 pts] Gradient

$$f(\mathbf{w}) = \frac{1}{2}\mathbf{w}^T\mathbf{w} + \frac{\lambda}{n}\sum_{i=1}^{n}\max(1 - y_i\mathbf{w}^T\mathbf{x}_i, 0)^2 \qquad \textbf{(1.1)}$$

$$\max(1 - y_i\mathbf{w}^T\mathbf{x}_i, 0)^2 = \begin{Bmatrix} (1 - y_i\mathbf{w}^T\mathbf{x}_i)^2, & if\ 1 - y_i\mathbf{w}^T\mathbf{x}_i < 0 \\ 0 & , if\ 1 - y_i\mathbf{w}^T\mathbf{x}_i \le 0 \end{Bmatrix} \qquad \textbf{(1.2)}$$

$$\nabla f(\mathbf{w}) = \frac{d}{d\mathbf{w}}\left(\frac{1}{2}\mathbf{w}^T\mathbf{w} + \frac{\lambda}{n}\sum_{i=1}^{n}\max(1 - y_i\mathbf{w}^T\mathbf{x}_i, 0)^2\right) \qquad \textbf{(1.3)}$$

$$= \mathbf{w} + \frac{d}{d\mathbf{w}}\left(\frac{\lambda}{n}\sum_{i=1}^{n}\max(1 - y_i\mathbf{w}^T\mathbf{x}_i, 0)^2\right) \qquad \textbf{(1.4)}$$

$$= \mathbf{w} + \frac{\lambda}{n}\sum_{i=1}^{n}\begin{bmatrix} \frac{d}{dw_1}\max(1 - y_i\mathbf{w}^T\mathbf{x}_i, 0)^2 \\ \frac{d}{dw_2}\max(1 - y_i\mathbf{w}^T\mathbf{x}_i, 0)^2 \\ \vdots \\ \frac{d}{dw_d}\max(1 - y_i\mathbf{w}^T\mathbf{x}_i, 0)^2 \end{bmatrix} \qquad \textbf{(1.5)}$$

$$\frac{d}{dw_k}\max(1-y_i\mathbf{w}^T\mathbf{x}_i,0)^2=\left\{\begin{matrix}2(1-y_i\mathbf{w}^T\mathbf{x}_i)(-y_i\mathbf{x}_{ik}), & if\ 1-y_i\mathbf{w}^T\mathbf{x}_i<0\\ 0 & ,\ if\ 1-y_i\mathbf{w}^T\mathbf{x}_i>0\\ Not\ differentiable & ,\ if\ 1-y_i\mathbf{w}^T\mathbf{x}_i=0\end{matrix}\right\} \qquad \textbf{(1.6)}$$

$$=\left\{\begin{matrix}2(y_i{}^2\mathbf{w}^T\mathbf{x}_i\mathbf{x}_{ik}-y_i\mathbf{x}_{ik}) & ,\ if\ 1-y_i\mathbf{w}^T\mathbf{x}_i<0\\ 0 & ,\ if\ 1-y_i\mathbf{w}^T\mathbf{x}_i>0\\ Not\ differentiable & ,\ if\ 1-y_i\mathbf{w}^T\mathbf{x}_i=0\end{matrix}\right\} \qquad \textbf{(1.7)}$$

$$=\left\{\begin{matrix}2(\mathbf{w}^T\mathbf{x}_i\mathbf{x}_{ik}-y_i\mathbf{x}_{ik}) & ,\ if\ 1-y_i\mathbf{w}^T\mathbf{x}_i<0\\ 0 & ,\ if\ 1-y_i\mathbf{w}^T\mathbf{x}_i>0\\ Not\ differentiable & ,\ if\ 1-y_i\mathbf{w}^T\mathbf{x}_i=0\end{matrix}\right\} \qquad \textbf{(1.8)}$$

$$=\mathbf{w}+\frac{\lambda}{n}\sum_{i\in I}\begin{bmatrix}2(\mathbf{w}^T\mathbf{x}_i\mathbf{x}_{i1}-y_i\mathbf{x}_{i1})\\ 2(\mathbf{w}^T\mathbf{x}_i\mathbf{x}_{i2}-y_i\mathbf{x}_{i2})\\ \vdots\\ 2(\mathbf{w}^T\mathbf{x}_i\mathbf{x}_{id}-y_i\mathbf{x}_{id})\end{bmatrix}+\frac{\lambda}{n}\sum_{i\notin I}\begin{bmatrix}0\\0\\ \vdots\\0\end{bmatrix} \qquad \textbf{(1.9)}$$

$$=\mathbf{w}+\frac{2\lambda}{n}\sum_{i\in I}\mathbf{x}_i(\mathbf{x}_i{}^T\mathbf{w}-y_i) \qquad \textbf{(1.10)}$$

$$=\mathbf{w}+\frac{2\lambda}{n}\mathbf{X}_{I,:}^T(\mathbf{X}_{I,:}\mathbf{w}-y_I) \qquad \textbf{(1.11)}$$

(2) [10 pts] Hessian

$$\nabla f(\mathbf{w})=\mathbf{w}+\frac{2\lambda}{n}\mathbf{X}_{I,:}^T(\mathbf{X}_{I,:}\mathbf{w}-y_I) \qquad \textbf{(1.12)}$$

$$\frac{d^2f(\mathbf{w})}{d\mathbf{w}^2}=\frac{d}{d\mathbf{w}}\left(\mathbf{w}+\frac{2\lambda}{n}\mathbf{X}_{I,:}^T(\mathbf{X}_{I,:}\mathbf{w}-y_I)\right) \qquad \textbf{(1.13)}$$

$$\frac{d}{d\mathbf{w}}\mathbf{w}=\begin{bmatrix}\frac{dw_1}{dw_1} & \frac{dw_1}{dw_2} & \cdots & \frac{dw_1}{dw_d}\\ \frac{dw_2}{dw_1} & \frac{dw_2}{dw_2} & \cdots & \frac{dw_2}{dw_d}\\ \vdots & \vdots & \cdots & \vdots\\ \frac{dw_d}{dw_1} & \frac{dw_d}{dw_2} & \cdots & \frac{dw_d}{dw_d}\end{bmatrix}=I_d \qquad \textbf{(1.14)}$$

$$\frac{d}{d\mathbf{w}}\left(\frac{2\lambda}{n}\mathbf{X}_{I,:}^T(\mathbf{X}_{I,:}\mathbf{w}-y_I)\right) \qquad \textbf{(1.15)}$$

$$=\frac{2\lambda}{n}\frac{d}{d\mathbf{w}}\left(\mathbf{X}_{I,:}^T(\mathbf{X}_{I,:}\mathbf{w}-y_I)\right) \qquad \textbf{(1.16)}$$

$$= \frac{2\lambda}{n}\frac{d}{d\mathbf{w}}\mathbf{X}_{I,:}^T\mathbf{X}_{I,:}\mathbf{w} \quad \textbf{(1.17)}$$

$$= \frac{2\lambda}{n}\mathbf{X}_{I,:}^T\mathbf{X}_{I,:}\frac{d}{d\mathbf{w}}\mathbf{w} \quad \textbf{(1.18)}$$

$$= \frac{2\lambda}{n}\mathbf{X}_{I,:}^T\mathbf{X}_{I,:} \quad \textbf{(1.19)}$$

$$= \frac{2\lambda}{n}\mathbf{X}^T\mathbf{D}\mathbf{X} \quad \textbf{(1.20)}$$

$$\therefore \frac{d^2 f(\mathbf{w})}{d\mathbf{w}^2} = \frac{d}{d\mathbf{w}}\mathbf{w} + \frac{d}{d\mathbf{w}}\left(\frac{2\lambda}{n}\mathbf{X}_{I,:}^T(\mathbf{X}_{I,:}\mathbf{w} - y_I)\right) = I_d + \frac{2\lambda}{n}\mathbf{X}^T\mathbf{D}\mathbf{X} \quad \textbf{(1.21)}$$

(3) [10 pts] Optimality

First, consider the objective function and such conditions.

$$\begin{aligned} &\min_{\mathbf{w},\xi} \frac{1}{2}\mathbf{w}^T\mathbf{w} + \frac{\lambda}{n}\sum_{i=1}^{n}\xi_i^2 \\ &s.t. \quad y_i\mathbf{w}^T x_i \geq 1 - \xi_i \\ &\qquad \xi \geq 0. \end{aligned} \quad \textbf{(1.22)}$$

If we take a look at the conditions, the conditions can be converted like below.

$$\xi_i \geq 1 - y_i\mathbf{w}^T x_i \quad and \quad \xi \geq 0 \quad \textbf{(1.23)}$$

Then, we can represent the conditions above as only single representation like as follows.

$$\xi_i \geq \max(1 - y_i\mathbf{w}^T x_i, 0) \quad \textbf{(1.24)}$$

If there exists the global minimum value of ξ_i, the minimum value is be the lower bound,

which is $\max(1 - y_i\mathbf{w}^T x_i, 0)$.

Finally, the problem can be represented equivalently as follows.

$$\min_{w} f(\mathbf{w}) = \frac{1}{2}\mathbf{w}^T\mathbf{w} + \frac{\lambda}{n}\sum_{i=1}^{n}\max(1 - y_i\mathbf{w}^T\mathbf{x}_i, 0)^2 \quad \textbf{(1.25)}$$

(4) [10 pts] Algorithm Pseudo Code

(Outline the pseudo code of the optimization update procedure for mini-batch stochastic gradient method and Newton method)

1) Mini-batch stochastic gradient method

```
Algorithm : Mini-batch SGD
Input : n, T;
Initialization : $w_0 \leftarrow \vec{0}$;
For k = 1, 2, …, T do
    $B \overset{unif}{\sim} \{1,\ 2,\ \ldots,\ n\}$;
    $l_i(w^{(k)}) \leftarrow l(f_w(x_i), y_i)$
    $l(w^{(k)}) \leftarrow \frac{1}{|B|}\sum_{i \in B} l_i(w^{(k)})$
    $Calculate\ \nabla l(w^{(k)})$
    $Set\ \eta_k$
    $w^{(k+1)} \leftarrow w^{(k)} - \eta_k \nabla l(w^{(k)})$
end
Output: $w^{(T+1)}$
```

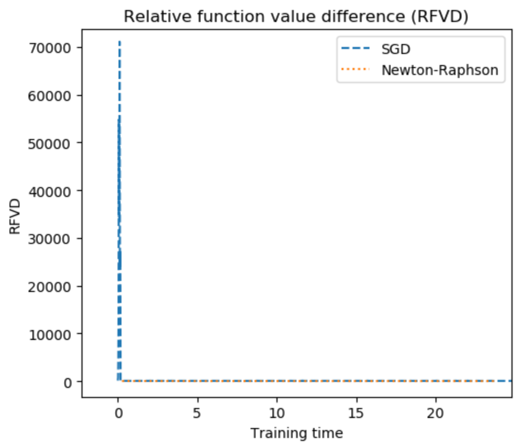
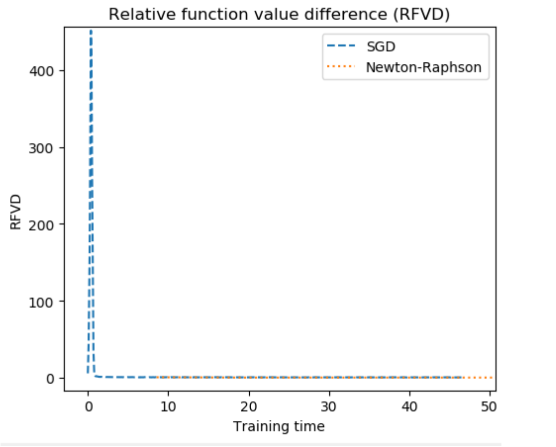
2) Newton method

```
Algorithm : Newton method
Input : n, T;
Initialization : $w_0 \leftarrow \vec{0}$;
For k = 1, 2, …, T do
    $l_i(w^{(k)}) \leftarrow l(f_w(x_i), y_i)$
    $l(w^{(k)}) \leftarrow \frac{1}{n}\sum_{i=1}^{n} l_i(w^{(k)})$
    $Calculate\ \nabla l(w^{(k)})$
    $\nabla l(w^{(k)}) \leftarrow \left( \frac{\partial}{\partial w_0} l(w^{(k)}), \frac{\partial}{\partial w_1} l(w^{(k)}),\quad , \frac{\partial}{\partial w_m} l(w^{(k)}) \right)^T$
    $Calculate\ \nabla\nabla l(w^{(k)})$
    $H(w^{(k)}) \leftarrow \nabla\nabla l(w^{(k)})$
    $w^{(k+1)} \leftarrow w^{(k)} - H(w^{(k)})^{-1} \nabla l(w^{(k)})$
end
Output: $w^{(T+1)}$
```

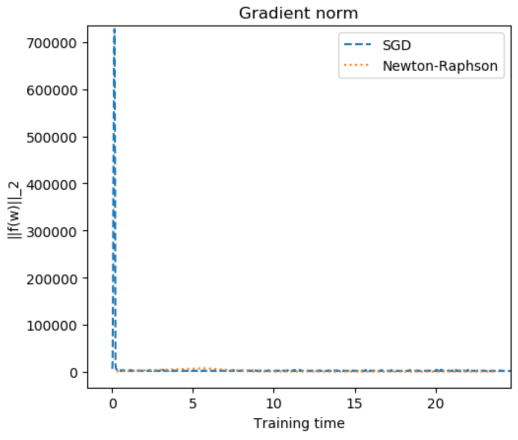
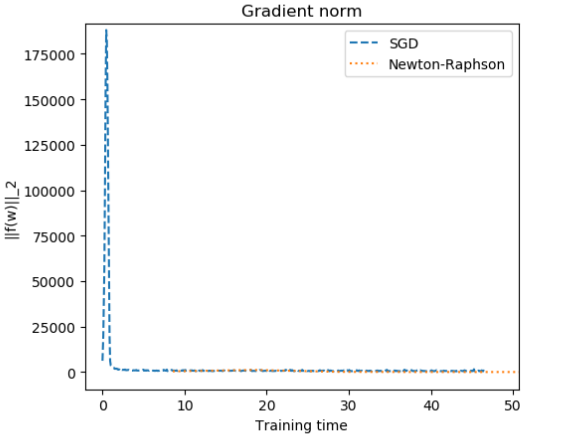
3. Experiments (20 pts)

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

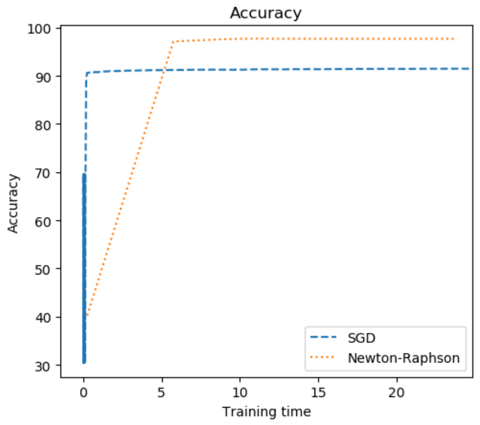
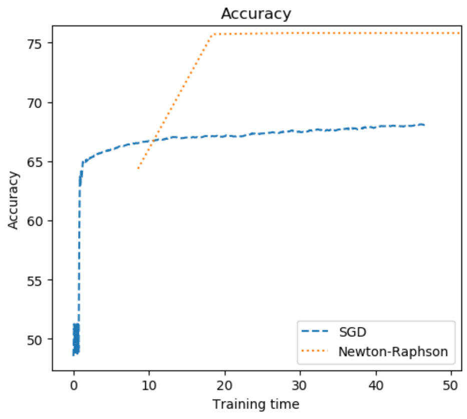
(1) [5 pts] Relative function value difference versus training time

realsim	covtype
	

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

realsim	covtype
	

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

realsim	covtype
	

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

First of all, as of experiments, Newton-Raphson method took longer time than SGD to run one epoch. It is because the SGD we used in this project is based on mini-batch. The mini-batch SGD randomly extract small dataset within the total dataset. Therefore, the mini-batch SGD tends to oscillate. We can see the tendency in all the graphs. Especially in Relative function value difference and Gradient norm, the graph of SGD oscillates with large variance when the training is started, while the Newton-Raphson does not oscillate. The most interesting things are shown in the graph of test set accuracies depicted in the last graphs above. In the accuracy graphs, the SGD converged faster than the Newton-Raphson method. Although the SGD converged faster, the Newton-Raphson method recorded higher accuracy in both Realsim and Covtype dataset. The oscillation is also appeared in both accuracy graphs at the beginning of the training.