Chih Yun Pai

1.

Part (1)

In logistic_reg, the for loop terminates either iteration times reach max_iterations or when each of element in the gradient goes below the tolerance.

[w, E_{in}] = logistic_reg(X_{train}, y_{train}, w_{init}, max_its, η)

[E_{clss_train}] = find_test_error(w, X_{train}, y_{train})

[E_{test}] = find_test_error(w, X_{test}, y_{test})

$\eta = 10^{-5}$, tolerance = 10^{-3},

where E_{in} : logistic regression in-sample error

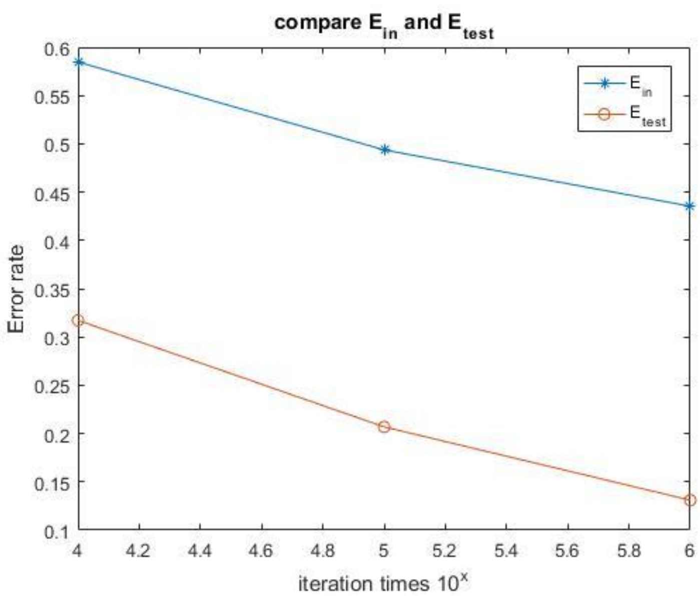
E_{clss_train} : classification error for training data set

E_{test} : test error (classification error for testing data set)

max_iterations	10,000	100,000	1,000,000
num_its	10,000	100,000	1,000,000
E_{in}	0.5847	0.4937	0.4354
E_{clss_train}	0.3092	0.2237	0.1513
E_{test}	0.3172	0.2069	0.1310
execution_time	4.9 sec	47.9 sec	477.2 sec

What can you say about the generalization properties of the model?

E_test would be generally lower than E_in, because E_in is an estimation bounded by a loose upper bound. Then the actual E_test would be lower than the error estimation in training. Furthermore, if the gradient is not below tolerance, more iterations basically come with lower error rate.



Part (2)

	Best result(max_its = 1 million)	glmfit function
E_{test}	0.1310	0.1103
Execution Time	477.2 sec	0.0403 sec

Part (3)

Scaling the features by subtracting the mean and dividing by the standard deviation for each of the features: $Z_{train} = \frac{X_{train} - \bar{X}_{train}}{S\_train}$, $Z_{test} = \frac{X_{test} - \bar{X}_{test}}{S\_test}$, where S is standard deviation.

Setting tolerance to 10^{-6}. Run iterations until each element of gradient below the tolerance 10^{-6}.

[w, E_{in}] = logistic_reg(Z_{train}, y_train, w_init, max_its = NONE, η)

[E_{test}] = find_test_error(w, Z_{test}, y_test)

	$\eta = 10^{-5}$	$\eta = 10^{-4}$	$\eta = 10^{-3}$	$\eta = 10^{-2}$
Number of iterations	23,371,190	2,337,115	233,707	23,367
E_{in}	0.4074	0.4074	0.4074	0.4074
E_{test}	0.1034	0.1034	0.1034	0.1034
Time	12859.8 sec	1086.8 sec	111.9 sec	11.2 sec

	$\eta = 10^{-1}$	$\eta = 1$	$\eta = 3$	$\eta = 5$	$\eta = 7$
Number of iterations	2,332	229	73	41	42
E_{in}	0.4074	0.4074	0.4074	0.4074	0.4074
E_{test}	0.1034	0.1034	0.1034	0.1034	0.1034
Time	1.1 sec	0.11 sec	0.035 sec	0.02 sec	0.022 sec

**when $\eta = 8$ each step is too large, it begins to bouncing around.

2. Problem 3.4

Adaptive Linear Neuron (Adaline) algorithm

(a)

$$e_n(w) = [\max(0,\ 1 - y_n w^T x_n)\ ]^2 = \begin{cases} 0, & 1 - y_n w^T x_n \le 0 \\ (y_n w^T x_n)^2, & 1 - y_n w^T x_n > 0 \end{cases}$$

when $e_n(w) = 0 \Rightarrow \lim_{(1-y_n w^T x_n)\to 0^+} \nabla e_n(w) = 0 = \lim_{(1-y_n w^T x_n)\to 0^-} \nabla e_n(w)$

$\Rightarrow \nabla e_n(w) = 0$

when $e_n(w) = (y_n w^T x_n)^2$

$$\lim_{(1-y_n w^T x_n)\to 0^+} \nabla e_n(w) = \lim_{(1-y_n w^T x_n)\to 0^+} -2[1-y_n w^T x_n]y_n x_n = 0$$

$$\lim_{(1-y_n w^T x_n)\to 0^-} \nabla e_n(w) = \lim_{(1-y_n w^T x_n)\to 0^-} -2[1-y_n w^T x_n]y_n x_n = 0$$

$$\Rightarrow \lim_{(1-y_n w^T x_n)\to 0^+} \nabla e_n(w) = \lim_{(1-y_n w^T x_n)\to 0^-} \nabla e_n(w) = 0$$

$\Rightarrow \nabla e_n(w) = 0$

Therefore, $e_n(w)$ is continuous and differentiable.

(b)

$[\![sign(w^T x_n) \neq y_n]\!] = \begin{cases} 0, & sign(w^T x_n) = y_n \\ 1, & sign(w^T x_n) \neq y_n \end{cases}$

$e_n(w) = [\max(0,\ 1 - y_n w^T x_n)\ ]^2 \begin{cases} 0, & 1-y_n w^T x_n \le 0 \\ (y_n w^T x_n)^2, & 1-y_n w^T x_n > 0 \end{cases}$

when $sign(w^T x_n) = y_n$, $[\![sign(w^T x_n) \neq y_n]\!] = 0$

$y_n w^T x_n > 0 \Rightarrow e_n(w) = [\ 0, 1\ ) \geq [\![sign(w^T x_n) \neq y_n]\!]$

when $sign(w^T x_n) \neq y_n$, $[\![sign(w^T x_n) \neq y_n]\!] = 1$

$y_n w^T x_n < 0 \Rightarrow e_n(w) > 1 = [\![sign(w^T x_n) \neq y_n]\!]$

Since for each n, $e_n(w) \geq [\![sign(w^T x_n) \neq y_n]\!]$, therefore $e_n(w)$ is upper bound of $[\![sign(w^T x_n) \neq y_n]\!]$.

$\Rightarrow \frac{1}{N}\sum_{n=1}^{N} e_n(w) \geq \frac{1}{N}\sum_{n=1}^{N}[\![sign(w^T x_n) \neq y_n]\!] = E_{in}(w)$

(c)

w(t+1) = w(t) + $\eta(1 - y_n w^T x_n)y_n x_n$

= w(t) + $\eta(y_n - (y_n)^2 w^T x_n)x_n$

= w(t) + $\eta(y_n - (1)w^T x_n)x_n$

= w(t) + $\eta(y_n - w^T x_n)x_n$

3. Problem 3.19

Pointing out potential problems

(a)

The potential problem is the complexity goes infinite as n goes infinite.

(b)

Same reason as part (a).

The potential problem is the complexity goes infinite as n goes infinite.

(c)

$\Phi(\mathrm{x}) = \{\phi_{i,j}\} \in \mathcal{R}^{(101\times101)}$

Since the transformation's dimension is fixed and finite, there's no potential problem that complexity would go too large.

4. Problem 4.8

$$\begin{cases} w^T\mathrm{w} = \sum_{i=1}^{n} w_i^2 \\ \nabla\mathrm{w} = (2w_1, 2w_2, \dots, 2w_n) = 2w \end{cases}$$

$E_{aug}(w) = E_{in}(w) + \lambda w^T\Gamma^T\Gamma\mathrm{w} = E_{in}(w) + \lambda w^T\mathrm{w}$

$E_{aug}(w) : w(t+1) \leftarrow w(t) - \eta[\nabla E_{in}(w(t)) + 2\lambda w(t)]$

$\Rightarrow \mathrm{w(t+1)} \leftarrow (1-2\eta\lambda)w(t) - \eta\nabla E_{in}(w(t))$

5. Problem 4.25

(a)

No, because $E_{out} = E_{in}(g) + O(\sqrt{\frac{d}{N}\ln(N)})$,

and for each leaner, $E_{out} = E_{val}(m) + O(\sqrt{\frac{d}{N-K_m}\ln(N-K_m)})$

Since the second term varies in each m leaner, selecting the smallest $E_{val}(m)$ doesn't guarantee the minimum E_{out}.

(b)

Yes, because for each leaner, $E_{out} = E_{val}(m) + O(\sqrt{\frac{d}{N-K_m}\ln(N-K_m)})$

For every m learner, K_m are all the same, then second term is the same.
Therefore, selecting minimum validation error will lead a minimum E_{out}.

(c)

Each $\mathrm{P}[E_{out}(m) > E_{val}(m) + \epsilon] \leq \mathrm{e}^{-2\epsilon^2 K_m}$

$\mathrm{P}[E_{out} > E_{val} + \epsilon] \leq \sum_{m=1}^{M} \mathrm{P}[E_{out}(m) > E_{val}(m) + \epsilon]$

$= \sum_{m=1}^{M} \mathrm{e}^{-2\epsilon^2 K_m} = M\,\mathrm{e}^{-2\epsilon^2 k(\epsilon)}$

where $k(\epsilon) = \frac{-1}{2\epsilon^2}\ln(\frac{1}{M}\sum_{m=1}^{M} \mathrm{e}^{-2\epsilon^2 K_m})$