465627
Zhenbang Guo

Homework 2

1.

Ten thousand iterations:

$$E_{in} = 0.5847$$
$$TestError = 0.3172$$

One hundred thousand iterations:

$$E_{in} = 0.4937$$
$$TestError = 0.2069$$

One million iterations:

$$E_{in} = 0.4354$$
$$TestError = 0.1310$$

It seems like our hypothesis set H contain target function since Test Error is smaller as iteration time increase, but H looks like too big to achieve target function in one million iterations, under current learning rate.

Using $glmfit$

$$TestError = 0.1103$$

$glmfit$'s running time is as fast as ten thousand iterations in previous method, and achieved 0.1103 Test Error, better than the best in previous.

Scale data result

$\eta = 6.5,\ iteration = 28,\ E_{in} = 0.4074$

2 LFD 2.22

We know that:

$$E_D[E_{out}(g^{(D)})] = E_x[E_D[(g^{(D)}(x) - \bar{g}(x))^2] + (\bar{g}(x) - y(x))^2]$$

Then replace $y(x)$ with $f(x) + \epsilon$:

$$\begin{aligned}
E_D[E_{out}(g^{(D)})] &= E_x[E_D[(g^{(D)}(x) - \bar{g}(x))^2] + (\bar{g}(x) - f(x))^2 + \epsilon^2 - 2\epsilon(\bar{g}(x) - f(x))] \\
&= E_x[var(x) + bias(x) + \epsilon^2 - 2\epsilon(\bar{g}(x) - f(x))] \\
&= var + bias + E_x[\epsilon^2 - 2\epsilon(\bar{g}(x) - f(x))] \\
&= var + bias + E_\epsilon[E_{x|\epsilon}[\epsilon^2 - 2\epsilon(\bar{g}(x) - f(x))|\epsilon]] \\
&= var + bias + E_\epsilon[\epsilon^2 - 2\epsilon(\bar{g}(x) - f(x))] \\
&= var + bias + E_\epsilon[\epsilon^2] - 2E_\epsilon[\epsilon(\bar{g}(x) - f(x))] \\
&= var + bias + var(\epsilon) - (E_\epsilon(\epsilon))^2 - 2E_\epsilon[\epsilon](\bar{g}(x) - f(x))
\end{aligned}$$

Since $E_\epsilon[\epsilon] = 0$

$$E_D[E_{out}(g^{(D)})] = var + bias + \sigma^2$$

Proved.

3 LFD 2.24

(a)

$$\begin{aligned}
a &= \frac{y_2 - y_1}{x_2 - x_1} \\
b &= y_1 - \frac{y_2 - y_1}{x_2 - x_1}x_1 \\
&= \frac{y_1x_2 + y_2x_1}{x_2 - x_1} \\
\bar{g}(x) &= E_D[\frac{x_2^2 - x_1^2}{x_2 - x_1}x + \frac{x_1^2x_2 - x_2^2x_1}{x_2 - x_1}]
\end{aligned}$$

Since $x_1,\ x_2$'s density function is $\frac{1}{2}$, $\frac{1}{2}$, respectively, then:

$$\begin{aligned}\bar{g}(x) &= \frac{1}{4}\int_{-1}^{1}\int_{-1}^{1} x_1 + x_2 dx_1 dx_2 x + \frac{1}{4}\int_{-1}^{1}\int_{-1}^{1} x_1 x_2 dx_1 dx_2 x \\ &= 0\end{aligned}$$

(b)

For n times, we generate $x_1,\ x_2$ from $[-1, 1]$, calculate its $g^{D_n}(x)$ and save it to a vector, after n times, we calculate $\bar{g}(x) = \frac{1}{n}\sum_{i=1}^{n}(g^{D_i})$, calculate $var(x) = \frac{1}{n}\sum_{i=1}^{n}(g^{D_i}(x) - \bar{g}(x))^2$, calculate $bias(x) = (\bar{g}(x) - f(x))^2$, calculate $E[E_{out}(g^D(x))] = \frac{1}{n}\sum_{i=1}^{n}(g^{D_i}(x) - f(x))^2$, then we generate total m points from $U[-1, 1]$, plug in those function and calculate mean value of $var(x),\ bias(x),\ E[E_{out}(g^D(x))]$ as $var,\ bias,\ and\ E[E_{out}(g^D)]$.

(c)

$E[E_{out}] = bias + var$, based on the experiment result.

$$\begin{aligned}E[E_{out}] &= 0.5251 \\ bias &= 0.1963 \\ var &= 0.3289\end{aligned}$$

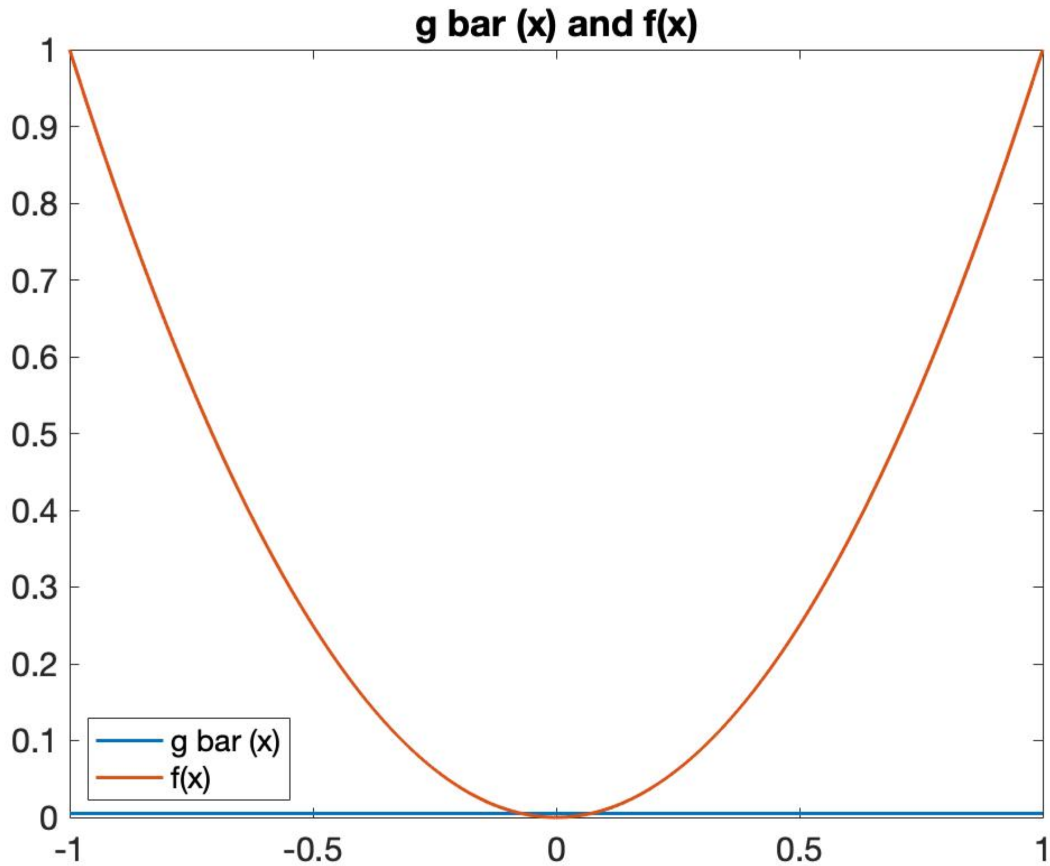

Figure 1: $\bar{g}(x)$ *and* $f(x)$

(d)

$$
\begin{aligned}
E_{out} &= E[g(x) - f(x)] = E[(ax + b - x^2)^2] \\
&= E[x^4] - 2aE[x^3] + (a^2 - 2b)E[x^2] + 2abE[X] + b^2 \\
&= \frac{1}{2}\int_{-1}^{1} x^4 dx - 2a \times \frac{1}{2}\int_{-1}^{1} x^3 dx + \frac{1}{2}(a^2 - 2b)\int_{-1}^{1} x^2 dx + ab\int_{-1}^{1} x dx + b^2 \\
&= \frac{1}{5} + \frac{1}{3}(a^2 - 2b) + b^2
\end{aligned}
$$

Then, take expectation of E_{out} with $a,\ b$ replace by $x_1,\ x_2$ in the (a):

$$\begin{aligned}
E[E_{out}] &= \frac{1}{5} + \frac{1}{3}E[(x_1+x_2)^2 + 2x_1x_2] + E[x_1^2x_2^2] \\
&= \frac{1}{3}\frac{1}{4}\int_{-1}^{1}\int_{-1}^{1}(x_1+x_2)^2 + 2x_1x_2dx_1dx_2 + \frac{1}{4}\int_{-1}^{1}\int_{-1}^{1}x_1^2x_2^2dx_1dx_2 \\
&= \frac{8}{15}
\end{aligned}$$

For $bias$:

$$\begin{aligned}
bias = E[bias(x)] &= E[(\bar{g}(x) - f(x))^2] = E[x^4] \\
&= \frac{1}{2}\int_{-1}^{1}x^4dx = \frac{1}{5}
\end{aligned}$$

For $var(x)$:

$$\begin{aligned}
var(x) &= E[(g(x) - \bar{g}(x))^2] = E[a^2x^2 + 2abx + E[b^2]] \\
&= E[a^2]x^2 + 2E[ab]x + b^2 \\
&= \frac{x^2}{4}\int_{-1}^{1}\int_{-1}^{1}(x_1+x_2)^2dx_1dx_2 + 2\times\frac{x}{4}\int_{-1}^{1}\int_{-1}^{1}(x_1+x_2)(-x_1x_2)dx_1dx_2 + \frac{x}{4}\int_{-1}^{1}\int_{-1}^{1}x_1^2x_2^2dx_1dx_2 \\
&= \frac{2}{3}x^2 + \frac{1}{9}
\end{aligned}$$

Then var is :

$$\begin{aligned}
var = E[var(x)] &= E[(\frac{2}{3}x^2 + \frac{1}{9})] \\
&= \frac{2}{3}\times\frac{1}{2}\int_{-1}^{1}x^2dx + \frac{1}{9} \\
&= \frac{1}{3}
\end{aligned}$$

4 LFD Exercise 3.4

(a)

$$\begin{aligned}\hat{y} = Hy &= HXw^* + H\epsilon \\ &= X(X^TX)^{-1}(X^TX)w^* + H\epsilon \\ &= Xw^* + H\epsilon\end{aligned}$$

Proved.

(b)

$$\begin{aligned}\hat{y} - y &= Xw^* + H\epsilon - Xw^* - \epsilon \\ &= (H - I)\epsilon\end{aligned}$$

So the matrix is $(H - I)$ where I is identity matrix with size of N.

(c)

$$\begin{aligned}E_{in}(w_{lin}) &= \frac{1}{N}(\hat{y} - y)^T(\hat{y} - y) \\ &= \frac{1}{N}(I - H)^2\epsilon^2\end{aligned}$$

By $(I - H)^k = I - H$:

$$E_{in}(w_{lin}) = \frac{1}{N}(I - H)\epsilon^2$$

5 LFD 3.4

(a)

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

It is continuous since:

$$\lim_{w:y_n w^T x_n \to 1} e_n(w) = 0$$

It is differentiable since:

$$\triangledown e_n(w) = \begin{cases} 0 & , \ y_n w^T x_n > 1 \\ -2y_n(1 - y_n w^T x_n)x_n & , \ y_n w^T x_n < 1 \end{cases}$$

(b)

If $[sign(w^T x_n) \neq y_n] = 1$, then we know that $y_n w^T x_n \leq 0 < 1$, so in this case:

$$[sign(w^T x_n) \neq y_n] = 1 \leq e_n(w)$$

Similarly, if $[sign(w^T x_n) \neq y_n] = 0$, then $y_n w^T x_n \geq 0$, which means $e_n(w) = 0$ if $y_n w^T x_n < 1$; $e_n(w) > 1$ if $y_n w^T x_n > 1$, so in conclusion:

$$[sign(w^T x_n) \neq y_n] \leq e_n(w)$$

Hence, $\frac{1}{N} \sum_{n=1}^{N} e_n(w)$ is an upper bound for the in-sample classification error $E_{in}(w)$.

(c)

Use this bound for gradient descent is exactly Adaline algorithm.
If $y_n s_n = y_n w^T x_n <= 1$, then update w:

$$w \leftarrow w - \eta \triangledown e_n(w) = w - 2\eta y_n(1 - y_n w^T x_n)x_n = w - \eta'(y_n - w^T x_n)x_n$$

Where $s_n = w^T x_n$, and if $y_n s_n = y_n w^T x_n > 1$, do nothing, just as Adaline algorithm.

6 LFD 3.19

(a)

This transform will increase feature dimension to N, which will obviously slow down computation. At the same time, it will have $d_{vc} = N + 1$, which will have far more weaker generalization ability.

(b)

It seems good, but I worry that if γ is intensively small, feature itself will be smoked by very small γ.

(c)

It seems good.