1. (a)

Max Iterations	E_{in}	Binary E_{in}	Binary E_{out}	Time (s)
10^4	0.586	0.309	0.317	0.24
10^5	0.469	0.224	0.207	2.18
10^6	0.436	0.151	0.131	21.48

The logistic regression model generalizes well. The E_{in} and E_{out} are consistently very similar indicating that there is enough data for learning to occur, at least with this simple model. Increasing the maximum number of iterations, at least to up to 10^6, continues to improve the model's performance.

(b)

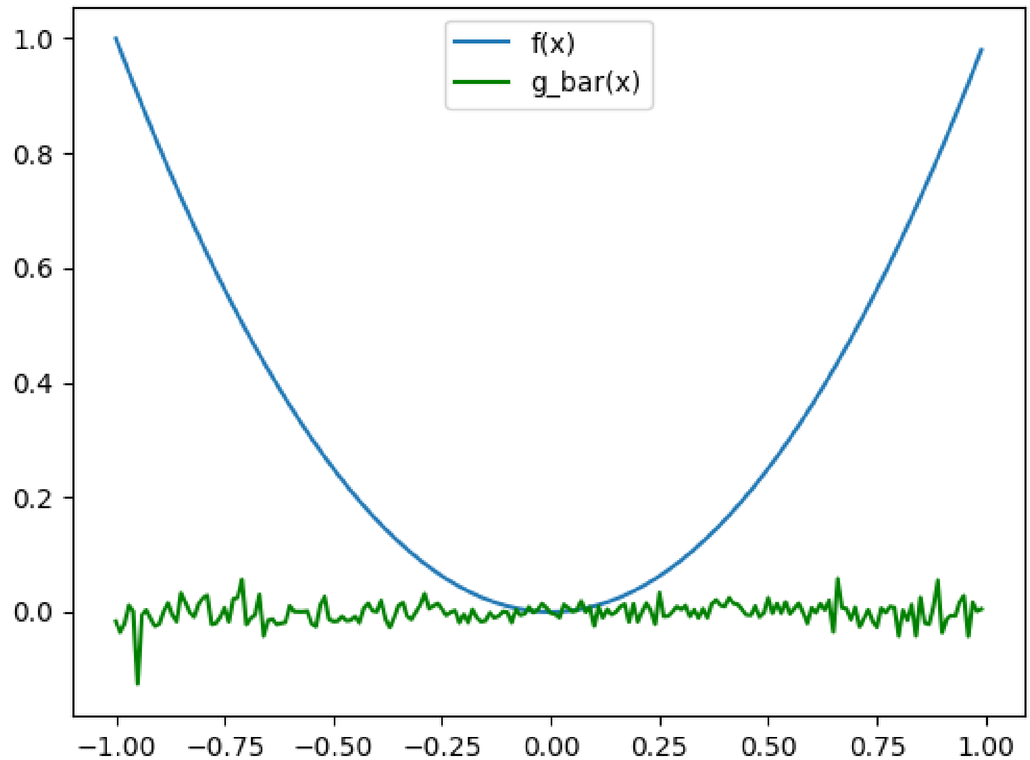
Learning Rate	E_{in}	Binary E_{in}	Binary E_{out}	Iterations	Time (s)
0.01	0.69	0.171	0.11	22429	0.49
0.1	0.663	0.171	0.11	2239	0.05
1	0.487	0.171	0.11	220	0.01
4	0.522	0.171	0.11	51	0.003
7	0.767	0.171	0.11	44	0.00299
7.5	0.813	0.171	0.11	180	0.00601
7.6	0.822	0.171	0.11	446	0.012
7.7	0.408	0.164	0.117	1000000	22.03

I normalized both the training and testing set to the mean and variance of the training set. This is because we never want to include the testing data in training.

Normalizing the data increases the performance of the model. Using the same parameters without normalized data yields E_{in} and E_{out} (binary) approaching random guessing.

Changing the learning rate usually arrives at the same error but just at different rates. The smallest learning rate takes > 20000 iterations to arrive at the same prediction model as does a learning rate of 7 (which takes just 44). Too big of a learning rate becomes cumbersome very quickly however. Increasing the learning rate to 7.6 makes us take 446 iterations and 7.7 makes us max out the number of iterations at 10^6!

3 (c) plot.



2.

Problem 2.22 With noise $E_{out}(g^{(D)}) = \mathbb{E}_{x,y}[(g^{(D)}(\vec{x}) - y(\vec{x}))^2]$

where $y(\vec{x}) = f(\vec{x}) + \varepsilon$ (ε zero mean w/var σ^2)

Show bias-variance decomp becomes

$$\mathbb{E}_D[E_{out}(g^{(D)})] = \sigma^2 + \text{bias} + \text{var}.$$

- Redo book derivation with

$$\mathbb{E}_D[\mathbb{E}_{out}(g^{(D)})] = \mathbb{E}_D[\mathbb{E}_{xy}[(g^{(D)}(\vec{x}) - (f(\vec{x}) + \varepsilon))^2]]$$

$$= \mathbb{E}_D[\mathbb{E}_{xy}[g^{(D)}(\vec{x})^2 - 2g^{(D)}(\vec{x})(f(\vec{x}) + \varepsilon) + (f(\vec{x}) + \varepsilon)^2]]$$

- switch $\mathbb{E}_D$ and $\mathbb{E}_{xy}$, distribute $\mathbb{E}_D$

$$= \mathbb{E}_{xy}[\mathbb{E}_D[g^{(D)}(\vec{x})^2] - 2\mathbb{E}_D[g^{(D)}(\vec{x})](f(\vec{x}) + \varepsilon) + (f(\vec{x}) + \varepsilon)^2]$$

- add in a $-\bar{g}(\vec{x})^2 + \bar{g}(\vec{x})^2$

$$= \mathbb{E}_{xy}[\mathbb{E}_D[g^{(D)}(\vec{x})^2] - \bar{g}(\vec{x})^2 + \bar{g}(\vec{x})^2 - 2\bar{g}(\vec{x})(f(\vec{x}) + \varepsilon) + (f(\vec{x}) + \varepsilon)^2]$$

$$= \mathbb{E}_{xy}[\mathbb{E}_D[g^{(D)}(\vec{x})^2 - \bar{g}(\vec{x})^2] + \underline{\bar{g}(\vec{x})^2} - \underline{2\bar{g}(\vec{x})f(\vec{x})} - 2\bar{g}(\vec{x})\varepsilon + \underline{f(\vec{x})^2} + 2f(\vec{x})\varepsilon + \varepsilon^2]$$

$$= \mathbb{E}_{xy}[\text{variance} + \bar{g}(\vec{x})^2 - 2\bar{g}(\vec{x})f(\vec{x}) + f(\vec{x})^2 - 2\bar{g}(\vec{x})\varepsilon + 2f(\vec{x})\varepsilon + \varepsilon^2]$$

$$= \mathbb{E}_{xy}[\text{variance} + (\bar{g}(\vec{x}) - f(\vec{x}))^2 + \varepsilon^2 + (2f(\vec{x}) - \bar{g}(\vec{x}))\varepsilon]$$

$$= \text{Variance} + \text{bias} + \mathbb{E}_{xy}[\varepsilon^2] - 2\mathbb{E}_{xy}[(\bar{g}(\vec{x}) - f(\vec{x}))\varepsilon]$$

$$= \boxed{\text{variance} + \text{bias} + \sigma^2}$$

this is 0 b/c $\mathbb{E}[\varepsilon] = 0$

3. 2.24 $\quad x \in [-1, 1], \quad f(x) = x^2$

$$D = \{(x_1, x_1^2), (x_2, x_2^2)\}$$

$$h \in \{h(x) = ax+b\}$$

(a) Give an expression for $\bar{g}(x)$.) ok

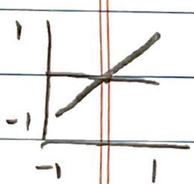
$$a = \frac{x_2^2 - x_1^2}{x_2 - x_1} \qquad b = \frac{x_2 x_1^2 - x_1 x_2^2}{x_2 - x_1}$$

Find $\mathbb{E}_D[ax+b]$

$$\bar{g}(x) = \mathbb{E}_D\left[\frac{x_2^2 - x_1^2}{x_2 - x_1} x + \frac{x_2 x_1^2 - x_1 x_2^2}{x_2 - x_1}\right]$$

$$= x \frac{1}{2}\frac{1}{2} \int_{-1}^{1}\int_{-1}^{1} \frac{x_2^2 - x_1^2}{x_2 - x_1} dx_1 dx_2 + \frac{1}{2}\frac{1}{2} \int_{-1}^{1}\int_{-1}^{1} \frac{x_1 x_2^2 - x_2 x_1^2}{x_2 - x_1} dx_1 dx_2$$

$$= \frac{x}{4} \int_{-1}^{1}\int_{-1}^{1} \frac{\cancel{(x_2 - x_1)}(x_2 + x_1)}{\cancel{x_2 - x_1}} dx_1 dx_2 + \frac{1}{4} \int_{-1}^{1}\int_{-1}^{1} \frac{- x_1 x_2 \cancel{(x_2 - x_1)}}{\cancel{x_2 - x_1}} dx_1 dx_2$$



$$= \frac{x}{4} \iint_{-1}^{1} x_2 + x_1 \, dx_1 dx_2 + \frac{1}{4} \int_{-1}^{1}\int_{-1}^{1} x_1 x_2 \, dx_1 dx_2$$

$$= \frac{x}{4} 0 + 0 + \frac{1}{4} 0 0 = \boxed{0}$$

(b) Numerically determine $\bar{g}(x)$, $\mathbb{E}[E_{out}]$, bias & Var

- $\bar{g}(x)$: generate N datasets $((x_1, x_1^2), (x_2, x_2^2))$
  - Calculate $a = \frac{1}{N}\sum_{n=1}^{N} (x_1 + x_2)$
    $b = \frac{1}{N}\sum_{n=1}^{N} -x_1 x_2$
- bias, Var, $\mathbb{E}[E_{out}]$:
  - sample N datasets, calculate $g^{D}(x)$ and $\bar{g}(x)$ (many, one)
  - resample above M times
  - Variance: $\mathbb{E}_x[\mathbb{E}_D[(g^D(x) - \bar{g}(x))^2]]$

$$= \boxed{\frac{1}{M}\frac{1}{N}\sum^{M,N}(g_N^D(x) - \bar{g}_M(x))^2}$$

- Bias: $\mathbb{E}_x[\mathbb{E}_D[(\bar{g}(x) - f(x))^2]]$

Numerically: $\frac{1}{MN}\sum_1^M \sum_1^N (\bar{g}_m(x) - f_{m,n}(x))^2$

- $\mathbb{E}[E_{out}]$: $\mathbb{E}_x[\mathbb{E}_D[(g^D(x) - f(x))^2]]$

Numerically: $\frac{1}{MN}\sum_1^M \sum_1^N (g^D_{m,n}(x) - f_{m,n}(x))^2$

(c) Variance $= 0.336$

Bias $= 0.207$

$\mathbb{E}[E_{out}] = 0.544$

Var + bias $= 0.543$

pretty close (figure on next page)

(d) Analytically compute $\mathbb{E}[E_{out}]$, bias, var.

$$\text{bias} = \mathbb{E}_x[\mathbb{E}_D[(\bar{g}(x) - f(x))^2]]$$
$$= \mathbb{E}_x[(0 - f(x))^2]$$
$$= \mathbb{E}_x[f(x)^2] = \mathbb{E}_x[x^4] = \frac{1}{2}\int_{-1}^{1} x^4 dx = \frac{1}{2}\frac{x^5}{5}\Big|_{-1}^{1}$$

$$\text{bias} = \frac{1}{2}\frac{2}{5} = \boxed{1/5}$$

$$\text{Variance} = \mathbb{E}_x[\mathbb{E}_D[(g^D(x) - \underbrace{\bar{g}(x)}_{\to 0})^2]]$$
$$= \mathbb{E}_x[\mathbb{E}_D[(ax+b)^2]] = \mathbb{E}_x[\mathbb{E}_D[((x_1+x_2)x - x_1x_2)^2]]$$
$$= \mathbb{E}_x[\mathbb{E}_D[(x_1+x_2)^2x^2 - 2x_1x_2(x_1+x_2)x + x_1^2x_2^2]]$$
$$= \mathbb{E}_x[x^2\mathbb{E}_D[(x_1+x_2)^2] - 2x\mathbb{E}_D[x_1x_2(x_1+x_2)] + \mathbb{E}_D[x_1^2x_2^2]]$$
$$= \mathbb{E}_x[x^2\mathbb{E}_D[x_1^2 + x_2^2 + 2x_1x_2] - 2x\mathbb{E}_D[x_1^2x_2 + x_2^2x_1] + \mathbb{E}_D[x_1^2x_2^2]]$$

$$\mathbb{E}_D[x_1] = 0, \quad \mathbb{E}_D[x_1^2] = \frac{1}{2}\int_{-1}^{1} x^2 dx = \frac{1}{2}\frac{x^3}{3}\Big|_{-1}^{1} = \frac{1}{2}\frac{2}{3} = \frac{1}{3}$$

$$= \mathbb{E}_x[x^2(\tfrac{1}{3}+\tfrac{1}{3}) - 2x(\tfrac{1}{3}(0) + \tfrac{1}{3}(0)) + (\tfrac{1}{3}\tfrac{1}{3})]$$
$$= \frac{1}{3}\left(\frac{2}{3}\right) - 0 + \frac{1}{9} = 3/9 = \boxed{1/3} = \text{Variance}$$

(d) cont.

$$\mathbb{E}[E_{out}] = \mathbb{E}_x[\mathbb{E}_D[(g(x)-f(x))^2]]$$ (forget about $\mathbb{E}_D$ for a sec)

$$= \mathbb{E}_x[(ax+b-x^2)^2] = \mathbb{E}_x[x^4 - 2x^2(ax+b) + (ax+b)^2]$$

$$= \mathbb{E}_x[x^4] - 2a\mathbb{E}_x[x^3] - 2b\mathbb{E}_x[x^2] + a^2\mathbb{E}[x^2] + 2ab\mathbb{E}[x] + b^2$$

$$= \frac{1}{5} - 0 - 2b\left(\frac{1}{3}\right) + a^2\left(\frac{1}{3}\right) + b^2$$

$$= \frac{1}{5} + \frac{1}{3}(a^2 - 2b) + b^2$$

$$= \mathbb{E}_D\left[\frac{1}{5} + \frac{1}{3}(a^2-2b) + b^2\right] \qquad a = x_1 + x_2,\ b = -x_1x_2$$

$$= \frac{1}{5} + \frac{1}{3}\mathbb{E}_D[(x_1+x_2)^2 + 2x_1x_2] + \mathbb{E}_D[(x_1x_2)^2]$$

$$= \frac{1}{5} + \frac{1}{3}\mathbb{E}_D[x_1^2 + 2x_1x_2 + x_2^2 + 2x_1x_2] + \mathbb{E}_D[x_1^2x_2^2]$$

$$= \frac{1}{5} + \frac{1}{3}\left(\frac{1}{3} + 0 + \frac{1}{3} + 0\right) + \left(\frac{1}{3}\right)\frac{1}{3}$$

$$= \frac{1}{5} + \frac{2}{9} + \frac{1}{9} = \boxed{8/15} = \mathbb{E}[E_{out}]$$ (also equals bias + var)

4. 3.4 Adaptive Linear Neuron (adaline)

(a) $e_n(\vec{\omega}) = \left(\max(0, 1 - y_n \vec{\omega}^T \vec{x})\right)^2$

- show $e_n(\vec{\omega})$ is continious and differentiable.
- Write down $\nabla e_n(\vec{\omega})$

- 0 is continious and differentiable $(1 - y_n \vec{\omega}^T \vec{x} < 0)$
- $(1 - y_n \vec{\omega}^T \vec{x})^2$ is continious and differentiable $(1 - y_n \vec{\omega}^T \vec{x} > 0)$
- if $1 - y_n \vec{\omega}^T \vec{x} = 0$, then $y_n = \vec{\omega}^T \vec{x}$ and there is no error!
  - $\rightarrow e_n(\vec{\omega}) = 0$
  - $\rightarrow$ so this function is continious between both its pieces

- $\nabla e_n(\vec{\omega}) = 2(1 - y_n \vec{\omega}^T \vec{x})(-y_n \vec{x})$
- This is 0 at $1 - y_n \vec{\omega}^T \vec{x}$.
  - $\rightarrow$ so this function is differentiable between both its pieces.

(b) Show $e_n(\vec{\omega})$ is an upper bound for $[\![\text{sign}(\vec{\omega}^T \vec{x}_n) \neq y_n]\!]$

- Two cases:
  1. $\text{sign}(\vec{\omega}^T \vec{x}_n) = y_n$
     - here $y_n \vec{\omega}^T \vec{x} \geq 0$, $[\![\text{sign}(\vec{\omega}^T \vec{x}_n) \neq y_n]\!] = 0$,
     - $e_n(\vec{\omega}) \geq 0$ if $y_n \vec{\omega}^T \vec{x} < 1$
     - $e_n(\vec{\omega}) = 0$ if $y_n \vec{\omega}^T \vec{x} \geq 1$
     - Thus: $[\![\text{sign}(\vec{\omega}^T \vec{x}_n) \neq y_n]\!] \leq e_n(\vec{\omega})$ (in this case)
  2. $\text{sign}(\vec{\omega}^T \vec{x}_n) \neq y_n$
     - here $y_n \vec{\omega}^T \vec{x} \leq 0$, $[\![\text{sign}(\vec{\omega}^T \vec{x}_n) \neq y_n]\!] = 1$
     - $e_n(\vec{\omega}) \geq 1$
     - Thus $[\![\text{sign}(\vec{\omega}^T \vec{x}_n) \neq y_n]\!] \leq e_n(\vec{\omega})$ (in both cases now)

Thus $\boxed{E_{in}(\vec{\omega}) = \frac{1}{N} \sum_{n=1}^{N} [\![\text{sign}(\vec{\omega}^T \vec{x}_n) \neq y_n]\!] \leq \frac{1}{N} \sum_{n=1}^{N} e_n(\vec{\omega})}$

(c) Argue Adaline algorithm performs stochastic gradient descent on $\frac{1}{N}\sum_{n=1}^{N} e_n(\vec{w})$.

$$s(t) = \vec{w}^T(t)\,\vec{x}(t)$$

$$\vec{w}(t+1) \leftarrow \vec{w}(t) - \eta \nabla e_n(\vec{w})$$ gradient descent

$$\vec{w}(t+1) \leftarrow \vec{w}(t) - \eta\left(2(1-y_n\vec{w}^T\vec{x})(-y_n\vec{x})\right)$$

$$\leftarrow \vec{w}(t) - 2\eta\,(1-y_n s(t))(-y_n\vec{x})$$

$$\leftarrow \vec{w}(t) + 2\eta\,(y_n - \cancel{y_n^2} s(t))\,\vec{x}$$

$$\leftarrow \vec{w}(t) + 2\eta\,(y_n - s(t))\,\vec{x}$$

$$\boxed{\vec{w}(t+1) \leftarrow \vec{w}(t) + 2\eta\,(y(t) - s(t))\,\vec{x}(t)}$$

This is Adaline algorithm (subsistute $\eta = 2\eta$).

5. 3.19 Is learning feasible?

(a) $\Phi(\vec{x}) = \begin{cases} (0, \dots 0, 1, 0, \dots) & \text{if } \vec{x} = \vec{x}_n \\ (0, 0 \dots 0) & \text{otherwise} \end{cases}$

This maps points of dimension 2 to dimension N (where N is total number of points). This dramatically increases the generazization error as N increases.

$$E_{out}(g) \leq E_{in}(g) + O\left(\sqrt{d_{vc} \frac{\ln N}{N}}\right)$$

here d_{vc} will grow with N, so the generalization error increases with N ☹

(b) $\phi_n(\vec{x}) = \exp\left(-\frac{\|\vec{x} - \vec{x}_n\|^2}{2\gamma^2}\right)$

This seems ok. It seems to just normaliz $\vec{x}_n$ to $\vec{x}$ an a standard $2\gamma^2$. Looks like the polar form of something. Since the dimensionality stays the same after the transform, the VC dimension should stay the same and generalization should decrease with N.

(c) $\phi_{i,j}(\vec{x}) = \exp\left(\frac{-\|\vec{x} - (i,j)\|^2}{2\gamma^2}\right)$ $i \in (0, \frac{1}{100}, \dots)$ $j \in (0, \frac{1}{100}, \dots)$

This converts every datapoint of dimension 2 to dimension 101×101. Dratimically increasing the dimensionality of the dataset. This gives us the same problem as in (a).