Name: CHAN King Yeung
SID: 1155119394
CSCI3230 Assignment 1

---

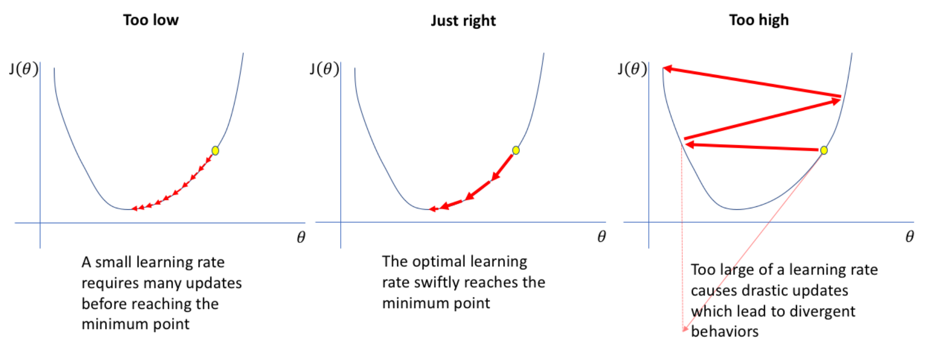
a) $$\begin{aligned} O &= f(\boldsymbol{w}\cdot\boldsymbol{I}) \\ &= f\left(\textstyle\sum_{j=0}^{n} w_j \cdot I_j\right) \end{aligned}$$

b) $$\begin{aligned} h_{i,k} &= f(\boldsymbol{w}\cdot\boldsymbol{h}) \\ &= f\left(\textstyle\sum_{j=0}^{H_{i-1}} w_{i-1,j,k}\cdot h_{i-1,j}\right) \end{aligned}$$

$$\begin{aligned} O_m &= f(\boldsymbol{w}\cdot\boldsymbol{h}) \\ &= f\left(\textstyle\sum_{j=0}^{H_K} w_{j,k,m}\cdot h_{j,k}\right) \end{aligned}$$

c) $$\begin{aligned} f'(z) &= -1\cdot(1+e^{-z})^{-2}\cdot e^{-z} \\ &= \frac{1}{1+e^{-z}}\left(\frac{e^{-z}+1-1}{1+e^{-z}}\right) \\ &= \frac{1}{1+e^{-z}}\left(1-\frac{1}{1+e^{-z}}\right) \\ &= f(z)[1-f(z)] \end{aligned}$$

d) Learning rate α is a tuning parameter in gradient descent which decides the step size at each iteration to search through the whole weight space. It ranges in between 0 and 1. Smaller α requires more training epochs to make slow changes on weights at each iteration, whereas larger α results in rapid changes with fewer training epochs. People often not to choose a large α because a large α may end up overshooting the minimum.


Too low
Just right
Too high
J(θ)
θ
A small learning rate requires many updates before reaching the minimum point
The optimal learning rate swiftly reaches the minimum point
Too large of a learning rate causes drastic updates which lead to divergent behaviors

source: https://www.kaggle.com/residentmario/tuning-your-learning-rate

e)

i. $$\begin{aligned} \frac{\partial E}{\partial w_{K,j,k}} &= (O_k - T_k)\frac{\partial f\left(\sum_{j=0}^{H_K} w_{j,K,k}\cdot h_{j,K}\right)}{\partial w_{K,j,k}} \\ &= (O_k - T_k) f\left(\textstyle\sum_{j=0}^{H_K} w_{j,K,k}\cdot h_{j,K}\right)\left[1 - f\left(\textstyle\sum_{j=0}^{H_K} w_{j,K,k}\cdot h_{j,K}\right)\right]\frac{\partial\left(\sum_{j=0}^{H_K} w_{j,K,k}\cdot h_{j,K}\right)}{\partial w_{K,j,k}} \\ &= (O_k - T_k) f\left(\textstyle\sum_{j=0}^{H_K} w_{j,K,k}\cdot h_{j,K}\right)\left[1 - f\left(\textstyle\sum_{j=0}^{H_K} w_{j,K,k}\cdot h_{j,K}\right)\right] h_{j,K} \end{aligned}$$

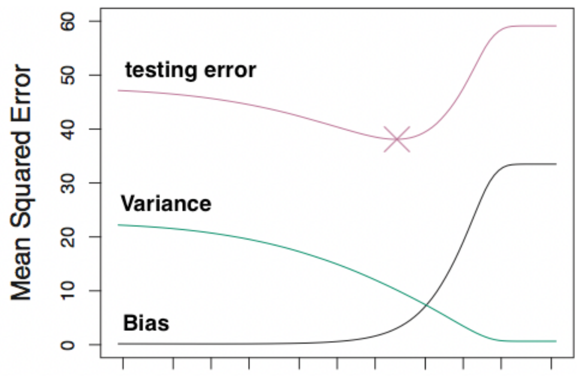
ii. $$\begin{aligned} \frac{\partial E}{\partial h_{i+1,k}} &= \sum_{m=1}^{H_{i+2}} \frac{\partial E}{\partial h_{i+2,\widehat{m}}}\cdot\frac{\partial h_{i+2,\widehat{m}}}{\partial h_{i+1,\widehat{m}}}\cdot\frac{\partial h_{i+1,\widehat{m}}}{\partial h_{i+1,k}} \\ &= \sum_{m=1}^{H_{i+2}} \frac{\partial E}{\partial h_{i+2,\widehat{m}}}\cdot\frac{\partial f\left(h_{i+1,\widehat{m}}\right)}{\partial h_{i+1,\widehat{m}}}\cdot\frac{\partial\left(\sum_{j=0}^{H_K} w_{i+1,k,\widehat{m}}\cdot h_{i+1,k}\right)}{\partial h_{i+1,k}} \\ &= \sum_{m=1}^{H_{i+2}} \frac{\partial E}{\partial h_{i+2,\widehat{m}}}\cdot h_{i+2,\widehat{m}}\left(1-h_{i+2,\widehat{m}}\right)\cdot w_{i+1,k,\widehat{m}} \\ &= \sum_{m=1}^{H_{i+2}} \Delta_{i+2,\widehat{m}}\cdot w_{i+1,k,\widehat{m}} \end{aligned}$$

iii. $$\frac{\partial E}{\partial w_{i,j,k}} = \frac{\partial E}{\partial h_{i+1,k}} \cdot \frac{\partial h_{i+1,k}}{\partial w_{i,j,k}}$$
$$= \sum_{m=1}^{H_{i+2}} \Delta_{i+2,\hat{m}} \cdot w_{i+1,k,\hat{m}} \cdot \frac{\partial f\left(\sum_{j=0}^{H_i} w_{i,j,k} \cdot h_{i,j}\right)}{\partial\left(\sum_{j=0}^{H_i} w_{i,j,k} \cdot h_{i,j}\right)} \cdot \frac{\partial\left(\sum_{j=0}^{H_i} w_{i,j,k} \cdot h_{i,j}\right)}{\partial w_{i,j,k}}$$
$$= \sum_{m=1}^{H_{i+2}} \Delta_{i+2,\hat{m}} \cdot w_{i+1,k,\hat{m}} \cdot f\left(\sum_{j=0}^{H_i} w_{i,j,k} \cdot h_{i,j}\right)\left[1 - f\left(\sum_{j=0}^{H_i} w_{i,j,k} \cdot h_{i,j}\right)\right] h_{i,j}$$

iv. $$w_{i,j,k} \leftarrow w_{i,j,k} + \alpha \cdot I_j \cdot \Delta_{i,k}$$
$$\leftarrow w_{i,j,k} - \alpha \cdot \frac{\partial E}{\partial w_{i,j,k}}$$

f) With a large number of layers in NN, the algorithm becomes more complex to calculate and update weights. Time cost would become a consider with an inordinately large number of layers. Moreover, a large number of layers may lead to overfitting. The NN has a huge information processing capacity to handle training data. When there is not enough training data for the NN, the NN will over learn (or process) from the training data and lead to overfitting. Besides, gradient descent may fail to update weights. After taking several partial derivatives, the gradient may vanish and will result to stop learning (or update) for weights.

g) In statistic, overfit a model would not affect the coefficient but also remain unbiased estimate. However, overfitting would lead to a model no longer adapt to a new environment. In other words, an overfit model may only favour to the training data. When there are unseen data, the predicted outcome may differ from the true outcome. More simply, it would form a lookup table instead of learning the pattern. Bias-variance trade-off is a possible solution.



source: Lecture from STAT4001

Given the fact that $test\ error \sim bias^2 + variance + irreducible\ error$, we can control the variance to reduce the amount of bias. At the point where variance equals bias, the test error is minimised, such that the issues of overfitting can be avoided.