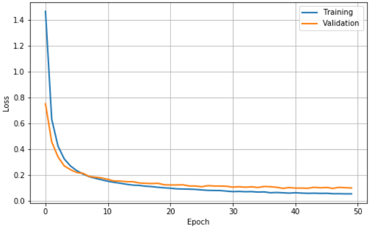
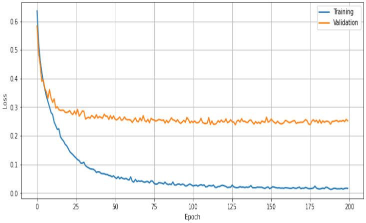
Outline

Introduction

- Training: learning model parameters.
- Validation: evaluate model on unseen data.
- Expect similar performance in training and validation data (both must come from the same underlying distribution).
- Chance hyper-parameters in the presence of performance gap.





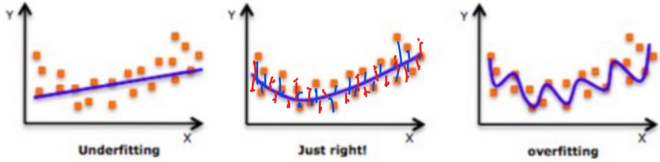
Overfitting

"Model fits training data extremely good, but fails to generalize for unseen data".

Common solution: give up some performance level on the training set in favor of improvement in the validation set.

High bias

High variance

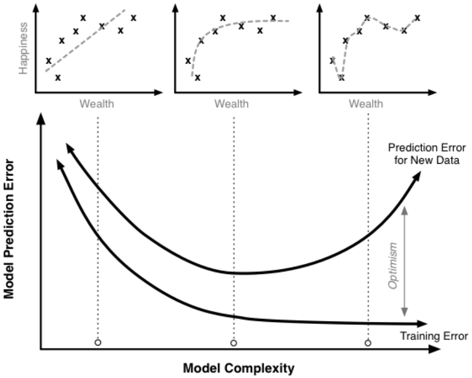


Bias and variance

- **Bias:** Inability of a model to learn the true mapping relationship between x and y, which implies underfitting, e.g., straight line.
- **Variance:** Difference in fits between training and validation sets, which might imply overfitting, e.g., high-order polynomial.

Bias-variance trade-off

Ideal model: Low bias (accurate mapping function) and low variance (performance is consistent between training and validation sets).



Common practices

To reduce underfitting and overfitting:

underffiting

- Add capacity.
- Add epochs.
- Add data.
- Add features.
- Data augmentation.

overffiting

- Early stopping.
- Gather more data.
- Decrease features.
- Add regularizers.
- Data augmentation.

Outline

Overfitting

Regularizers

Introduction

Limit the model capacity, by adding a penalty to the parameters.

$$\mathcal{L}_T(x, y; \Omega) = \mathcal{L}(x, y; \Omega) + \alpha \mathcal{P}(\Omega),$$

where,

- $\mathcal{L}(x, y; \Omega)$ corresponds to the target loss already known,
- $\mathcal{P}(\Omega)$ is the penalty function on the parameters,
- α weights the penalty of the parameters, and
- $\mathcal{L}_T(x, y; \Omega)$ indicates the total loss.

Weight decay $L2$

$y = w_1 x_1 + w_2 x_2 + \underline{w_3 x_1 x_2}$

a.k.a., Ridge regression or Tikhonov regularization.

$$\mathcal{P}(\Omega) = \frac{1}{2}\|\Omega\|_2^2.$$

$\min E(y, \hat{y}) + \frac{1}{2}\|\Omega\|_2^2$

$\frac{\partial E}{\partial w_i} =$

Derivative:

$$\Omega.$$

Lasso $L1$

$$\mathcal{P}(\Omega) = \|\Omega\|_1 = \sum_i |\omega_i|.$$

Derivative:

$$\text{sign}(\Omega).$$

Besides keeping small values for ω_i, it also induces sparsity.

Elastic $L2L1$

$\min \; E(y, \hat{y}) + \alpha \mathcal{P}(\Omega)$

Ridge + Lasso.

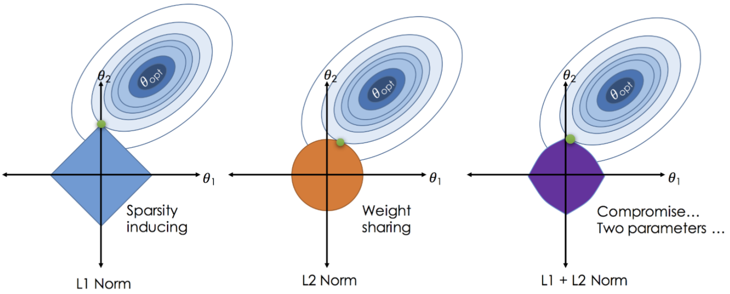
$$\mathcal{P}(\Omega) = \alpha_R \|\Omega\|_2^2 + \alpha_L \|\Omega\|_1.$$

Derivative:

$$\Omega + \mathsf{sign}(\Omega).$$

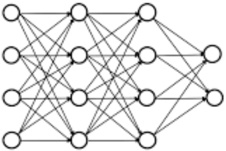
Regularized space

Keep parameter values at the intersection between the Loss space and the Penalty space.



Dropout

- Add stochasticity.
- During training, limit the capacity of the model at random.
- Deactivate neurons (dropout) or weights (dropconnect).
- Forces the network to become redundant.
- Also helps make the model robust against variations.



(a) Standard Neural Network. (b) Network applying dropout

Srivastava et al., 2014. “Dropout: A Simple Way toPrevent Neural Networks from Overfittin”.

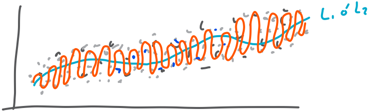
Batch normalization, I

a.k.a., batchnorm.

- Normalization is often applied to input layer (accelerates learning).
- We can do the same to hidden layers.
- Adds noise and learns to be robust against it.
- Induces independence between layers.

Ioffe & Szegedy, 2015. ICML. "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift".

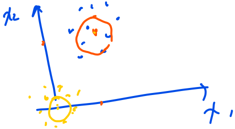
Batch normalization, II



Compute the batch mean and variance:

$$\mu_B = \frac{1}{M}\sum_{m=1}^{M} x^{(m)}, \qquad \sigma_B^2 = \frac{1}{M}\sum_{m=1}^{M}\left(x^{(m)} - \mu_B\right)^2 .$$

Normalize and feed to next layer:



$$\hat{x}^{(m)} = \gamma \cdot \frac{x^{(m)} - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}} + \beta.$$