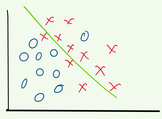
Week5
Tuesday, May 12, 2020 10:59 PM

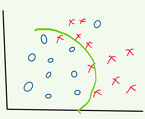
Hyperparameter tuning & optimization

- Data division → 60% (Train) 20% (dev) 20% (Test)

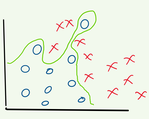
Bias, Variance trade-off



underfitting
bias ↑



just right



overfitting
variance ↑

Error				
Train	15%	0.5%	1%	15%
Test	16%	1%	16%	30%
		bias ↓ variance ↓		bias ↑ variance ↑

High bias → Train longer (not larger data) } help it learn
→ bigger n/w

High variance → Supply more data } help it generalize
Regularization

Frobenius Norm (L2 regularization)

In general,

$$J(w^{[1]}b^{[1]} \dots w^{[l]}b^{[l]}) = \frac{1}{m}\sum_{i=1}^{m} \mathcal{L}(\hat{y}^{(i)}, y^{(i)})$$

(training example)

Now

frobenius form (L2 regularization)

$$J(w^{[1]}b^{[1]} \dots w^{[l]}b^{[l]}) = \frac{1}{m}\sum_{j=1}^{m} \mathcal{L}(\hat{y}^{(i)}, y^{(i)}) + \frac{\lambda}{2m}\sum_{l=1}^{L} \|w_l\|^2$$

Effect in back propagation!-

$$dw^{[l]} = (\text{from b/p}) + \frac{\lambda}{m} * \|w_l\|$$

$\therefore$ in gradient descent

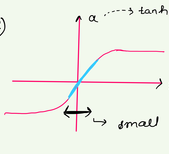
$$dw = dw - \alpha * \left(\text{from b/p} + \frac{\lambda}{m} * \|w_l\|\right)$$

↳ that's why L2 is called as weight decay algorithm.

Why regularization reduces overfitting?

1) L2: it penalizes the weights in G.D
↳ eliminates few nodes
↳ makes the neural n/w simpler
↳ reduces learning
↳ forces toward higher bias
↳ good value of α → optimizes the tradeoff.

2)



↳ small value of z
↳ a is linear (blue line)

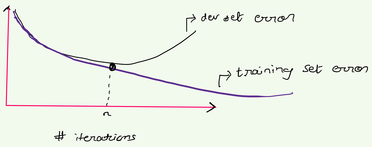
$z = w^T x + b$
↳ smaller after L2
↳ z smaller
↳ predicted a will be from linear highlighted range
↳ decision boundaries will be linear
↳ overfitting will be reduced

} important

few bias
high variance

low bias
low variance.

(*) Early Stopping:-



stop training at 'n'.

* Disadvantages:-

→ it mixes gradient descent with Regularization } orthogonalization.
↳ * Both are separate tasks. should never be mixed.