Machine Learning from Data HW7

Shane O'Brien

October 2017

Exercise 4.3

a

Deterministic noise will go up. This is because as the complexity of f goes up, the ability of H to match f gets worse. There is also a lower tendency to overfit. This is because our H stays the same. Overfitting happens when our H is so large and complex that it matches the noise from f.

b

Deterministic noise will go up. This is because our H will increasingly not be able to match f as the complexity of H goes down. The tendency to overfit goes down as well. This is because we have a less complex H. If our H is really complex, we have a chance of outputting a complex g that matches noise. If our H simplifies, this happens less often.

Exercise 4.5

a

We essentially want Γ to disappear, so we set Γ equal to the identity matrix I

b

If $\mathbf{w}$ is a d-dimensional vector, we set Γ as a d-dimensional vector of 1's.

Exercise 4.6

The hard-order constraint will be more useful for binary classification. This is because of the nature of binary classification. It doesn't matter how far or close a data point is to our classification line.

In the case of regression, distance matters. A soft-order constraint could use this so pick a complex classifier that minimizes distances. But for binary classification, we could use a simpler H.

Exercise 4.7

a

We will use two things for this problem:

$$E_{val}(g^-) = \frac{1}{K} \sum_{\mathbf{x}_n \in D_{val}} e(g^-(\mathbf{x}_n), y_n)$$

$$\sigma^2(g^-) = Var_x[e(g^-(\mathbf{x}), y)]$$

So for our problem:

$$\sigma^2_{val} = Var_{D_{val}}[E_{val}(g^-)]$$

$$= Var_{D_{val}}[\frac{1}{K} \sum_{\mathbf{x}_n \in D_{val}} e(g^-(\mathbf{x}_n), y_n)]$$

$$= \frac{1}{K} Var_{D_{val}}[\sum_{\mathbf{x}_n \in D_{val}} e(g^-(\mathbf{x}_n), y_n)]$$

$$= \frac{1}{K} Var_x[e(g^-(\mathbf{x}_n), y_n)]$$

$$= \frac{1}{K} \sigma^2(g^-)$$

b

We need to establish a few things before starting this problem. Due to the nature of binary classification (we can only get 0 or 1 for our error), $E[e^2] = E[e]$. We also know that the probability that g^- classifies a point incorrectly is $P[g^-(x) \neq y]$.

$$\sigma^2_{val} = \frac{1}{K} Var_x[e(g^-(\mathbf{x}_n), y_n)]$$

$$= \frac{1}{K}(E[e^2] - E[e]^2)$$

$$= \frac{1}{K}(P[g^-(x) \neq y] - P[g^-(x) \neq y]^2)$$

c

Our previous problem boils down to $\frac{1}{K}(P - P^2)$. So if we want to analyze worst case, we need to know when $P - P^2$ gives the largest result. By intuition, we know that $P = 0.5$ is this number.

We can put this into our previous result:

$$= \frac{1}{K}(P[g^-(x) \neq y] - P[g^-(x) \neq y]^2)$$

$$= \frac{1}{K}(0.5 - (0.5)^2)$$

$$= \frac{1}{K}(0.25)$$

$$= \frac{1}{4K}$$

Since this is the worst case, we have our final answer:

$$\sigma_{val}^2 \leq \frac{1}{4K}$$

d

No, there is no uniform upper bound for variance. This is because squared error in general has no upper bound, it could go infinitely high.

e

I expect $\sigma^2(g^-)$ to be higher. If you train on fewer points, then the possibilities for g^- are generally less accurate. A higher squared error average leads to highers variance.

f

As always, there is a trade-off. If you adjust the size of your validation set, two things could happen:

1. You decrease the amount of data you train off of, leading to a worse g_-
2. You decrease the amount of data you test off of, leading a a worse connection from E_{val} to E_{out}

There is a sweet spot, which can be hard to find. Increasing the size of the validation set leads to the first scenario.

Exercise 4.8

Yes, E_m is an unbiased estimate for the out-of-sample error $E_{out}(g_m^-)$. That is there is no choice yet. It is just am estimate based off of a model. Once we introduce choice (like for selecting E_m^*), then we have a biased estimate E_m^*.