* Bias Variance decomposition

We try to decompose the generalized error and understand what leads to the error. What terms exactly contribute to that error.

A parameter is some quantity about a distribution we are interested in. We estimate this parameter θ using an estimator $\hat{\theta}$.

(1) Bias $(\hat{\theta}, \theta) = E(\hat{\theta}) - \theta$

What the bias measures is how off is the estimator from being a true estimator. This is the error we will still get, even if we train with infinite data. This is due to classifier being biased to a particular type of solution.

For eg. Linear classifiers have a bias that the features and labels are linearly related.

In other words, bias is inherent to the Model & it does not depend on data as such.

(2) Variance $(\hat{\theta}) = E(\hat{\theta} - E(\hat{\theta}))^2$

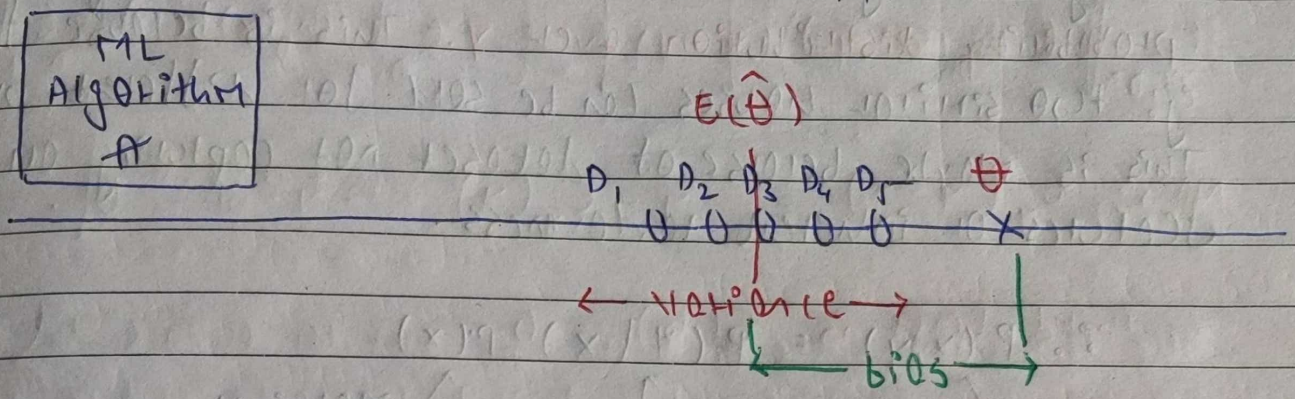
What variance captures is how much do the classifiers vary in general. Thus it measures how different our classifiers become if we change the training data.

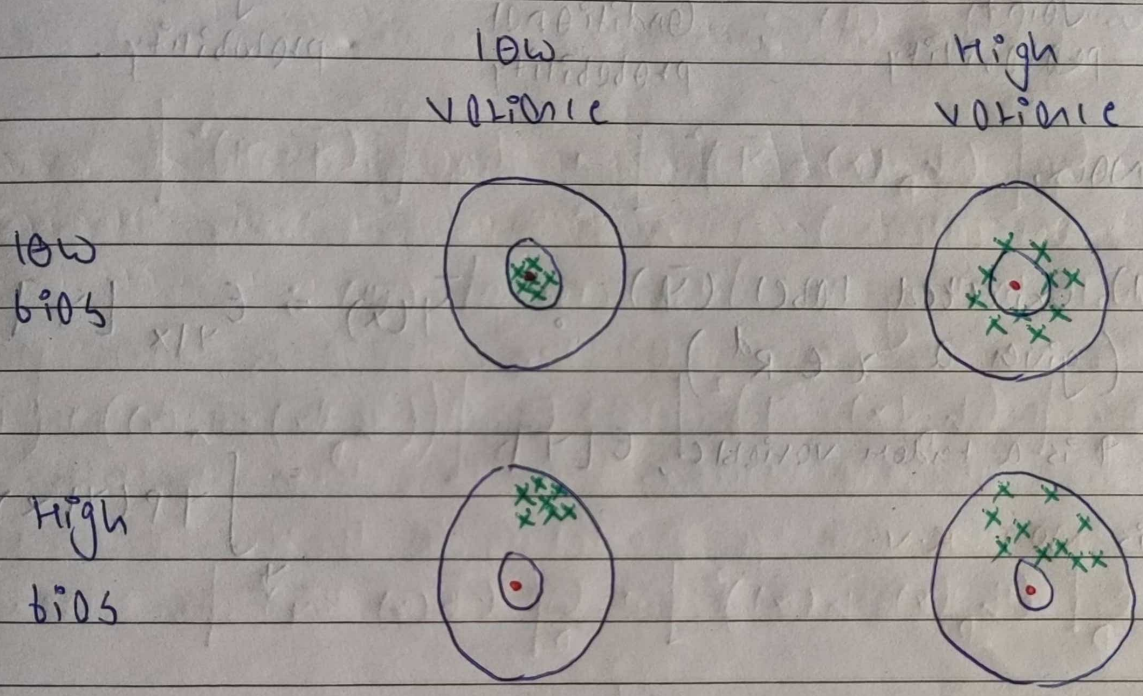
For eg. Non linear classifiers get affected even by slight change of data. Thus, they depict high variance.

Variance is independent of true prediction. It just tells how much I will vary.

Thus;

D_i are different training data subsets drawn from $P(X, Y)$.





∴ Variance measures spread while bias measures accuracy.

This Bias-Variance tradeoff is what we need to consider while choosing a Model. This is what tells us that Non-linear models are NOT always better than linear Models. It is the tradeoff over which a Model is selected.

Consider a distribution $P(x,y)$ where $y \in \mathbb{R}$; i.e. Regression setting. For a given x; we do NOT have a single y; but probability distribution over y. This is because for eg. two similar houses can be sold for different price. This is maybe because of dataset not capturing all features or inherent randomness.

$$\therefore P(x,y) = P(y|x)\,P(x)$$

- $P(x,y)$: Joint probability
- $P(y|x)$: Conditional probability
- $P(x)$: Marginal probability.

Now,

(1) Expected label ($\bar{y}$) (given a $x \in \mathbb{R}^d$): $\bar{y}(x) = \mathbb{E}_{y|x}[y]$

$$= \int_y y\,P(y|x)\,dy$$

Note: If y is a random variable, $\mathbb{E}[y]$ or $\bar{y}$ is NOT R.V.

(2) Expected Test error (given a classifier h_d):

$$\mathbb{E}_{x,y \sim P}\left[(h_D(x) - y)^2\right]$$

$x, y \sim P$: test points

$(h_D(x) - y)^2$: squared loss

classifier obtained from algorithm A & dataset D

$h_D = A(D)$

$$= \int_x \int_y (h_D(x) - y)^2\, y\, P(y|x)\,dy\,dx$$

for all x; y to find the most possible y or expected y & then find test error over all x.

(3) Expected classifier ($\bar{h}(x)$) : $\mathbb{E}_{D\sim\hat{P}}[\mathcal{A}(D)]$
(given Algorithm $\mathcal{A}$)

$$= \int_D h_D P(D)\, dD$$

This is nothing but $\mathbb{E}(\hat{\theta})$.

(4) Expected generalization error of Algorithm $\mathcal{A}$: $\mathbb{E}_{\substack{x,y\sim P \\ D\sim\hat{P}}}\left[(h_D(x)-y)^2\right]$

$$= \int_D \int_y \int_x \left[(h_D(x)-y)^2\right] \times y\, P(y|x) \times P(D)\, dx\, dy\, dD$$

This is nothing but expected square loss error of the Algorithm $\mathcal{A}$. Hence, this is what we want to decompose & understand.

Consider the error at a fixed x, i.e. (we will introduce $\mathbb{E}_x$ in the end)

$$= \mathbb{E}_{y,D}\left[(h_D(x)-y)^2\right]$$

Note : $\bar{y}$ is NOT a random variable

$$= \mathbb{E}_{y,D}\left[\left((h_D(x)-\bar{y})+(\bar{y}-y)\right)^2\right]$$

$$= \mathbb{E}_{\not{y},D}\left[(h_D(x)-\bar{y})^2\right] + \mathbb{E}_{y}\left[(\bar{y}-y)^2\right]$$

This term does not depend on D

This term does not depend on y

$$+ 2\,\mathbb{E}_{y,D}\left[(h_D(x)-\bar{y})(\bar{y}-y)\right]$$

—(I)

However; if we look at third term;

$$E_{Y,D}\left[(h_D(x)-\bar{y})(\bar{y}-y)\right] = E_D\left[(h_D(x)-\bar{y})\right] \times E_Y\left[(\bar{y}-y)\right]$$

Now; $E_Y(\bar{y}-y) = \bar{y} - E_Y(y) = \bar{y} - \bar{y} = 0$.

∴ Third term is $E(0) = 0$. —— (II)

Using (I) & (II);

$$E_D\left[(h_D(x)-\bar{y})^2\right] + E_Y\left[(\bar{y}-y)^2\right] \quad \text{—— (III)}$$

Just looking at first term of (III);

$$E_D\left[(h_D(x)-\bar{y})^2\right] = E_D\left[\left((h_D(x)-\bar{h}(x)) + (\bar{h}(x)-\bar{y})\right)^2\right]$$

$$= E_D\left[(h_D(x)-\bar{h}(x))^2\right] + E_D\left[(\bar{h}(x)-\bar{y})^2\right]$$

$$+ 2\,E_D\left[(h_D(x)-\bar{h}(x))(\bar{h}(x)-\bar{y})\right]$$

This is 0, similar to above

$$= E_D\left[(h_D(x)-\bar{h}(x))^2\right] + E_D\left[(\bar{h}(x)-\bar{y})^2\right]$$

—— (IV)

Resubstituting (IV) in (III);

Note: if x is R.V, E(x) is NOT R.V. Thus; $E_D[(\bar{h}(x) - \bar{y})^2]$ is $(\bar{h}(x) - \bar{y})$.

KENY Platinaa
Page No.:
Date

$$= E_D\left[(h_D(x) - \bar{h}(x))^2\right] + E_D\left[(\bar{h}(x) - \bar{y})^2\right] + E_y\left[(\bar{y} - y)^2\right]$$

$$= E_D\left[(h_D(x) - \bar{h}(x))^2\right] + (\bar{h}(x) - \bar{y})^2 + E_y\left[(\bar{y} - y)^2\right]$$

Variance + bias^2 + Noise

This term just tells us how varied our classifiers are from the mean classifier. It does NOT depend on y.	This term tells us how off our classifier is from the true prediction and is squared. It is not dependent on D or y	This term tells us how off our true prediction is from the true value. Even bayes optimal classifier suffers from noise.

Now;

$$E_{x,y,D}\left[(h_D(x) - y)^2\right] = E_{x,D}\left[(h_D(x) - \bar{h}(x))^2\right] + E_x\left[(\bar{h}(x) - \bar{y})^2\right] + E_{x,y}\left[(\bar{y} - y)^2\right]$$

Variance + Bias^2 + Noise

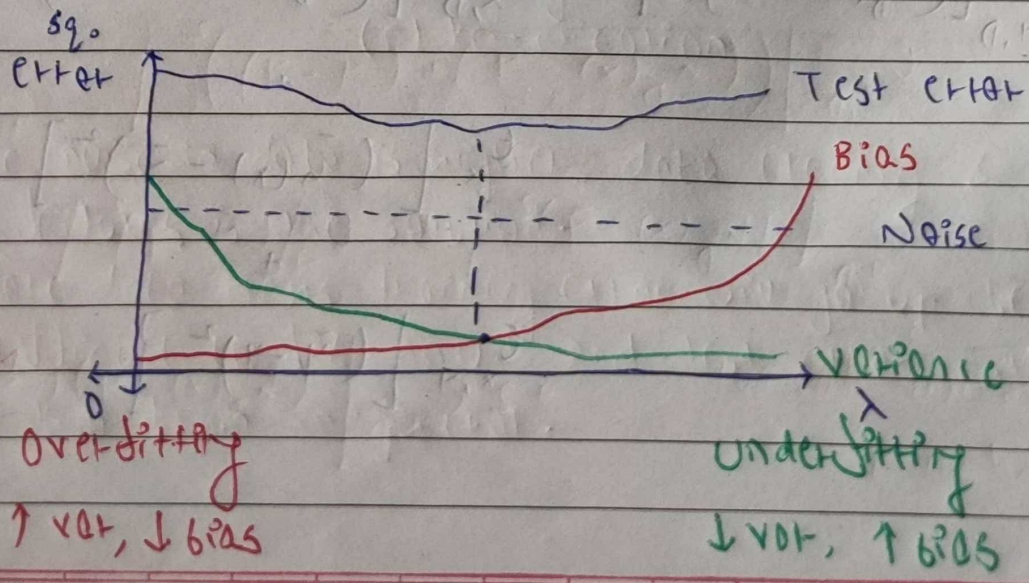
In other words:

$$E_{x,y,D}\left[(h_D(x)-y)^2\right] = E_x\left[\text{var}_{y|x}(y) + \text{var}_D(h_D(x)) + \text{bias}(h_D(x))^2\right]$$

The trade off here is that we want to choose h_D to balance between second & third term to reduce MSE. We can NOT just minimize one or other, but we can find a balance between these terms.
For eg. At times, increasing bias by little (for eg. regularization) allows us to substantially drop variance.

The variance term controls the wiggliness, hence might prefer a simple function that can't yield predictions which are too varied. The bias term controls how close the average model is to truth. We need to pay attention to data to reduce the bias term.

Introducing structures like regularization allows us to tune bias-variance tradeoff.

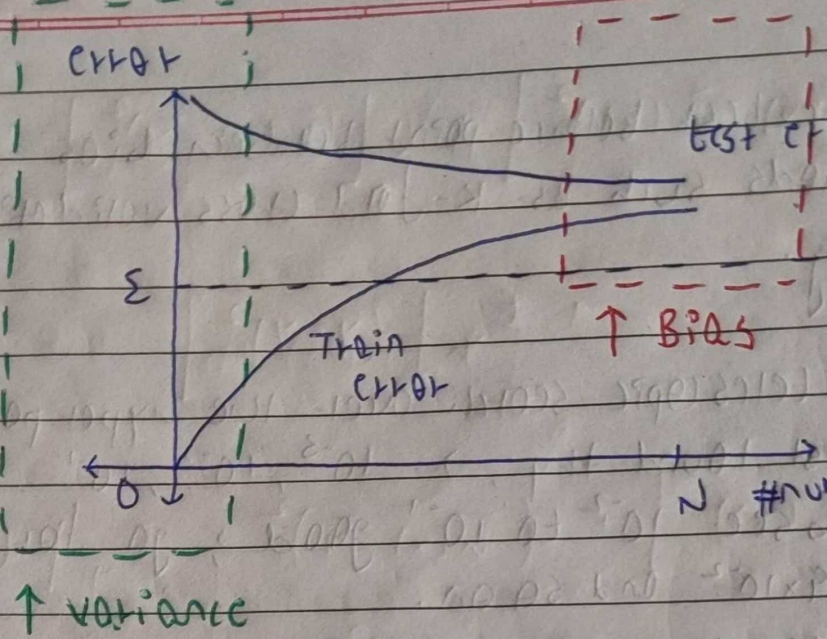


To choose the best balance point between bias & variance, we use Methods such as K-fold cross validation or early stopping.

You perform telescopic search over the hyper parameters for eg. on first round try, $\lambda = 10^{-3}, 10^{-2}, 10^{-1} \dots 10^{3}$. Then, if you see 10^{-2} to 10^{-1} good; go for 1×10^{-2}, 2×10^{-2} — 9×10^{-2} and so on.
But if tune too much on validation set, we will overfit on validation set. This is because if you tune too hard to get accuracy from 80 to 90%; that would mean you tuned too hard to get 4-5 examples right, cause validation set is small. Thats why we do K-fold cross validation.
Here; Training data is split in K buckets and for K times you can take K-1 buckets to train & 1 bucket to validate. The larger K is better, but of course it takes more time. So if K=N; that is leave out one cross validation.

Now when we get high error, we can estimate where the loss exactly is. If the variance of model is low, reducing variance won't help.

One idea for ML debugging is to plot error vs num instances. Here, we start from 1 instance in training data to all N instances. We train the model every time & plot train error & test error. ε is the error limit we want by the model.



increasing training data here won't help at all but only converge the line

If (test error is $< \varepsilon$) ; No problem
else if (test error $> \varepsilon$) & (train error $< \varepsilon$) ;
This is ↑ variance. OR Overfitting
Solutions : (1) Add more training data
(2) Bagging
(3) Reduce model complexity
(4) ↓ regularization
else (test error $> \varepsilon$) & (train error $> \varepsilon$) ;
This is ↑ Bias OR Underfitting
Solutions : (1) use more complex model (non-linear Maybe
(2) Add more features
(3) Boosting
(4) ↑ regularization
(5) kernelization

If the noise in data is higher, data cleaning & adding more features is the solution.
Also, more data means training error will go up because its more difficult to fit now. And test error will be greater than train in general.