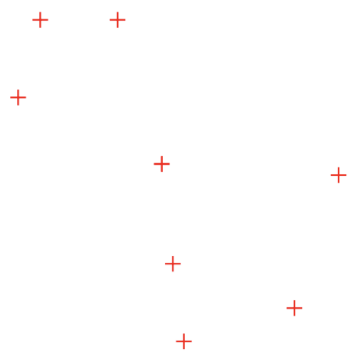
# Bias and Variance

Badrinath Singhal

March 2021

Lately I am reading about some concepts in deep learning through NPTEL from instructor Prof. Mitesh M Khapra. His lectures are very technical and makes the concepts very clear mathematically which gives me the confidence about exploring issues in deep learning. I am making notes along with the lectures and thus I am posting some of the notes here. I have taken some diagrams directly from Prof. Khapra's lecture. Although his lectures are more than enough for anyone to understand the concepts clearly I am posting my notes here so I can revist them quickly and in-case if anyone stumbles upon my page they can find this document useful.

Let us consider problem of fitting a curve given set of points. Below is the image which represents the set of points.



The points were drawn from a sinusoidal function (the true $f(x)$)

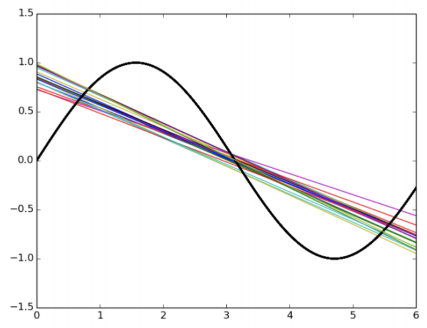
Now we will consider two models to fit the curve on given set of points. First model will be a simple model and second will be a complex model.

$$Simple : y = \hat{f(x)} = w_1x + w_0$$

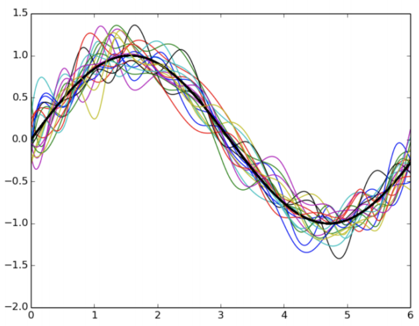
$$Complex : y = \hat{f(x)} = \sum_{i=1}^{25} w_ix^i + w_0$$

We know that complex model can fit the given data much better than the simple one and we are assuming while training that we have no information about the underlying function where the points are drawn from. We have 100 training points, we sample 25 points at a given time and train our model with those 25 points. We repeat this process multiple times and plot our model.

Below is the plot for simple model



We see that multiple simple models trained are very close to each other but doesn't fit the data quite well to our original points. Below is the plot for complex model



In this plot we observe that models vary more but they fit quite well on the training data.

Let $f(x)$ be true model and $f(\hat{x})$ be estimated model, we define bias as below

$$Bias(f(\hat{x})) = E[f(\hat{x})] - f(x)$$

$E[f(\hat{x})]$ is expected value of $f(\hat{x})$

Given our observation above, we can say that simple model have high bias and complex model have low bias.

$$Variance(f(\hat{x})) = E[(f(\hat{x}) - E[f(\hat{x})])^2]$$

Based on above plot we can say that simple model have low variance whereas complex model have high variance. In summary:

**Simple Model:** low variance, high bias **Complex Model:** high variance, low bias

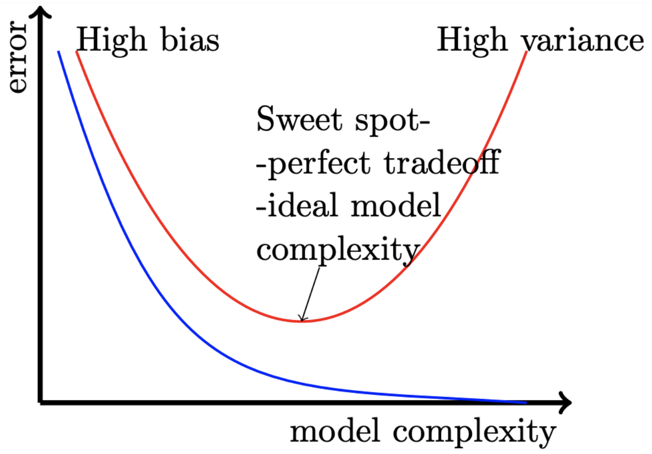
There is always a tradeoff between bias and variance and both of them contribute to mean square error.

## 1 Train Error vs Test Error

Consider new point $(x, y)$ which was not seen during training, if we use $f(\hat{x})$ to predict then mean squared error would be $E[(y - f(\hat{x}))^2]$. We can show that

$$E[(y - f(\hat{x}))^2] = Bias^2 + Variance + \sigma^2(irreducibleerror)[2]$$

There are two errors, train error and test error. Train error is the error on training data whereas test error is error on test data using the model trained.



Let there be $m$ test data $(x_i, y_i)$ and for any point we assume some noise on test data i.e.

$$y_i = f(x_i) + \epsilon_i$$

$\epsilon_i$ is noise and we assume it to come from normal distribution, and we don't know $f(x_i)$ because it's from test data. Using our train we are interested in calculating

$$E[(f(\hat{x_i}) - f(x_i))^2]$$

We cannot estimate it directly because we don't know $f$ so we will estimate it empirically.

$$E[(\hat{y_i} - y_i)^2] = E[(f(\hat{x_i}) - f(x_i) - \epsilon_i)^2]$$

$$E[(\hat{y_i} - y_i)^2] = E[(f(\hat{x_i}) - f(x_i))^] - 2\epsilon_i(f(\hat{x_i}) - f(x_i)) + \epsilon_i^2]$$

$$E[(f(\hat{x_i}) - f(x_i))^2] = E[(\hat{y_i} - y_i)^2] - E[\epsilon_i^2] + 2E[\epsilon_i(f(\hat{x_i}) - f(x_i))]$$

Now here, by empirically estimating we mean, we will have certain number of test data (in this case $m$) and model's prediction on each test data. So we will have certain numbers of $\hat{y_i}$ and $y_i$. We can replace $E[(\hat{y_i} - y_i)^2]$ with

$$E[(\hat{y_i} - y_i)^2] = \frac{1}{m}\sum_{i=1}^{m}(\hat{y_i} - y_i)^2$$

Also we say $E[(f(\hat{x_i}) - f(x_i))^2]$ as our true error, so

$$trueerror = \frac{1}{m}\sum_{i=1}^{m}(\hat{y_i} - y_i)^2 - \frac{1}{m}\sum_{i=1}^{m}\epsilon_i^2 + 2E[\epsilon_i(f(\hat{x_i}) - f(x))]$$

$2E[\epsilon_i(f(\hat{x_i}) - f(x))]$ is covariance since mean of noise is 0 so you can solve the equation $E[(X - \mu_x)(Y - \mu_y)]$ where $X$ is noise distribution and $Y$ is $f(\hat{x_i}) - f(x_i)$ and $\mu_x = 0$ and $E[X] = 0$

We call $\frac{1}{m}\sum_{i=1}^{m}(\hat{y_i} - y_i)^2$ as empirical estimation error.
Therefore,

$$trueerror = empiricalestimationerror - smallconstant + covariance$$

For test data, $\epsilon_i$ is independent of $f(\hat{x_i}) - f(x_i)$ since none of the test data was used to estimate $f(\hat{x_i})$

$$E[\epsilon_i(f(\hat{x_i}) - f(x_i)] = E[\epsilon_i]E[f(\hat{x_i}) - f(x_i)] = 0$$

$$trueerror = empiricalestimatederror + smallconstant$$

We can see how the true error is very closely related to empirical estimated error, thus while training model we should always set aside validation set so we can calculate the empirical error to get the sense of where the true error lies.
In training set $\epsilon_i$ is not independent of $f(\hat{x_i}) - f(x_i)$ and thus doesn't give true picture of the error.
We can relate this to model complexity using Stein's Lemma [3] and say that complex model will be more change in estimation of $f(\hat{x_i})$. So we can further say that

$$trueerror = empiricaltrainerror + smallconstant + \Omega(modelcomplexity)$$

Hence, while training instead of minimizing the training error we should minimize with additional factor related to model complexity. That factor will be high for complex models and small for simple models. This is the basis of all regularisation.

## References

[1] Lecture 8 of DeepLearning CS7015 by Prof. Mitesh M Khapra, NPTEL

[2] The Bias Variance Tradeoff

[3] Stein's Lemma