Lecture 2: Model Bias and Variance

Iain Styles

7 October 2019

Introduction

Noisy Data and Overfitting

As we saw at the end of the last lecture, the addition of noise to a dataset can have significant effects on the result of a regression analysis. This is particularly so when the model that we are using is very expressive and can model quite complicated patterns. In this lecture, we will study this in more detail, and this will lead us to both a set of practical methods for choosing a model, and an understanding of the theoretical limits of a learning algorithm.

We will develop these ideas in the context of a more complex example than a "simple" straight line fit. This will allow us to both explore the power that linear methods can have, and also to see more clearly how changing a model can affect the outcome of a regression analysis. The function we will study is a simple one in many ways, but is rather more complex in others:

$$y(x) = \sin(2\pi x) \tag{1}$$

One can of course perform a very simple linear fit on this by choosing $f(\mathbf{w}, x) = w_0 \sin(2\pi x)$, but the more general case $f(\mathbf{w}, x) = w_0 \sin(w_1 x)$ cannot be solved directly via linear methods. However, we can acheive a good approximate solution using a polynomial basis. That is, we seek to find coefficients $\mathbf{w}$ such that

$$\sin(2\pi x) = \sum_{i=0}^{M-1} w_i x^i. \tag{2}$$

Should we expect to be able to do this? Yes – it is a well known mathematical result that $\sin(ax) = ax - \frac{a^3x^3}{3!} + \frac{a^5x^5}{5!} - \frac{a^7x^7}{7!} + \cdots$ (a Maclaurin series), and the coefficients of this for $a = 2\pi$ are $\mathbf{w} \approx (0, 6.28, -41.34, 0, 81.61, 0, -76.7, 0, 42.1, \dots)$. We will perform two experiments. First, we will attempt to fit a polynomial of different degrees to $y = \sin(2\pi x)$. Then, we will investigate the effect of a little additive noise by fitting a polynomial to $y = \sin(2\pi x) + \epsilon$, where ϵ will be drawn from $\mathcal{N}(0, 0.25)$, a normal distribution with mean 0 and variance 0.25.

In Figure 1, we see the results of different orders M of fit to $N = 10$ data points sampled from $y = \sin(2\pi x)$ in the range $x \in [0, 5]$. It is common to measure the quality of the fit by plotting the root-mean-square error, $R = \sqrt{\frac{1}{N}\sum_i r_i^2}$, which is shown in Figure 2. Note that this is related to the least-square loss function by $R = \sqrt{\mathcal{L}_{\text{LSE}}/N}$ From this curve, we see that the error converges rapidly towards zero, with little change after $M = 7$ which visually looks to be a good model of the data, and the estimates of the values of y

for values of x that were not sampled in the original data are good (note how well matched the black (true) and red (estimated) curves match). Notice the plateaus in the RMS plot which reflect the even terms with a coefficient of zero in the Taylor series.

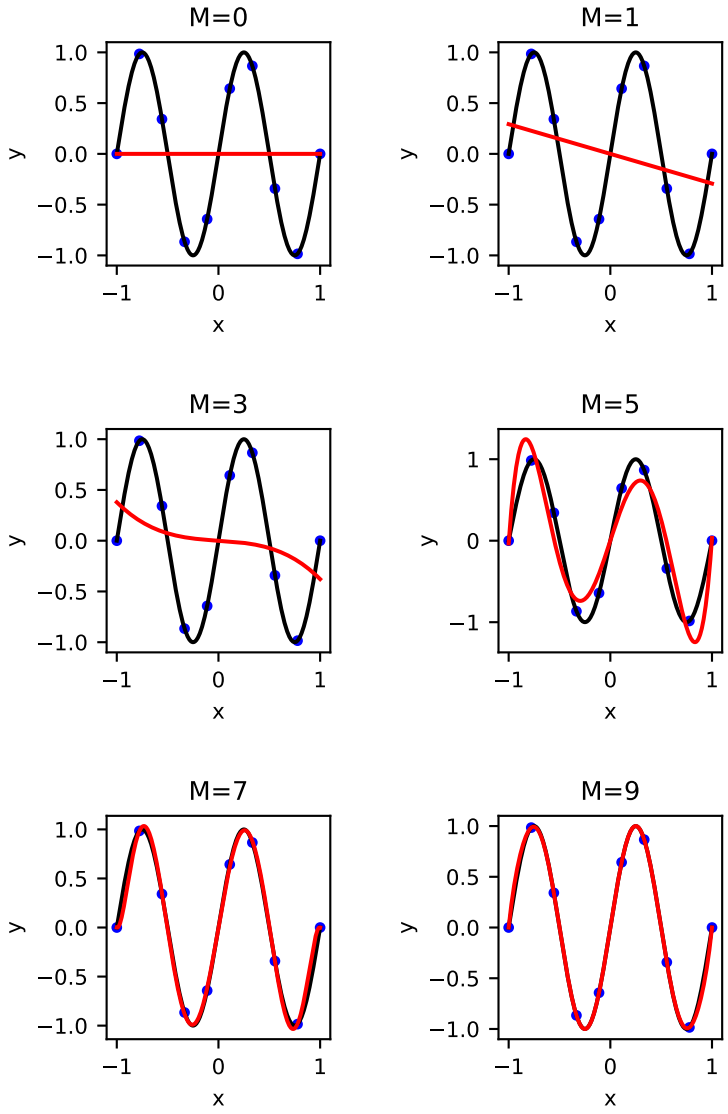

Figure 1: Fitting $y = \sin(2\pi x)$ with polynomial fits of increasing degree M. The blue line represents the true function; the black points are the sampled points (x_i, y_i); the red line is the best-fit polynomial of that order.

It is also instructive to study the coefficients that we obtain from the fit to make sure they are consistent with the Maclaurin series. These are shown in Table 1. We should not expect these to match our expectations exactly because we have only modelled the range $x \in [0, 5]$ and should have no expectation that our result will hold outside this. We notice that the correspondence between the terms and their true values gradually improves as we add high order terms in to the model until for $M = 9$, the difference is very small for the first few terms.

Let us now consider the effect of adding some noise to the sampled data. This is much more realistic in most situations where the data has been obtained through some measurement process.

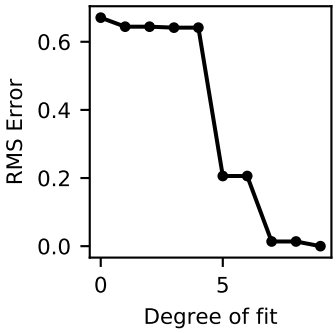

Figure 2: RMS Error of polynomial fits of different degree to $y = \sin(2\pi x)$.

M	w_0	w_1	w_2	w_3	w_4	w_5	w_6	w_7	w_8	w_9
0	0.00									
1	0.00	-0.29								
2	0.00	-0.29	-0.00							
3	0.00	-0.07	-0.00	-0.31						
4	0.00	-0.07	0.00	-0.31	-0.00					
5	0.00	3.85	-0.00	-16.51	0.00	12.69				
6	0.00	3.85	-0.00	-16.51	0.00	12.69	-0.00			
7	-0.00	6.00	0.00	-35.84	-0.00	54.04	0.00	-24.20		
8	-0.00	6.00	0.00	-35.84	-0.00	54.04	0.00	-24.20	-0.00	
9	-0.00	6.28	0.00	-41.12	-0.00	78.61	0.00	-63.77	-0.00	20.00
True	0	6.28	0	-41.34	0	81.61	0	-76.7	0	42.1

Table 1: Coefficients of polynomial fits of different degree to $y = \sin(2\pi x)$.

We generate ten uniformly spaced data points in $x \in [0, 5]$ using $y = \sin(2\pi x) + \epsilon$ where $\epsilon \sim \mathcal{N}(0, 0.25)$. In Figure 3 we show different degrees of polynomial fit to this data.

For low order fits, the fitted curves look very similar to the noise-free case and $M = 5$ gives a result that is beginning to resemble a sin curve. However, at $M = 9$ something odd happens. The fit appears to get much worse with large divergences from the ground truth. Yet the RMS error, shown in Figure 4, shows the error between the model and the data continuing to decrease. What is going on here?

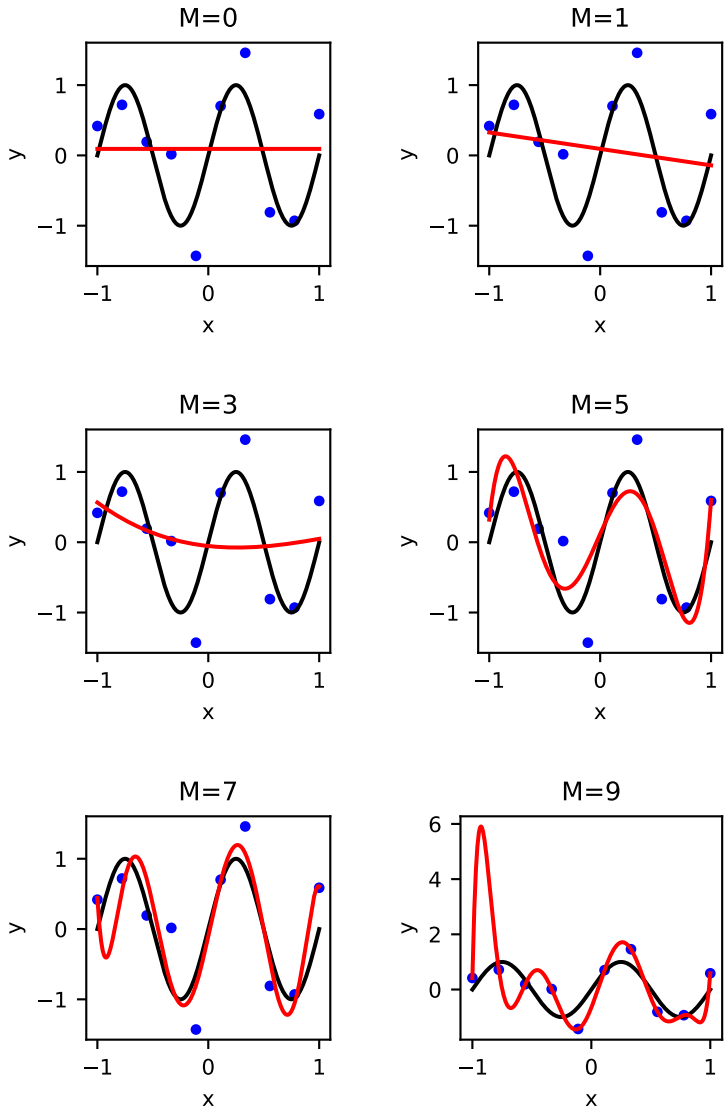

Figure 3: Fitting $y = \sin(2\pi x)$ with polynomial fits of increasing degree M to data with added noise. The blue line represents the true function; the black points are the sampled points (x_i, y_i); the red line is the best-fit polynomial of that order.

To understand this, we should think about what we are trying to achieve. We have N data points from which we are trying to derive the values of M free parameters. In the case $M < N$ we have fewer free parameters than we have data points, and the solution that minimises the least-squares loss function, but *does not necessarily pass through any of them*, as we see for small values of M. In other words, there are no parameters that we can choose for a straight-line that will exactly fit this data. The case $M = 9$, however, has ten unknowns, and ten free parameters. It is therefore possible (assuming the basis is suitably expressive) to choose these parameters so that the function passes exactly through all of the data points. In the noise-free case, we saw that this didn't matter – the data points all lie exactly on the curve $y = \sin(2\pi x)$, but when we add a little noise to the data, we introduce high-frequency fluctuations into the data that cannot be represented using low-order fits, but can be matched exactly by high-order fits. At $M = 9$ we have ten free parameters and so we can engineer these to construct a curve that exactly passes through all of the data points. For a more familiar examples, it is *always* possible to fit a straight line to two data points.

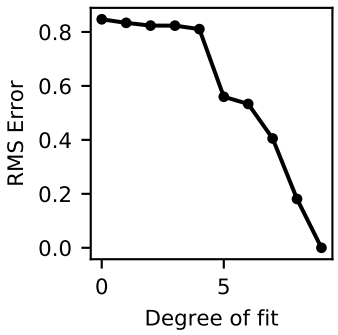

Figure 4: RMS Error of polynomial fits of different degree to $y = \sin(2\pi x)$.

The price of this "perfect fit" is that the curve deviates wildly from the ground truth between the sampled points. In order to exactly fit the high-frequency noise components, higher order terms tend to have large amplitudes (Table 2) so that the high order polynomial terms can generate the high-frequency fluctuations needed, but cancel each other out at the data points so that the fit is exact. This leads to the large deviations we see between the sampled points. This is know as *overfitting* and is a very common problem when working with noisy datasets and complex models. A model that overfits its training data tends to be unable to generalise to unseen data, and this is very clear when looking at the black (true) and red (estimate) curves: they intersect at the data points, and then diverge at values of x that were not in the data.

M	w_0	w_1	w_2	w_3	w_4	w_5	w_6	w_7	w_8	w_9
9	-0.66	10.98	25.62	-117.80	-143.29	405.10	246.74	-561.32	-127.91	263.129

Table 2: Coefficients of a high-order polynomial fit to noisy data show characteristic large values of high-order coefficients.

We can see one of the consequences of this if we repeat this for a different sampling of noise, as shown in Figure 5

This is instructive because it shows us two important things:

1. Low-order models tend to be reasonably robust to noise, but do not fit the data well.
2. High-order model fit the data very well, but can be very sensitive to noise, giving different results for different noise realisations.

These observations imply that there is some trade-off between models that are robust, and models that are accurate. Why is this

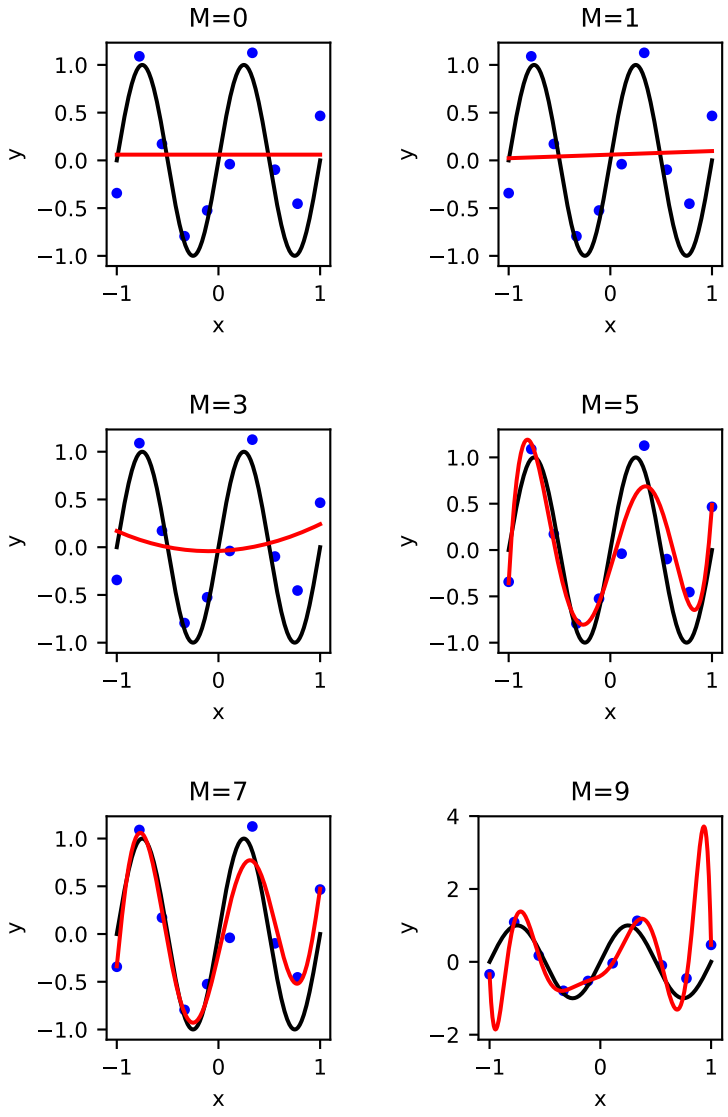

Figure 5: Fitting $y = \sin(2\pi x)$ with polynomial fits of increasing degree M to data with a different realisation of added noise. The blue line represents the true function; the black points are the sampled points (x_i, y_i); the red line is the best-fit polynomial of that order.

so? To understand, we turn to one of the most important results in machine learning: the *bias-variance decomposition*.

The Bias-Variance Decomposition

Let us ask a very simple question (using a simplified notation for convenience):given data y generated by an underlying function $h(x) + \epsilon$, and an estimated model $f(x)$, what is the expected value (i.e. the average) of the least-squares loss, $\mathcal{L} = (y - f)^2$? Since the data is a random variable drawn from some probability distribution, both the loss function and the predictions of the estimated model must also be random variables and will therefore have some distribution of values with an expected value. Writing $f(x) = f$ and $h(x) = h$ for simplicity, the expected loss is given by

$$\begin{aligned} \mathbb{E}[\mathcal{L}] &= \mathbb{E}[(y - f)^2] && (3) \\ &= \mathbb{E}[y^2] + \mathbb{E}[f^2] - 2\mathbb{E}[yf] && (4) \end{aligned}$$

We will rewrite this using the definition of the variance of a random variable:

$$\text{var}[X] = \mathbb{E}[X^2] - (\mathbb{E}[X])^2, \tag{5}$$

and a property of two independent variables

$$\mathbb{E}[XY] = \mathbb{E}[X]\mathbb{E}[Y] \tag{6}$$

The expected loss then becomes

$$\mathbb{E}[\mathcal{L}] = \text{var}[y] + (\mathbb{E}[y])^2 + \text{var}[f] + (\mathbb{E}[f])^2 - 2\mathbb{E}[y]\mathbb{E}[f] \tag{7}$$

Now we can apply what we know about the problem. Since $y = h(x) + \epsilon$, then under the assumption that $\mathbb{E}[\epsilon] = 0$ and $\text{var}[\epsilon] = \sigma^2$, then $\mathbb{E}[y] = h$ and $\text{var}[y] = \sigma^2$. The expected loss becomes

$$\begin{aligned} \mathbb{E}[\mathcal{L}] &= \sigma^2 + h^2 + \text{var}[f] + (\mathbb{E}[f])^2 - 2h\mathbb{E}[f] && (8) \\ &= \sigma^2 + \text{var}[f] + h^2 + (\mathbb{E}[f])^2 - 2h\mathbb{E}[f] && (9) \\ &= \sigma^2 + \underbrace{\text{var}[f]}_{\text{variance}} + \underbrace{(h - \mathbb{E}[f])^2}_{\text{bias}} && (10) \end{aligned}$$

How can we interpret this result? First, we notice that the data is no longer explicitly present in this expression, only a measure of its variance σ^2. All dependence on the *specific sample*, y, of the data has been absorbed into the other terms. In particular, the variance of f is a consequence of the variance in the data: if the data has no noise, we will always learn the same model, but different samples will lead to different models. Therefore, the term $\text{var}\, f$ represents the degree to which the estimated model is sensitive to the choice of data.

The third term in the expression, $(h(x) - \mathbb{E}[f(x)])^2$, is the square-difference between the true underlying model h and the "average" estimated model f. This term represents the ability of the estimated model to accurately represent the true model: it is the *bias* of the

estimate. For examples, fitting a linear function $f(x) = mx * c$ to $h(x) = \sin(2\pi x)$ has a high bias, because it cannot accurately match the true underlying function.

Minimising the expected loss is achieved by choosing a model that simultaneously minimises the dependence on the data sampling, and the ability of the model to represent the data. This result demonstrates that these two desires conflict with each other. A very complex model will have a low bias, because it can represent the data very accurately, but a high variance, because a different sampling of the data will give a completely different model. A very simple model will conversely have a low variance but a high bias. The need to simultaneously minimise both terms, and the trade-offs that are necessary is at the core of modern learning theory.

The implication of this results is that the choice of model for our estimator $f(x)$ matters a lot. If we know the data generating function $h(x)$, then we can make the principled choice $f = h$. If, however, we do not know h then we have to determine the best model empirically. We will now consider some techniques for doing this.

Reading

Sections 1.1 and 3.2 of Bishop, Pattern Recognition and Machine Learning.