Introduction to Machine Learning (67577)

Recitation 05
Polynomial Fitting & Bias-Variance

Second Semester, 2021

Contents

1 Polynomial Fitting

Let us expand the model of linear regression. Suppose that instead of modeling the linear relation between $\mathcal{X}$ and $\mathcal{Y}$ as $y = \mathbf{x}^\top \mathbf{w} + \varepsilon$ we introduce a set of functions $h_1, \ldots, h_k$ such that $h_j : \mathbb{R}^d \to \mathbb{R}$. We then define the relation as:

$$y_i = \sum_{j=1}^{k} h_j(\mathbf{x}_i) \cdot \mathbf{w}_j$$

These functions $h_1, \ldots, h_k$, that are referred to as *basis functions*, enable us to describe a relation that is **linear in the parameters w** but could be non-linear in the original input $\mathbf{x}$. One specific case is of polynomial fitting. Let $x_1, \ldots, x_m \in \mathbb{R}$ and $y_1, \ldots, y_m \in \mathbb{R}$. We would like to describe a polynomial relation between $\mathcal{X}$ and $\mathcal{Y}$, of degree at most $k \in \mathbb{N}$. The hypothesis class is:

$$\mathcal{H}^k_{poly} = \left\{ x \mapsto p_{\mathbf{w}}(x) \,|\, \mathbf{w} \in \mathbb{R}^{k+1} \right\} \tag{1}$$

where $p_{\mathbf{w}}(x) = \sum_{i=1}^{K} w_i x^i$. In this case, we define the set of basis functions to be $h_j(x) = x^j$ for any $j \in \{0, \ldots, k\}$. Then denote $h(x) := (h_0(x), \ldots, h_k(x))^\top$. Thus, a polynomial of degree k is given by:

$$p_{\mathbf{w}}(x) = \sum_{j=0}^{k} h_j(x)\,\mathbf{w}_j = \langle h(x), \mathbf{w} \rangle = h(x)^\top \mathbf{w}$$

As before we want to find a coefficients vector $\mathbf{w}$. Given a sample $S = \{(\mathbf{x}_i, y_i)\}_{i=1}^m$ we transform it to a sample $\widetilde{S} := \{(h(x_i), y_i)\}_{i=1}^m$, and solve the following LS problem:

$$\widehat{w} := \underset{\mathbf{w} \in \mathbb{R}^{k+1}}{argmin} \frac{1}{m} \sum_{i=1}^{m} \left(h(x_i)^\top \mathbf{w} - y_i \right)^2$$

Notice that the design matrix $\mathbf{X}$ defined over the transformed data is the Vandermonde matrix:

$$\mathbf{X} = \begin{bmatrix} - & h(x_1) & - \\ - & h(x_2) & - \\ & \vdots & \\ - & h(x_m) & - \end{bmatrix} = \begin{bmatrix} 1 & x_1 & x_1^2 & \cdots & x_1^k \\ 1 & x_2 & x_2^2 & \cdots & x_2^k \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & x_m & x_m^2 & \cdots & x_m^k \end{bmatrix}$$

Since we assume that the x_i's are different from one another, the design matrix $\mathbf{X}$ is of full rank. This means that solving this linear (in $\mathbf{w}$) system of equations can be done as we have seen for the non-singular case above.

R Here we have seen polynomial fitting where $\mathcal{X} = \mathbb{R}$. With very little adaptation, we could also allow the input data to be $\mathcal{X} = \mathbb{R}^d$, $d > 1$. In such cases the defined polynomial could include terms of multiplication of two (or more) features. We will encounter such an example in **??**.

2 Bias and Variance of Estimators

Given a regression problem $\mathbf{X}, \mathbf{y}$ we have seen how to solve $\mathbf{y} = \mathbf{X}\mathbf{w} + \varepsilon$. That is, finding a vector $\mathbf{w}$ that satisfies: $\mathbf{y} \approx \mathbf{X}\mathbf{w}$. We showed that the vector minimizing the sum of square distances is given by

$$\widehat{\mathbf{w}}^{ols} = \left[\mathbf{X}^\top \mathbf{X} \right]^{-1} \mathbf{X}^\top \mathbf{y}$$

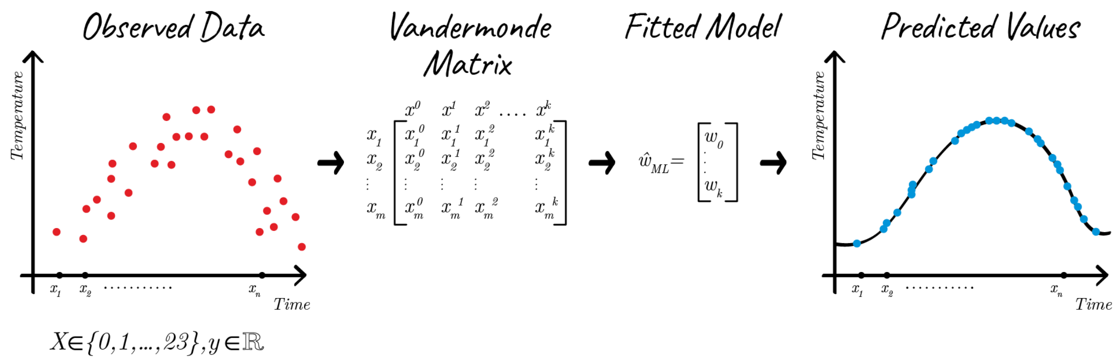


Figure 1: Scheme of Polynomial Fitting: *Dataset of hourly temperature where x_i denotes time of day and y_i the temperature. Fitting polynomial of degree* 4.

As $\mathbf{y}$ is a random variable, the $\widehat{\mathbf{w}}^{ols}$ estimator is a random variable. Therefore, we could look at different properties of such estimator. Specifically, we will look at the *bias* and *variance* of an estimator.

Definition 2.1 Let $\widehat{\theta}$ be an estimator of θ. The *bias* of $\widehat{\theta}$ is the expected deviation between θ and the estimator: $B\left(\widehat{\theta}\right) := \mathbb{E}\left[\widehat{\theta}\right] - \theta$. $\widehat{\theta}$ is said to be *unbiased* if $B\left(\widehat{\theta}\right) = 0$.

Definition 2.2 Let $\widehat{\theta}$ be an estimator of θ. The variance of $\widehat{\theta}$ is the expected value of the squared sampling deviations: $var\left(\widehat{\theta}\right) := \mathbb{E}\left[\left(\widehat{\theta} - \mathbb{E}\left[\widehat{\theta}\right]\right)^2\right]$.

What is the expectation in both the bias (2.1) and the variance (2.2) been calculated over? An estimator is a function over a sample $S = x_1, \ldots, x_m \in \mathbb{R}^d$ used to estimate some parameter: $\widehat{\theta}\left(x_1, \ldots, x_m\right) \overset{?}{\approx} \theta$. As such, the expectation of $\widehat{\theta}$ is over the selection of the samples. Going back to the $\widehat{\mathbf{w}}^{ols}$ estimator, we could ask what is it's bias and variance.

Exercise 2.1 Let $\mathbf{X}, \mathbf{y}$ be a regression problem such that $\mathbf{y} = \mathbf{X}\mathbf{w} + \varepsilon$, $\varepsilon \sim \mathcal{N}\left(0, \sigma^2 I_d\right)$ and $\widehat{\mathbf{w}}$ being the OLS estimator. Show that $\widehat{\mathbf{w}}$ is an unbiased estimator. ■

Proof.

$$\begin{aligned}
\mathbb{E}\left[\widehat{\mathbf{w}}\right] &= \mathbb{E}\left[\left[\mathbf{X}^\top\mathbf{X}\right]^{-1}\mathbf{X}^\top\mathbf{y}\right] \\
&= \mathbb{E}\left[\left[\mathbf{X}^\top\mathbf{X}\right]^{-1}\mathbf{X}^\top\left(\mathbf{X}\mathbf{w} + \varepsilon\right)\right] \\
&= \mathbb{E}\left[\left[\mathbf{X}^\top\mathbf{X}\right]^{-1}\mathbf{X}^\top\mathbf{X}\mathbf{w}\right] + \mathbb{E}\left[\left[\mathbf{X}^\top\mathbf{X}\right]^{-1}\mathbf{X}^\top\varepsilon\right] \\
&= \mathbb{E}\left[\mathbf{w}\right] + \left[\mathbf{X}^\top\mathbf{X}\right]^{-1}\mathbf{X}^\top\mathbb{E}\left[\varepsilon\right] = \mathbf{w}
\end{aligned}$$

where the last equality is because $\mathbb{E}\left[\varepsilon\right] = 0$ and $\mathbf{w} \in \mathbb{R}^d$. ■

To get some intuition about these two properties, let us revisit polynomial fitting. Consider the polynomial $y = x^4 - 2x^3 - 0.5x^2 + 1 + \varepsilon$ for $\varepsilon \sim \mathcal{N}(0,2)$. In Figure 2 we choose a set of x values and create 10 different datasets from this model, keeping the x values consistent between datasets but sampling the noise each time: $\left\{\left\{(\mathbf{x}_i, y(\mathbf{x}_i) + \varepsilon_{ij})\right\}_{i=1}^{m}\right\}_{j=1}^{10}$, $\varepsilon_{ij} \overset{iid}{\sim} \mathcal{N}(0,2)$. Then, we fit a polynomial of degree 1. Black, red and blue points represent the true model, the observed data-points (with the sample noise) and the fitted model over the observed data-points. Notice how the different datasets yield different predicted models. This is the randomness of the prediction, driven by the randomness of the trainset. Over these datasets we can now ask, for each value of x, what is the average prediction and its variance:

- In green is the average prediction of y for a given x across all datasets. The difference between the green and black lines capture the concept of the bias.
- In grey is the area of $\mathbb{E}[\widehat{y}] \pm 2 \cdot Var(\widehat{y})$ for a given x, also known as the confidence interval. The wider this area is, the more out prediction of $\widehat{y}$ varies for different samples. This area captures the concept of the variance.

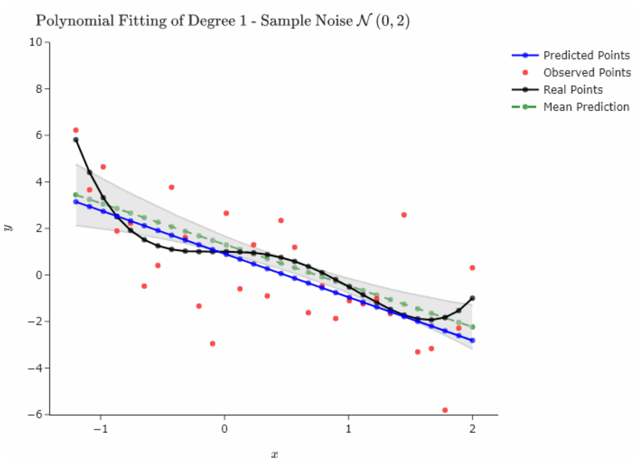


Figure 2: Polynomial Fitting: *Fitted polynomial of degree* 1 *over different datasets differing only in values of added sample noise. Chapter 2 Examples - Source Code*

Two phenomena are visible. The first is that the average distance of the fitted model (in green) and the true model (in black) is large. This means that our hypothesis class doesn't have sufficient expressive power to learn the true model. As such, we conclude that the **bias** of our estimator is high. The second is that the fitted models over different datasets do not differ by much. As such, we conclude that the **variance** of our estimator is low.

Next, consider the same setup as before but with the fitting of a polynomial of degree 8 (Figure 3). This time the different between the average prediction at each x and the true value of x is lower, while the differences between the fitted models (as indicated by the confidence intervals) is much higher. So the **bias** is low and the **variance** is high. As we enable more "flexible" (i.e. complex) models we are able to fit a model better to our given sample. However, as seen in Figure 3, if the model is too complex we might actually be fitting a model to the noise, rather than the actual true signal.

R It is important to note that what is seen in the figures are not the bias and variance of $\widehat{\mathbf{w}}^{ols}$

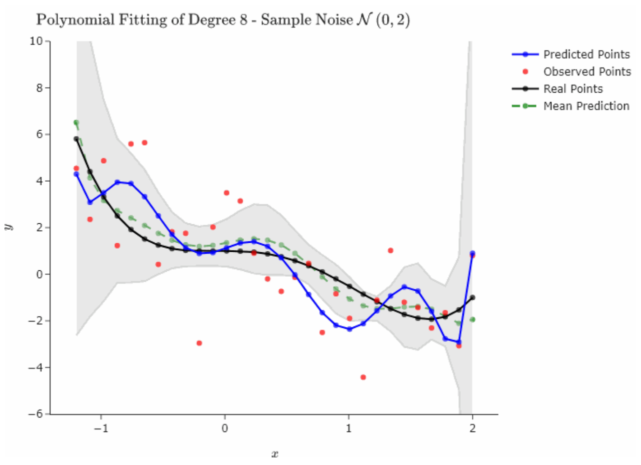


Figure 3: Polynomial Fitting: *Fitted polynomial of degree* 8 *over different datasets differing only in values of added sample noise.* *Chapter 2 Examples - Source Code*

themselves but how these manifest over the shown datasets.

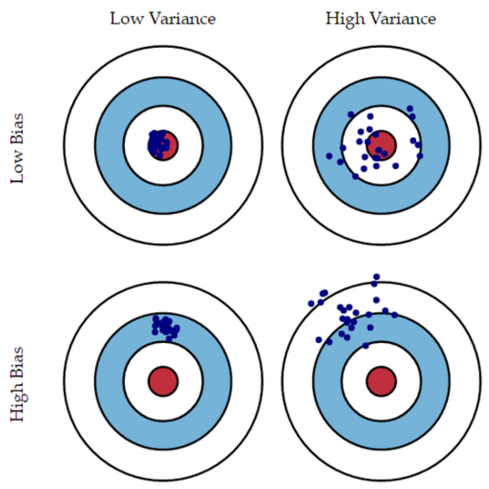
Although using an unbiased estimator seems like a good idea, this is generally not the case. To see why, suppose we use quadratic loss, so the corresponding risk is the MSE. We now derive a very useful decomposition of the MSE. (All expectations and variances are wrt the true distribution, but we drop the explicit conditioning for notational brevity.) Let $\hat{\mathbf{y}} = \hat{\mathbf{y}}(S)$ denote the least squares estimator, and $\bar{\mathbf{y}} = \mathbb{E}(\hat{\mathbf{y}})$ denote the expected value of the estimator, and let $\mathbf{y}^*$ be the ground truth values we are looking for. Then we have

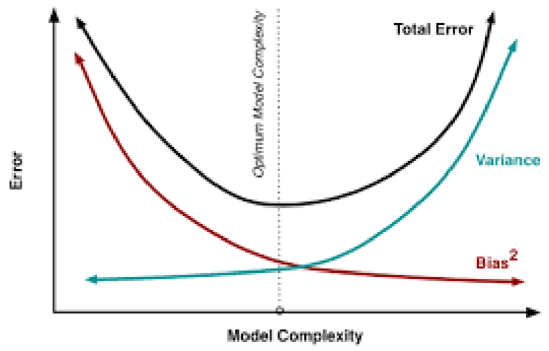
$$\begin{aligned}
\mathbb{E}\left[(\hat{\mathbf{y}}-\mathbf{y}^*)^2\right] &= \mathbb{E}\left[\left[(\hat{\mathbf{y}}-\bar{\mathbf{y}})+(\bar{\mathbf{y}}-\mathbf{y}^*)\right]^2\right] \\
&= \mathbb{E}\left[(\hat{\mathbf{y}}-\bar{\mathbf{y}})^2\right]+2(\bar{\mathbf{y}}-\mathbf{y}^*)\mathbb{E}[\hat{\mathbf{y}}-\bar{\mathbf{y}}]+(\bar{\mathbf{y}}-\mathbf{y}^*)^2 \\
&= \mathbb{E}\left[(\hat{\mathbf{y}}-\bar{\mathbf{y}})^2\right]+(\bar{\mathbf{y}}-\mathbf{y}^*)^2 \\
&= \mathrm{Var}[\hat{\mathbf{y}}]+\mathrm{bias}^2[\hat{\mathbf{y}}] \qquad (2)
\end{aligned}$$

In words,

$$\mathrm{MSE} = \mathrm{Variance} + \mathrm{bias}^2 \qquad (3)$$

This is called the bias-variance trade off. What it means is that it might be wise to use a biased estimator, so long as it reduces our variance, assuming our goal is to minimize squared error.





3 Feature Selection

■ Example 3.1 Let us find the OLS estimator over some mock scenario. Suppose we are interested in regressing running times in a $100m$ on an athlete's height and weight. We gathered the details of the 8 athletes:

Athlete	Weight (*kg*)	Height (*cm*)	Leg's Length (*cm*)	Foot Size (*cm*)	Running Time (*sec*)
Usain Bolt	94	194	80	32	9.81
Justin Gatlin	90	190	86	30	9.89
Andre de Grasse	85	185	91	28	9.91
Yohan Blake	80	180	82	31	9.93
Evelyn Ashford	82	182	89	35	9.94
Glenn Cunningham	65	165	72	33	9.95
Barney Ewell	78	178	83	34	9.98
Jesse Owens	70	170	74	30	10.01

So the features are the *weight, height, leg's length and foot size* and the response is *running time*. Let us fit an OLS model for this data. We begin with arranging it in a matrix and adding the intercept:

$$\mathbf{X} = \begin{bmatrix} 1 & 94 & 194 & 80 & 32 \\ 1 & 90 & 190 & 86 & 30 \\ 1 & 85 & 185 & 91 & 28 \\ 1 & 80 & 180 & 82 & 31 \\ 1 & 82 & 182 & 89 & 35 \\ 1 & 65 & 165 & 72 & 33 \\ 1 & 78 & 178 & 83 & 34 \\ 1 & 70 & 170 & 74 & 30 \end{bmatrix}, \quad \mathbf{y} = \begin{bmatrix} 9.81 \\ 9.89 \\ 9.91 \\ 9.93 \\ 9.94 \\ 9.95 \\ 9.98 \\ 10.01 \end{bmatrix}$$

As we have proven above, the OLS estimator is given by the closed form of $\widehat{\mathbf{w}} = \left(\mathbf{X}^\top \mathbf{X}\right)^{-1} \mathbf{X}^\top \mathbf{y}$. But

now we have 5 features for only 8 records. Therefore, to avoid overffiting, we need a model that uses only some of the features. This brings us to the question, which features to choose.
In most of the cases, we are going to choose features using heuristics. Our first heuristic is going to be picking the features with the best correlation to **y**. We want the dummy feature anyway, so we need to calculate the correlation of the rest of the features.
$cor(\mathbf{X},\mathbf{y}) = [-0.82,-0.82,-0.29,0.09]$
So it seems like foot size is irrelevant to our problem and doesn't give us a lot of info on our prediction problem. So selecting the remaining features might be a good option.
But, in contrast to other models, linear regression is a model that can be calculated really fast. Thus, we can use it to get a better feature selection heuristic:

S is the set of features we already chose. Initially it is only the dummy feature.
F is the set of feature we are not using. Initially it is all the features except the dummy feature.
1. $w = LinearRegression(\mathbf{X}[S],\mathbf{y})$
2. $\hat{y} = X[S]\dot{w}$
3. $z = \hat{y} - \mathbf{y}$
4. Find the feature in F with the best correlation to z and move it to S.
As long as the correlation you found is not too close to 0, do $\mathbf{y} = z$ and go back to 1.

At each iteration, we can predict $\hat{y}$, so what we have left to predict is the direction from $\hat{y}$ to **y** - that is z. Therefore what we need from the rest of the features is to predict z. Thus we choose the feature that predicts z in the best way.
In our example, the first feature we pick is the height, since -0.82 is a high correlation. Now
$w = [10.03,-0.00514]$

$$\hat{y} = \begin{bmatrix} 9.858 \\ 9.879 \\ 9.904 \\ 9.93 \\ 9.92 \\ 10.007 \\ 9.94 \\ 9.981 \end{bmatrix}$$

$$z = \begin{bmatrix} 0.0481 \\ -0.0113 \\ -0.0056 \\ 0 \\ -0.0202 \\ -0.0571 \\ -0.0397 \\ -0.0286 \end{bmatrix}$$

Now $cor(\mathbf{X}[2:],z) = [0,-0.44,0.19]$. Pay attention that now the height has a really low correlation. This is because the height is a linear combination of the first two features: $\mathbf{x}_3 = 100\mathbf{x}_1 + \mathbf{x}_2$. Therefore, after we have the weight, the height doesn't give us much information. In this data, $\mathbf{x}_3$ is exactly a linear combination of $\mathbf{x}_1,\mathbf{x}_2$. But in real life a feature that is really close to be a linear combination of other features, might get us to the same results. This is why this heuristic is usually better.

In this way we can choose to stop when the correlation to z is too low for us, so we conclude that we have all the info that we can and all that is left is just noise. In this way we can produce large number of features, and only pick the feature that correlate to our goal.

$\blacksquare$