Handout 11: Regression Splines

Non-linearity

Linear regression has excellent theoretical properties and, as we have seen, can be readily computed from observed data. Using ridge regression and principal component analysis we can tune these models to optimize for predictive error loss. Indeed, linear models are used throughout numerous fields for predictive and inferential models. One situation in which linear models begin to perform non-optimally is when the relationship between the response y and the data is not linear nor can it be approximated closely by a linear relationship.

As an example of a non-linear model consider observing a variable y_i governed by

$$y_i = \cos(\beta_1 \cdot x_i) + e^{-x_i \cdot \beta_2} + \epsilon_i \tag{11.1}$$

for some scalar value x_i, unknown constants β_1 and β_2, and the random noise variable ϵ_i. A common approach for estimating the unknown parameters given a set of observations is to again minimize the sum of squared residuals. This sum is a well-defined function over the set of allowed β_j's and often, as in this case, twice differentiable. While there is no analogous closed-form solution to the linear case, the minimizing estimate values can usually be found using a general purpose first, or second-order optimization technique. This approach is known as *non-linear least squares* and has significant theoretical guarantees over a wide class of problem formulations.

What happens when we do not know a specific formula for y_i that can be written down in terms of a small set of unknown constants β_j? Models of the form seen in Equation 11.1 often arise in engineering and science applications where the specific causal mechanism for the response y_i is well understood. In statistical learning this is rarely the case. More often we just know that the expected value is equal to some function

$$\mathbb{E}y_i = f(x_i) \tag{11.2}$$

holds for some unknown function f. We may suspect that f has some general properties; depending on the application it may be reasonable to assume that f is continuous, has a bounded derivative, or is monotonically increasing in x_i. As we do not know a specific formula for f in terms of parameters β_j, the model given in Equation 11.1 is known as non-parametric regression. Common estimators for estimating the non-parametric regression function f are the topic of this chapter.

In parametric regression it is clear that a point estimator will yield a single prediction $\widehat{\beta}$ inside of $\mathbb{R}^p$ for the unknown regression vector. For non-parametric regression it is not even clear what an estimator $\widehat{f}$ would look like. Two methods we evaluate

will produce an explicit formula in terms of x_i as a prediction of y_i. The other two instead provide an algorithm for computing $\widehat{f}(x)$ for any input value of x. While it is possible to do this for a large set of x's, these techniques will not yield an estimated parametric model such as that given in Equation 11.1. The added computational time for predictions, which often require non-trivial computations for every value of x should be taken into account when considering the value added to the predictive performance by a non-parametric model. We will discuss techniques for minimizing the burden of the prediction time.

As with most predictive models, non-parametric regression techniques have tuning parameters to control the trade-off between variance and bias. Often this can be visualized in terms of the smoothness of the function $\widehat{f}$. If the regression function is penalized from changing too fast, this reduces the variance in the estimates but introduces additional bias for values at x_i where it *should* vary more. Likewise, if the function is allowed to change rapidly this will generally give nearly unbiased estimates at the price of a high variance in the estimated values. Much of our discussion about non-parametric regression will focus on methods for setting tuning parameters and the performance of these methods on various types of data.

Basis expansion

Recall that linear regression requires only that the relationship with respect to the parameter β_j be linear. The terms multiplied by each parameter may be any known quantity derivable from the data matrix X. For example, with a scalar value of x_i both

$$y_i = \sum_{k=0}^{K} x_i^k \cdot \beta_j + \epsilon \tag{11.3}$$

and

$$y_i = \sum_{k=1}^{K} sin(x_i/(2\pi K)) \cdot \beta_j + \epsilon \tag{11.4}$$

are valid linear regression models. The first corresponds to the first K terms of the polynomial basis and the second to the first K odd terms of the Fourier basis. If we construct a new matrix Z consisting of columns that are copies of x taken to various powers or to varying applications of the sine function, an estimate of the relationship between y and x can be determined using the standard techniques for calculating a linear regression model. In general we model the relationship

$$y_i = \sum_{k=1}^{K} B_{k,K}(x_i) \cdot \beta_j + \epsilon \tag{11.5}$$

for some basis function $B_{k,K}$. This method is known as a *basis expansion*.

Regression splines

Here we discuss an important set of basis functions, known as splines, that are particularly well-suited to this task. While technically just a specific application of basis expansion, the derivation of the splines is subtle enough and their application sufficiently important to warrant a separate treatment of their form and usage.

When using a polynomial or Fourier basis to represent a non-linear function, small changes in a coefficient will lead to changes in the predicted values at every point of the unknown regression function f. The global nature of the estimation problem in these cases leads to poor local performance in the presence of high noise variance or with regression functions f that have many critical points. Local regression, as we have seen, offers a solution to this problem by using a relatively small basis expansion locally at each point. A downside of this approach, however, is that it requires fitting a regression model for every desired point where a prediction is needed. The computational load of this calculation can become considerable and in the case of larger datasets, the requirement that all training data must be accessible to run this regression may also limit its application. Fortunately there is a way of doing basis expansion that mimics the primary benefits of local regression.

We start by picking a point k within the range of the data points x_i. A natural choice would be the median or mean of the data. In lieu of a higher-order polynomial fit, imagine fitting two linear polynomials to the data: one for points less than k and another for points greater than k. Using indicator functions, we can describe this approach with a specific basis expansion, namely

$$B_0(x) = I(x \leq k) \tag{11.6}$$
$$B_1(x) = x \cdot I(x \leq k) \tag{11.7}$$
$$B_2(x) = I(x > k) \tag{11.8}$$
$$B_3(x) = x \cdot I(x > k). \tag{11.9}$$

It will be useful going forward to re-parameterize this in terms of a baseline intercept and slope for $x \leq k$ and changes in these values for points $x > k$

$$B_0(x) = 1 \tag{11.10}$$
$$B_1(x) = x \tag{11.11}$$
$$B_2(x) = I(x > k) \tag{11.12}$$
$$B_3(x) = (x - k) \cdot I(x > k). \tag{11.13}$$

It is left as an exercise to show that these are equivalent bases.

A shortcoming of the space spanned by these splines is that at the point k, known as a *knot*, the predicted values will generally not be continuous. It is possible to modify our original basis to force continuity at the knot k by removing the secondary

intercept described by $B_2(x)$ in Equations 11.10–11.13. The basis now becomes

$$B_0(x) = 1 \tag{11.14}$$
$$B_1(x) = x \tag{11.15}$$
$$B_2(x) = (x-k) \cdot I(x > k). \tag{11.16}$$

Notice that forcing one constraint, continuity at k, has reduced the degrees of freedom by one, from 4 down to 3. How might we generalize this to fitting separate quadratic term on the two halves of the data? One approach would be to use the basis functions

$$B_0(x) = 1 \tag{11.17}$$
$$B_1(x) = x \tag{11.18}$$
$$B_2(x) = x^2 \tag{11.19}$$
$$B_3(x) = (x-k) \cdot I(x > k) \tag{11.20}$$
$$B_4(x) = (x-k)^2 \cdot I(x > k). \tag{11.21}$$

The number of parameters here works out correctly; we have two quadratic polynomials (2×3) minus one constraint, for a total of $6 - 1 = 5$ degrees of freedom. What will a function look like at the knot k using the basis from Equations 11.17–11.21? It will be continuous at the knot but is not constrained to have a continuous derivative at the point. This is easy to accomplish, however, by removing the $B_3(x)$ basis. Notice that once again the inclusion of an additional constraint, a continuous first derivative, reduces the degrees of freedom by one.

Defining the positive part function $(\cdot)_+$ as

$$(x)_+ = \begin{cases} x, & x \geq 0 \\ 0, & \text{otherwise} \end{cases} \tag{11.22}$$

we may generalize to an arbitrarily large polynomial of order M by using the basis

$$B_0(x) = 1 \tag{11.23}$$
$$B_j(x) = x^j, \quad j = 1, \ldots, M \tag{11.24}$$
$$B_{M+1}(x) = (x-k)_+^M \tag{11.25}$$

This basis results in a function with continuous derivatives of orders 0 through $M-1$. We can further generalize this by considering a set of P knots $\{k_p\}_{p=1}^P$, given by

$$B_0(x) = 1 \tag{11.26}$$
$$B_j(x) = x^j, \quad j = 1, \ldots, M \tag{11.27}$$
$$B_{M+p}(x) = (x-k_p)_+^M, \quad p = 1, \ldots, P \tag{11.28}$$

Equations 11.26–11.28 defines the *truncated power basis* of order M. It yields piecewise Mth order polynomials with continuous derivatives of order 0 through $M-1$.

Note that once again the degrees of freedom math works out as expected. There are $P+1$ polynomials of order M and P sets of M constraints; the truncated power basis has $(P+1)(M+1)-PM$, or $1+M+P$, free parameters.

Now that we have defined these basis functions, we can fit a regression model to learn the representation of the unknown function $f(x)$ by minimizing the sum of squared residuals over all functions spanned by this basis. This is equivalent to the basis expansion we used at the start of these notes. When used over the spline basis, the resulting estimator is known as a *regression spline*. As with any basis expansion, we can compute the solution by explicitly constructing a design matrix G as

$$G_{i,j} = B_{j-1}(x_i), \quad i = 1, \dots n,\ j = 1, \dots, 1+M+P. \tag{11.29}$$

Then, to calculate $\widehat{f}(x_0)$, we simply compute the basis expansion at x_{new}

$$g_i = B_{j-1}(x_0), \quad i = 1, \dots n,\ j = 1, \dots, 1+M+P. \tag{11.30}$$

The regression spline can be written in this basis using a vector $\beta \in \mathbb{R}^{1+M+P}$

$$\widehat{f}(x) = \sum_{j}^{1+M+P} \widehat{\beta}_j B_{j-1}(x) \tag{11.31}$$

where $\widehat{\beta}$ is given by

$$\widehat{\beta} = (G^t G)^{-1} G^t y. \tag{11.32}$$

We can then compute $\widehat{f}$ this for any new point x_0, thus providing an estimate of the entire function f.

By far the most commonly used truncated power basis functions are those with M equal to three. These are justified by the empirical evidence that higher order rarely offer performance gains and that human observers are unable to detect changes in the third derivative of a function (the idea being that you will not be able to point out the knots in a cubic spline).