1 Moving Beyond Linearity

In this section, we explore some modifications to the linear regression model in order to incorportate some non linearity as well for reducing bias/

1.1 Polynomial Regression

This is a simply modification to the original linear regression setting where we incorporate higher powers of the predictor as well. This helps incorporate non linear trends in the data as well while retaining an additive relationship between the variables.

$$\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 x \qquad \text{Linear Regression}$$
$$\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 x + \hat{\beta}_2 x^2 + \cdots + \hat{\beta}_d x^d \qquad \text{Polynomial Regression}$$

Generally, is is **unusual to have d to be greater than 3 or 4** because higher degree of polynomial makes the model highly flexible with very curvy shapes. This may reduce bias to a large extent but the variance obtained thus is quite high, expecially near the end values of the range.

$$\begin{aligned} \hat{C} &= Cov(\hat{\beta}, \hat{\beta}) \\ l_i &= (1, x_i, x_i^2, \ldots, x_i^d)^T \\ Var[\hat{y_i}] &= l_i^T \hat{C} l_i \end{aligned}$$

The same form of polynomial regression terms works in the context of classification as well for Logistic Regression.

1.2 Step Function

Polynomial Regression, like Linear Regression, still imposes a global structure on the data. The functional form of the model is same throughout. Step Function on the other hand has a local structure and divided the range into k **intervals such that each interval is fitted with a different constant**.

$$\begin{aligned} C_0(X) &= I(X < c_1) \\ C_1(X) &= I(c_1 \leq X < c_2) \\ C_2(X) &= I(c_2 \leq X < c_3) \\ &\cdots \\ C_{k-1}(X) &= I(c_{k-1} \leq X < c_k) \\ C_k(X) &= I(c_k \geq X) \\ \hat{y} &= \beta_0 + \beta_1 C_1(x) + \cdots + \beta_k C_k(x) \end{aligned}$$

Note that, β_0 and $C_0(x)$ are equivalent since both will act as the intercept and the mean value of y in the range $x < c_1$. Similarly, each of the β_i captures the average of the response in the corresponding interval defined by the indicator function.
This approach works well when there are natural breakpoints in the data and if there are indeed constant trends in those intervals. As soon as a non constant trend emerges locally, the approach fails.

1.3 Basis Function

Basis function is the generalized form of the above two approaches. Basis are functions on X that transform it. Thus, when doing regression with basis functions, we simply regress y with a family of transformations of x.
For polynomial regression, the basis are powers of x and for step function, the basis are constant values in different ranges.

1.4 Regression splines (Polynomials)

This is a class of methods that extends upon the family of polynomial and step regressions.

Instead of fitting a global polynomial to X, we fit **local piecewise polynomials** to X such that they are continuous at the breakpoints. The points wherer the polynomials change are called **knots**.

As we increase the number of knots, we can get a more flexible predictor. In general, if there are K knots, we have $K + 1$ polynomials.

Having this many splines creates a lot of degrees of freedom in the data. To solve these many equations, we need to impose constraints of **continuity and continuity of derivatives at knots**. In general, for a d degree spline, we will impose the function to be continuous at knots and also have the $d-1$ derivatives to be equal at the knots. The reduced degrees of freedom are essential for obtaining unique solutions to the coefficients such that to overall curve still looks continuous.

Instead of fitting multiple splines, we can also defined the function in a single format as follows. A truncated power basis function is defined as

$$h(x, \xi) = (x - \xi)_+^d = \begin{cases} (x - \xi)^d & \text{if } x > \xi \\ 0 & 0 \text{ otherwise} \end{cases}$$

$$\hat{y} = \beta_0 + \beta_1 b_1(x) + \beta_2 b_2(x) + \cdots + \beta_{K+d} b_{K+d}(x)$$

where the terms $b_1, \ldots, b_d$ are the usual terms $x, x^2, \ldots, x^d$ and the remaining terms correspond to a truncated power function per knot. This ensures that the polynomials are continuous at the knots upto the $d - 1$ derivates.

A **natural spline** is a regression spline that has the additional constraint of being linear at the boundaries, i.e., outside the extreme knots. This reduced flexibility allows for stable estimates at the knots and also makes the confidence bands narrower.

1.4.1 Choosing the degrees of freedom

While it may make sense to put knots at the points where the function seems to change rapidly (and not at the points where the function seems to be relatively stable), in practice the knots are usually placed at equal quantiles of the data, or generally are placed in an equidistant fashion. For deciding the degree of freedoms, another popular choice is cross validation. We always use say 10% of the data as test and gauge the RSS on this set across different degrees of freedom. A plot of RSS vs. the degrees of freedom (similar to elbow curve) can aid choosing the desired point where the RSS significantly drops.
In practice, cubic splines are popular as higher order polynomials tend to give high variance, and the plots with cubic splines also look relatively stable.

1.5 Smoothing Splines

The idea is to fit a smooth function that predicts the reponse well. We minimize the following

$$Loss = \sum_{i=1}^{n}(y_i - g(x_i))^2 + \lambda \int g''(t)^2 dt \qquad \lambda > 0$$

which can be decomposed into two parts, the first part encourages the function g to fit the data well and the second part encourages the function to be smooth throughout.
If the function is constant or straight line, the double derivative is zero while on the other extreme, if the function changes rapidly, double derivative will also take large values.

When λ is zero, the function simply takes the values of y and thus becomes very jumpy (introducing high variance). On the other extreme, if $\lambda \to \text{inf}$, the function is forced to be linear (which makes the double derivative zero). This is exactly the least square fit that we have seen earlier.

The function that minimizes the above loss is nothing but a cubic spline with knots at $x_1, x_2, \ldots, x_n$ with the additional constraint that the first and second derivatives are smooth at the knots. (The function is cubic in between the points). Outside the extreme knots, the function simply takes a linear form.
In some sense, this is similar to the natural splines but is a shrinked version of the same, λ controlling the level of shrinkage (or how large the roughness can be).

$$\hat{g}_\lambda = S_\lambda y$$
$$df_\lambda = \sum_{i=1}^{n}\{S_\lambda\}_{ii}$$

where $\hat{g}_\lambda$ is a n-vector containing the values at the points x_is as a solution to a particular value of λ. S_λ is a $n \times n$ matrix obtained as the solution of the above error function.

1.5.1 Choosing Reguularization Parameter

It is possible to show that as λ increases from $0 \to \text{inf}$, the effective degrees of freedom (df_λ) go from $n \to 2$.

Cross validation can also be a useful approach to determine the value of λ. However, similar to linear regression, there exists a formula for computing the loss for LOOCV by just fitting a single model to the entire data set.

$$RSS_{cv}(\lambda) = \sum_{i=1}^{n}(y_i - \hat{g}_\lambda^{(-i)}(x_i))^2 = \sum_{i=1}^{n}\left[\frac{y_i - \hat{g}_\lambda(x_i)}{1 - \{S_\lambda\}_{ii}}\right]^2$$

where $\hat{g}_\lambda^{(-i)}(x_i)$ denotes the estimate at any x_i without using the point i for calculating the loss (leave one out), while $\hat{g}_\lambda(x_i)$ denotes the estimate at x_i using all of the data.

1.6 Local Regression

As the name suggests, we fit regression models in the neighborhood of the point for which we want to make a prediction. But the difference from a normal linear regression is that we give weights to the points such that the points closest get higher weights and points further away

get zero.

Suppose we want to make the predicton at the point x_0. The algorithm is as follows

1. gather the fraction $s = k/n$ points closest to x_0 where s is also called the span an acts like a regularizer. Higher s will make a global fit and vice versa.

2. Assign a weight $Ki0 = K(x_i, x_0)$ to all the points in the neighborhood of x_0 such that the point closest gets the highest weight while the one furthest away gets the weight zero. All the other points in the data get the weight zero.

3. Minimize the error given by

$$RSS = \sum_{i=1}^{n} K(x_i, x_0)(y_i - \beta_0 - \beta_1 x_i)^2$$

   to estimate the values of β_0 and β_1. Note that K is a function known beforehand and thus constant.

4. Make the prediction as $\hat{y}_0 = \beta_0 + \beta_1 x_0$

These models are very useful for adapting to changes in the neighborhood of a given point. The distance function can take several forms like constant, linear, quadratic or even normal distribution.
The approach can be extended to multiple dimensions as well by defining corresponding distances. However, the approach performs **poorly in dimensions higher than** 3 **or** 4 as it becomes difficult to find neighbors in the surrounding area (dimensionality curse).

1.7 Generalized Additive Models

The approches discussed above are extensions of the linear regression model for a single predictor by introducing more flexbility into the models. This idea can be extended for p predictors in the framework of *Generalized Additive Models*. These are applicable for both classification and regression.

$$\hat{y}_i = \beta_0 + \sum_{j=1}^{p} \beta_j x_{ij} \qquad \text{Linear Rgerssion}$$

$$\hat{y}_i = \beta_0 + \sum_{j=1}^{p} f_j(x_{ij}) \qquad \text{Generalized Additive Models}$$

where $f_i(x)$ are non-linear smooth functions applied to each of the predictors separately.

Each of the functions above can be fit using all of the sections defined above. For example, some of the predictors can be smooth splines, some can be just dummy variables, and some can be polynomials. Thus, we expand from p predictors to a multitude of them, where we can choose a different expansion method for each of them individually. We have combined individual simple linear regression models into a general framework for p predictors.
Fitting smoothing splines is not trivial as the loss function is not simple least squares. However, softwares can still fit using *partial residuals*. We repeatedly update the fit for a single predictor, keeping the others constant.
Summarizing,

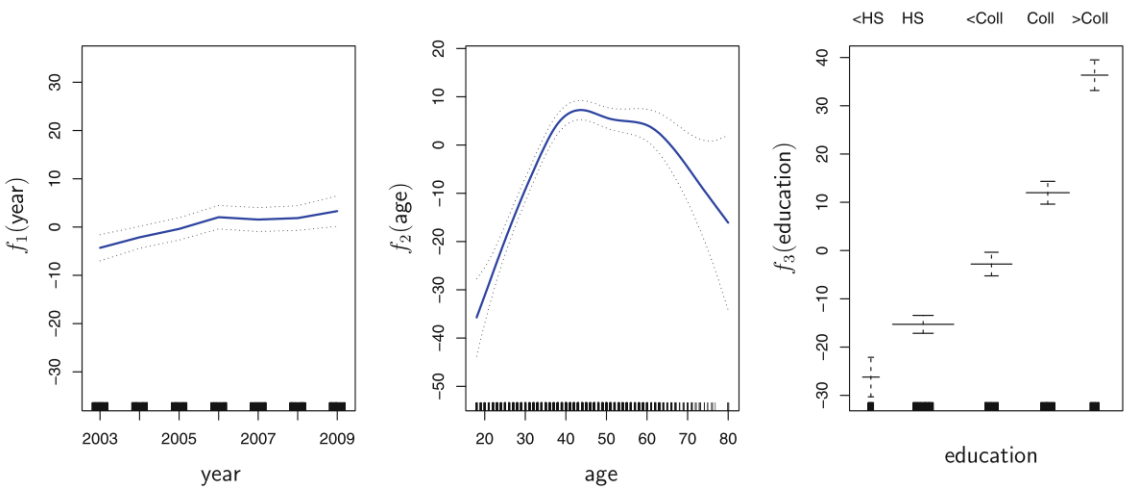

Figure 1: Separate non-linear functions for three different variables. y-axis is the response. Left two plots are amoothing splines with different degrees of freedom. Right plot is dummy variables.

- GAMs allow us to fit a non-linear f_j to each X_j, so that we can automatically model non-linear relationships that standard linear regression will miss. This means that we do not need to manually try out many different transformations on each variable individually.
- The non-linear fits can potentially make more accurate predictions for the response Y .
- Because the model is additive, we can still examine the effect of each X_j on Y individually while holding all of the other variables fixed. Hence if we are interested in inference, GAMs provide a useful representation.
- The smoothness of the function fj for the variable Xj can be summarized via degrees of freedom.
- The main limitation of GAMs is that the model is restricted to be additive. With many variables, important interactions can be missed. However, as with linear regression, we can manually add interaction terms to the GAM model by including additional predictors of the form $X_j \times X_k$. In addition we can add low-dimensional interaction functions of the form $f_{jk}(X_j, X_k)$ into the model; such terms can be fit using two-dimensional smoothers such as local regression, or two-dimensional splines.