Detailed review on GAM Smoothers

The building blocks of GAM is to estimate the functional effects for individual covariates. These functions are usually assumed to be smooth with derivatives. The following estimators are most commonly adopted in the literature.

- **Bin smoother :** This is also known as regressogram in which we partition the predictor into disjoint regions and average the response variable in each region. For a response variable Y, let a single predictor x is partitioned with cutpoints at $c_0 < c_1 < \ldots c_K$ such that $c_0 = -\infty$ and $c_K = \infty$. Indices of the data-points in each region are defined as

  $$R_k = \{i : c_k \leq x_i < c_{k+1}\} \qquad k = 0, 1, \ldots, K-1$$

  Then the smooth function is estimated as $\hat{f}(x_0) = ave_{i \in R_k} Y_i$ for $x_0 \in [c_k, c_{k+1})$. This is a simplest form of smoother however the estimate is not smooth due to sudden jump at the cut-points.

- **Moving averages or Running mean smoother :** This smoother averages the response variable in the neighbourhood of target predictor value e.g. x_0. The neighbourhood can be symmetric (equal number of points on left and right sides of target x_0) or Asymmetric (Just k number of points near x_0 regardless of which side they are). The level of smoothness is defined by the width of the neighbourhood. Let's say $N^S(x_0)$ denotes the indices of the data points in a neighbourhood of x_0 then $\hat{f}(x_0) = ave_{i \in N^S(x_0)} Y_i$. This smoother is popular due to its simplicity but it hardly gives a smooth curve due to lot of wiggliness. The running mean is not a satisfactory smoother because it creates large biases at the end points and doesn't give a straight line when the data is in the straight line.

- **Running line smoother :** The bias problem of running mean smoother is resolved by a simple generalization - Running line smoother. We compute least square line instead of a mean in each neighbourhood. Running line smoother is defined by

  $$f(x_0) = \alpha(x_0) + \beta(x_0) x_0$$

  where $\alpha(x_0)$ and $\beta(x_0)$ are the regression parameters for the neighbourhood $N^S(x_0)$ of x_0. If we consider $N^S(x_0)$ as a symmetric neighbourhood

around x_0 with k number of points on the left and k number of points on the right then the number of points in the neighbourhood $N^S(x_0)$ is equal to $2k+1$. Parameter k controls the smoothness of running line smoothers. Large values of k produce a more smooth curves while small values produce jagged curves. For convenience, we often consider proportion of points in the neighbourhood or 'span' as a parameter denoted by $w = (2k+1)/n$ where n is total number of observations. Running line smoother captures the trend in data nicely but produces jagged fit.

- **Locally Estimated Scatterplot smoother (loess) :** Running line smoothers can be improved by adding more smoothing i.e. instead of only least square fits, we can use the weighted least square fits in the neighbourhoods. In running line smoother, we give equal weights to the data points in the neighbourhood and zero weights to the points outside the neighbourhood. To avoid these discrete change, we give higher weights to the data points near the target x_0 and it gradually reduces as we go far from x_0 in the neighbourhood. This method is also called as locally weighted running line smoother (loess).

- **Kernel smoother :** Kernels are used in this smoother as weights such that weight reduces as we move away from the predictor target value. The weight given to the predictor value x for producing the estimate at target value x_0 is

$$K_h(x_0, x) = \frac{1}{h} K\left(\frac{||x_0, x||}{h}\right)$$

  Here h is a bandwidth and needs to be estimated by using cross validation for model fitting. There are different types of kernel functions $K(.)$ such as Guassian kernel (standard normal density), Epanechnikov kernel etc.

$$\text{Epanechnikov Kernel :} \quad K(t) = \frac{3}{4}(1-t^2) I(|t| \leq 1)$$

  Kernel function $K(.)$ is an even and decreasing function in $|t|$. A typical way to estimate $f(x)$ with kernel smoother is to use locally weighted running line smoother with kernels as weights. Typically, we also use Nadaraya-Watson estimator as

$$\hat{f}(x_0) = \frac{\sum_{i=1}^n K_h(x_0, x_i) Y_i}{\sum_{i=1}^n K_h(x_0, x_i)}$$

- **Regression Spline :** The smoothers discussed so far are explicitly local in nature. Regression splines are nothing but peicewise polynomials. The regions are separated by knots $\zeta_1, \zeta_2, \ldots, \zeta_k$. The piecewise polynomials join smoothly at the knots. Piecewise polynomial can be expressed as a linear combination of a finite set of basis functions that do not depend

on the response variable Y. The function $f(.)$ is assumed to be a spline function as follows :

$$f(x) = \sum_{j=1}^{k+d+1} \beta_j N_j(x)$$

where $N_j(.)$ are the basis functions and β_k are the associated coefficients. With k knots there are $k+1$ polynomials of degree d along with $d.k$ constraints, leading to $(d+1)(k+1)-d.k = d+k+1$ parameters. Following are some most popular basis functions :

Truncated power series basis : A simple choice of basis functions for the peicewise spline is truncated power series basis which is given as

$$N_j(x) = x^{j-1} \quad j = 1, 2, ..., d+1$$
$$N_{j+d+1}(x) = (x - \zeta_j)_+^d \quad j = 1, \ldots, k$$

For cubic spline $d = 3$. Hence, we get $k+4$ basis functions. where a_+ is the positive part of a. This spline has $k+4$ basis functions. It has two continuous derivatives and the third derivative jumps at the knots as a step function. Even though algebraically this is good basis but it is expensive.

B-spline basis : A B-spline basis is built with augmented knot sequence,

$$\tau_1 \leq \ldots \tau_d \leq \tau_{d+1} \leq \tau_{d+2} \leq \ldots \tau_{d+k+1} \leq \tau_{d+k+2} \leq \tau_{d+k+3} \leq \cdots \leq \tau_{2d+k+2}$$

where $\tau_{d+2} = \zeta_1, \ldots, \tau_{d+k+1} = \zeta_k$ are the inner knots while $\tau_{d+1} = \zeta_0$ and $\tau_{d+k+2} = \zeta_{k+1}$ are boundary knots. The choice of additional knots $\tau_1, \ldots, \tau_d$ and $\tau_{d+k+3}, \ldots, \tau_{2d+k+2}$ is arbitrary. A common strategy is to set them equal to the boundary knots. B-spline basis can be built by starting with a set of Haar basis functions,

$$B_{i,0}(x) = \begin{cases} 1 & \text{if } \tau_i < x < \tau_{i+1} \\ 0 & \text{otherwise} \end{cases}$$

and then applying a simple linear recursion relationship d times, yielding the $K+d+1$ needed basis functions as

$$B_{i,d}(x) = \frac{x - \tau_i}{\zeta_{i+d} - \tau_i} B_{i,d-1}(x) + \frac{\tau_{i+d+1} - x}{\tau_{i+d+1} - \tau_{i+1}} B_{i+1,d-1}(x) \quad i = 1, 2, \ldots, k+d+1$$

This is a B-spline basis function of order d. If $d = 0$, this basis function is just a step function while $\{B_{i,3}; i = 1, 2, \ldots, k+4\}$ are the cubic B-spline basis functions. The important feature of B-spline function is that it is non-zero on a few adjacent knots span, more precisely, basis $B_{i,d}(x)$ is non-zero on $d+1$ knot-spans or $d+2$ knots. B-spline is computationally efficient than truncated power series basis because it does not require evaluating powers.

Natural cubic spline : An important variant of cubic spline is natural cubic spline. They are the cubic splines that have an additional constraint that the function is linear beyond the boundary knots i.e. we impose two additional constraints $f''' = f'' = 0$ in boundary regions.

Choosing the effective number of knots and their position is one of the biggest drawbacks of regression splines. Also there are more than one smoothing parameters to vary to attain smoothness.

- **Smoothing Splines :** In smoothing splines, instead of using knots, it minimizes the penalized log likelihood in generalized additive models. Here, the penalisation term handles the roughness of the fit. For instance, for a single predictor x and binary response Y with the logit GAM model $\log\left(\frac{p}{1-p}\right) = f(x)$ such that $p = \Pr(Y = 1|x)$, the function $f(.)$ is fitted by minimizing

$$\sum_{i=1}^{n}[Y_i \log(p_i) + (1 - Y_i) \log(1 - p_i)] + \frac{\lambda}{2} \int (f''(x))^2 dx$$

where the first term is log likelihood function and second term is penalty term. λ is a non-negative smoothing parameter. $\int (f''(x))^2 dx$ measures the wiggliness of the function f. It turns out that the function that minimizes the penalized sum of squares is a natural cubic spline with knots at every data point.