Moving Beyond Linearity

Yuan Yao

Department of Mathematics
Hong Kong University of Science and Technology

Most of the materials here are from Chapter 7 of Introduction to Statistical Learning by Gareth James, Daniela Witten, Trevor Hastie and Robert Tibshirani.

Fall, 2024

Polynomial regression

Step functions

Regression splines

Smoothing spline

Local regression

Generalized additive model

Recall: dimension reduction methods (using derived inputs)

- When p is large, we may consider to regress on, not the original inputs x, but some small number of derived features $\phi_1, ..., \phi_k$ with $k < p$.

$$y_i = \theta_0 + \sum_{j=1}^{k} \theta_j \phi_j(x_i) + \epsilon_i, \quad i = 1, ..., n.$$

  - ϕ_j can be linear: linear combinations of $X_1, \ldots, X_p$
  - ϕ_j can be nonlinear: basis, kernels, neural networks, trees, etc.

Principal Component Regression (PCR)

For $X = (X_1, \ldots, X_p)$,

- Define ϕ_j be the projection on the j-th eigenvector of centralized data covariance $\hat{\Sigma} = U \Lambda U^T$:

$$Z_j = \phi_j(X) = u_j^T (X - \hat{\mu})$$

- Principal Component Regression (PCR) model:

$$y_i = \theta_0 + \sum_{j=1}^{k} \theta_j Z_j + \epsilon_i, \;\; i = 1, ..., n.$$

Recall: Principal Component Analysis (PCA)

- Suppose there are n observations of p variables presented as $\mathbf{X} = (\mathbf{x}_1, \ldots \mathbf{x}_n)^T \in \mathbf{R}^{n \times p}$, where $\mathbf{x}_i^T \in \mathbf{R}^p$.
- Define the sample covariance matrix

$$\hat{\Sigma} = \frac{1}{n} \sum_{i=1}^{n} (\mathbf{x}_i - \hat{\mu})^T (\mathbf{x}_i - \hat{\mu})$$

  where the sample mean $\hat{\mu} = \frac{1}{n} \sum_i \mathbf{x}_i$.
- $\hat{\Sigma}$ has an eigenvalue decomposition

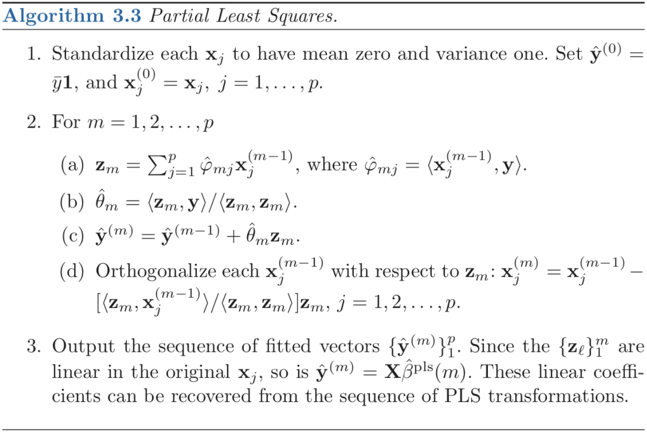
$$\hat{\Sigma} = U \Lambda U^T,$$

  with $U^T U = I_p$ ($U = [u_1, \ldots, u_p]$), $\Lambda = \mathbf{diag}(\lambda_1, \ldots, \lambda_p)$, $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_p \geq 0$.

Partial Least Square (PLS)

- In PCR, derived inputs ϕ (Z_j) only depends on inputs X, independent to y.
- PLS, exploits linear combinations of both X and y as derived inputs.
- Assume that $\mathbf{x}_j$'s are standardized with zero mean and unit variance. The PLS procedure is:
  - First, let $\hat{\phi}_{1j} = \langle \mathbf{x}_j, y \rangle$ and $Z_1 = \sum_j \hat{\phi}_{1j}\mathbf{x}_j$, the first PLS direction;
  - Regressing y on z_1 gives $\hat{\theta}_1$;
  - Orthogonalizing $\mathbf{x}_j$ with respect to z_1 and repeating the procedure above.

Partial Least Square (PLS)



Algorithm 3.3 *Partial Least Squares.*

```
1. Standardize each $\mathbf{x}_j$ to have mean zero and variance one. Set $\hat{\mathbf{y}}^{(0)} = \bar{y}\mathbf{1}$, and $\mathbf{x}_j^{(0)} = \mathbf{x}_j,\ j = 1, \dots, p$.

2. For $m = 1, 2, \dots, p$

    (a) $\mathbf{z}_m = \sum_{j=1}^{p} \hat{\varphi}_{mj}\mathbf{x}_j^{(m-1)}$, where $\hat{\varphi}_{mj} = \langle \mathbf{x}_j^{(m-1)}, \mathbf{y} \rangle$.
    (b) $\hat{\theta}_m = \langle \mathbf{z}_m, \mathbf{y} \rangle / \langle \mathbf{z}_m, \mathbf{z}_m \rangle$.
    (c) $\hat{\mathbf{y}}^{(m)} = \hat{\mathbf{y}}^{(m-1)} + \hat{\theta}_m \mathbf{z}_m$.
    (d) Orthogonalize each $\mathbf{x}_j^{(m-1)}$ with respect to $\mathbf{z}_m$: $\mathbf{x}_j^{(m)} = \mathbf{x}_j^{(m-1)} - [\langle \mathbf{z}_m, \mathbf{x}_j^{(m-1)} \rangle / \langle \mathbf{z}_m, \mathbf{z}_m \rangle]\mathbf{z}_m,\ j = 1, 2, \dots, p$.

3. Output the sequence of fitted vectors $\{\hat{\mathbf{y}}^{(m)}\}_1^p$. Since the $\{\mathbf{z}_\ell\}_1^m$ are linear in the original $\mathbf{x}_j$, so is $\hat{\mathbf{y}}^{(m)} = \mathbf{X}\hat{\beta}^{\text{pls}}(m)$. These linear coefficients can be recovered from the sequence of PLS transformations.
```

Figure: Partial Least Square Algorithm (Alg. 3.3 in ELS)

About this chapter

- Linear model is the most fundamental statistical model.
- Its limitation is the mean response must be a linear function of inputs/covariates.
- This relation in practice often does not hold.
- Nonlinear models are needed

The nonlinear models.

- Polynomial regression.
- Step functions
- Regression splines
- Smoothing splines
- Local regression
- Generalized additive models.
- Trees, SVM, neural nets, ...

Outline

Polynomial regression

Step functions

Regression splines

Smoothing spline

Local regression

Generalized additive model

Some general remark

- Rather than directly using inputs, we use polynomials, or step functions, of the inputs as the “derived inputs”, in linear regression.
- The approach can be viewed as derived inputs approach.
- More generally, the basis function approach.
- Starting from now, we only consider one input, for simplicity of illustration.

- Data: $(y_i, x_i), i = 1, ..., n$.
- The general model

$$y_i = f(x_i) + \epsilon_i$$

- We can safely assume $f(\cdot)$ to be continuous:
  - **Stability**: Any small change of inputs must have small influence on outputs
- Cannot search for arbitrary function $f(\cdot)$.
- Limit the search space.
- Continuous functions? (still infinite dimension but can be approximated.
- by Polynomial functions, or step functions, or certain basis functions, ...

- Linear model (restricting $f(\cdot)$ to be linear) :

$$y_i = \beta_0 + \beta_1 x_i + \epsilon_i$$

- Polynomial regression model (restricting $f(\cdot)$ to be polynomial of degree p):

$$y_i = \beta_0 + \beta_1 x_i + \beta_2 x_i^2 + ... + \beta_p x_i^p + \epsilon_i$$

- This is a multiple linear regression model with p inputs: $(x_i, x_i^2, ..., x_i^p)$.
- All linear regression results apply.
- Problem: how to determine the appropriate degree p.
- *Drawback: difficult to fit locally highly varying functions.*

The generalized linear model

- **Generalized linear model**:

$$E(Y|X) = g(X^T\beta)$$

  where g is a given *link function*

- Examples:
  1. linear regression: $g(x) = x$
  2. logistic regression: $g(x) = 1/(1 + e^{-x})$, the sigmoid function. $Y = 1$ or 0.
  3. Probit model: $g(x) = \Phi(x)$, the cdf of $N(0, 1)$. $Y = 1$ or 0.
  4. Poisson model: $g(x) = e^x$. Y is count data.
  5. ...

- They can be extended to generalized non-linear model in the same fashion.

logistic model with polynomial regression

- For binary response y_i, coded the binary events as 1 and 0.

$$p(y_i = 1|x_i) = \frac{\exp(\beta_0 + \beta_1 x_i + ... + \beta_p x_i^p)}{1 + \exp(\beta_0 + \beta_1 x_i + ... + \beta_p x_i^p)}$$

- This is essentially just logistic model with p inputs.
- All results on logistic model apply here.

Example: Wage dataset

- income (wage) survey information for males from the central Atlantic region of the United States
- $p = 12$ variables, e.g. year, age, education, wages, logwages, etc. Hybrid numeric and categorical variables.
- $n = 3000$ samples
- Consider numeric y as wages, or binary y of indicator: wages$> 250,000$

Degree–4 Polynomial

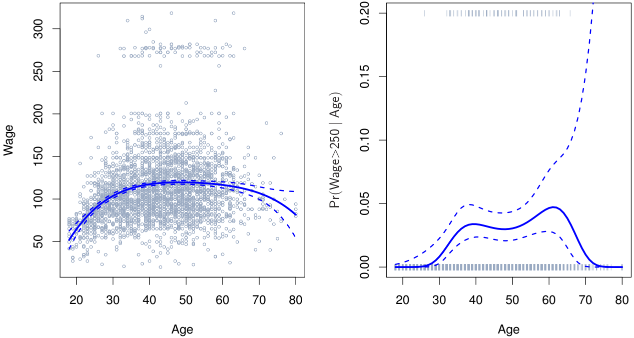


Figure: 7.1. The Wage data. Left: The solid blue curve is a degree-4 polynomial of year (in thousands of dollars) as a function of age, fit by least squares. The dotted curves indicate an estimated 95% confidence interval. Right: We model the binary event wage> 250 using logistic regression, again with a degree-4 polynomial. The fitted posterior probability of wage exceeding $250,000 is shown in blue, along with an estimated 95% confidence interval.

Outline

Polynomial regression

Step functions

Regression splines

Smoothing spline

Local regression

Generalized additive model

Step functions (piecewise constant functions)

- Step functions are piece-wise constants.
- Continuous functions can be well approximated by step functions using linear combinations.
- Create the cutpoints

$$-\infty = c_0 < c_1 < \dots < c_p < c_{p+1} = \infty$$

- The entire real line is cut into $p + 1$ intervals.
- Set $c_k(x) = I(c_k \leq x < c_{k+1})$, for $k = 0, \dots, p$.
- Use linear combination of $c_k(x)$ to approximate functions.

Regression model based on step functions

- Model:

$$y_i = \beta_0 + \beta_1 c_1(x_i) + ... + \beta_p c_p(x_i) + \epsilon_i.$$

- Again a multiple linear regression model.
- Same extension works for generalized linear model.
- Difficulty in creating the number and locations of cutpoints
- Drawback: non-smooth, not even continuous.

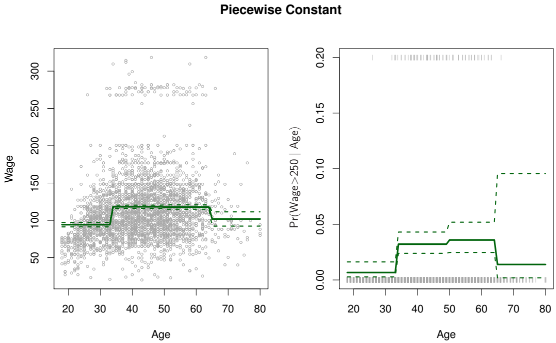


Figure: 7.2. The Wage data. Left: The solid curve displays the fitted value from a least squares regression of wage (in thousands of dollars) using step functions of age. The dotted curves indicate an estimated 95% confidence interval. Right: We model the binary event wage> 250 using logistic regression, again using step functions of age. The fitted posterior probability of wage exceeding $250,000 is shown, along with an estimated 95% confidence interval.

Basis functions

- In general, let $b_1(x), \ldots, b_p(x)$ be a set of *basis functions.*
- We limit the search space of $f(\cdot)$ to the space that is linearly spanned by these basis functions:

$$\{g(x) : g(x) = a_0 + \sum_{j=1}^{p} a_i b_i(x)\}$$

- The model is

$$y_i = \beta_0 + \beta_1 b_1(x_i) + \ldots + \beta_p b_p(x_i) + \epsilon_i.$$

- Again a multiple linear regression model.
- The polynomial functions or step functions are special cases of basis functions approach.
- Other choices: wavelet functions or Fourier series or regression splines.

Outline

Polynomial regression

Step functions

Regression splines

Smoothing spline

Local regression

Generalized additive model

Piecewise polynomial functions

- A hybrid of step function approach and polynomial function approach.
- Cut the entire real line (or the range of values of covariates) into sub-intervals same as step function approach.
- These cut-points are called knots.
- Use a polynomial function on each sub-interval.
- Still a multiple linear regression model.
- Step function approach is a special case of piecewise polynomial of degree 0.
- Advantage: capture local variation; the degree of polynomial is generally low.
- disadvantage: dis-continuity at knots.

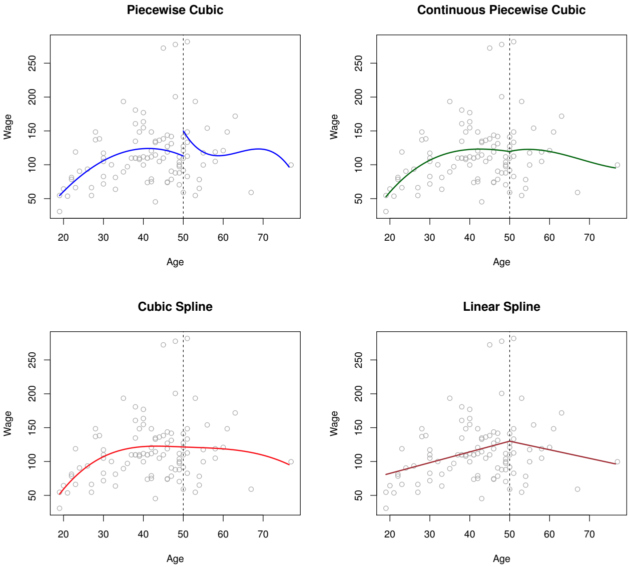

Piecewise Cubic
Continuous Piecewise Cubic
Cubic Spline
Linear Spline
Wage
Age
50
100
150
200
250
20
30
40
50
60
70

Figure 7.3. (Figure of previous page) Various piecewise polynomials are fit to a subset of the Wage data, with a knot at age= 50. Top Left: The cubic polynomials are unconstrained. Top Right: The cubic polynomials are constrained to be continuous at age= 50. Bottom Left: The cubic polynomials are constrained to be continuous, and to have continuous first and second derivatives. Bottom Right: A linear spline is shown, which is constrained to be continuous

Constraining the piecewise polynomial

- When fit the least squares, one can add constraints to the least squares minimization
- The constraints can be such that the piecewise polynomial is forced to be continuous at knots.
- The constraints can be stronger such that the piecewise polynomial is forced to be differentiable at knots with continuous first derivatives.
- The constraints can be stronger such that the piecewise polynomial is forced to be differentiable at knots with continuous second derivatives.
- ...

The effect of constraints

- Each constraint can be expressed as a linear equation.
- It reduces one degree of freedom.
- And reduces the complexity of the model.

Spline functions

- Spline functions of degree d are piecewise polynomial functions of degree d but have continuous derivatives up to order $d-1$ at knots.
- Cubic spline: piecewise cubic polynomials but are continuous and have continuous 1st and second derivatives at knots.
- The degree of freedom of a cubic spline with K knots is:

$$4 \times (K+1) - 3K = K + 4.$$

  Totally $K+1$ cubic functions, each has 4 free parameters, but each of the K knot has 3 constraints on continuity, continuity of 1st and 2nd derivatives.

Spline basis representation

- Suppose the K knots $\xi_1 < ... < \xi_K$ are determined.
- We may find $1, b_1(x), ..., b_{K+3}$ to form the space of cubic splines with knots at $\xi_1, ..., \xi_K$.
- Then the spline regression model is

$$y_i = \beta_0 + \beta_1 b_1(x_i) + ... + \beta_{K+3} b_{K+3}(x_i) + \epsilon_i$$

- *How to find these basis functions* $b_k(x)$?
- Each must be a polynomial of order 3 and must be continuous, continuous at 1st and 2nd derivates at all knots.

Spline basis representation

- x , x^2 and x^3 satisfy the requirement.
- Let

$$h(x,\xi) = (x-\xi)_+^3 = \begin{cases} (x-\xi)^3 & \text{if } x > \xi \\ 0 & \text{otherwise} \end{cases}$$

- $h(x,\xi_k)$ also satisfy the requirement.
- The basis functions of **cubic splines** can be

$$1, x, x^2, x^3, h(x,\xi_1), ..., h(x,\xi_K)$$

- Totally $K+4$ dimension with $K+3$ features.

Natural spline

- The behavior of the cubic spline at boundary can be quite unstable.
- **Natural spline** is cubic spline but require the function to be linear on $(-\infty, \xi_1]$ and $[\xi_K, \infty)$.
- With further restriction near boundary, natural spline regression generally behaves better than cubic spline regression.

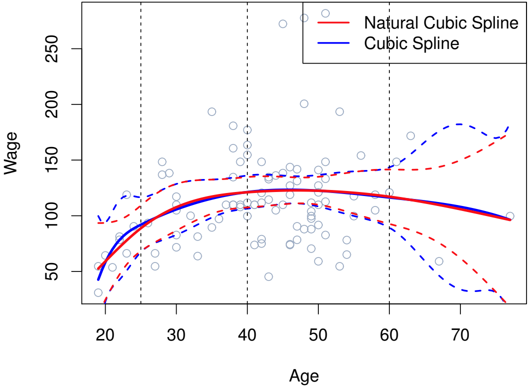


Figure: 7.4. A cubic spline and a natural cubic spline, with three knots, fit to a subset of the Wage data. Natural spline has narrower confidence intervals near boundary

Degree of Freedom

- Degree of freedom of natural spline with K knots is $K + 4 - 4 = K$ as only 2 parameters on 2 boundary intervals, but excluding the constant (absorbed in intercept), we usually call it $K - 1$ **degree of freedom**.
- Example: natural cubic splines has $4 = K - 1$ degree of freedom corresponds to $K = 5$ knots and $K - 2 = 3$ interior knots.

Natural spline basis representation

- K knots $\xi_1, \ldots, \xi_K$, and K basis (as DF is $K + 4 - 4 = K$).
- Recursive construction of **Natural spline basis**:

$$N_1(x) = 1, N_2(x) = x, N_{k+2} = d_k(x) - d_{K-1}(x)$$

  where

$$d_k(x) = \frac{(x - \xi_k)^3_+ - (x - \xi_K)^3_+}{\xi_K - \xi_k}$$

- Regression function is

$$f(x) = \sum_{j=1}^{K} \beta_j N_j(x)$$

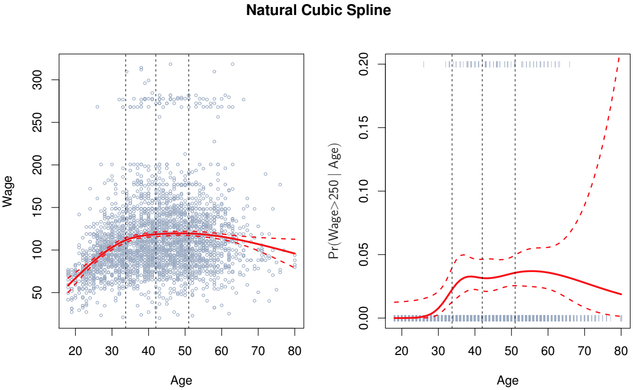


Figure: 7.5. A natural cubic spline function with four degrees of freedom is fit to the Wage data. Left: A spline is fit to wage (in thousands of dollars) as a function of age. Right: Logistic regression is used to model the binary event wage> 250 as a function of age. The fitted posterior probability of wage exceeding $250,000 is shown.

Choice of number and locations of knots

- Usually choose equally spaced knots within the range of values of inputs.
- If we know a function is highly varying somewhere, place more knots there, so that the spline function is also highly varying in the area.
- Try several choices of the number of knots, and use validation/cross-validation approach to determine the best.
- Many statistics software provide automatic choice of number and location of knots.

Example: Wage data

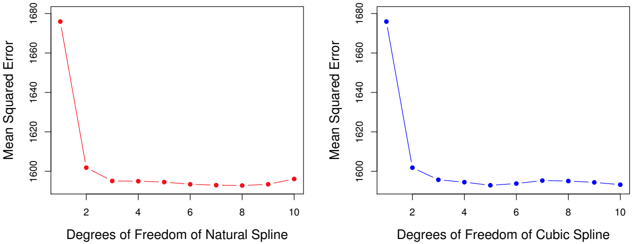


Figure: Ten-fold cross-validated mean squared errors for selecting the degrees of freedom when fitting splines to the Wage data. The response is wage and the predictor age. Left: A natural cubic spline. Right: A cubic spline. It seems that three degrees of freedom for the natural spline and four degrees of freedom for the cubic spline are quite adequate

Comparison with Polynomial Regression

- Regression splines often give superior results to polynomial regression.
- Splines introduce flexibility by increasing the number of knots but keeping the degree fixed.
- Polynomial increase model flexibility by increased order of power function, which can be dangerously inapproximate for moderately large or small X in absolute value.
- Polynomial function has poor boundary behavior.
- Natural spline is much better.

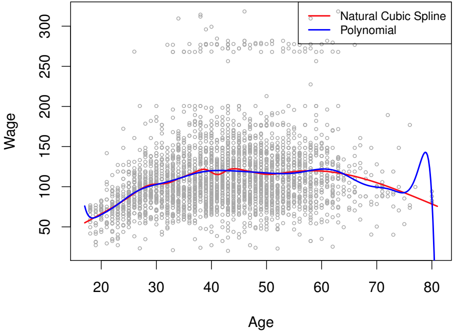


Figure: 7.7. On the Wage data set, a natural cubic spline with 15 degrees of freedom is compared to a degree-15 polynomial. Polynomials can show wild behaviour, especially near the tails.

Outline

Polynomial regression

Step functions

Regression splines

Smoothing spline

Local regression

Generalized additive model

Data Adaptive Choice of knots

- It remains an art to choose the number and locations of knots in **Natural Splines**.
  - Equally spaced knots within the range of values of inputs, or
  - More knots in highly varying regions.
- **Smoothing Spline** will use a *data adaptive choice of knots*:
  - Every sample will be a knot
  - Knot selection is equivalent to sample selection
  - It constructs *reproducing kernel Hilbert spaces* [Spline Models for Observational Data, by Grace Wahba]

Smoothing spline

- With to minimize
$$\sum_{i=1}^{n}(y_i - f(x_i))^2$$
subject to certain smoothness constraints on $f(\cdot)$.
- The most common constraint is $\ddot{f}$, the second derivative do not vary much.
- A natural choice is: minimizng

$$\sum_{i=1}^{n}(y_i - f(x_i))^2 \quad \text{subject to} \int \ddot{f}(x)^2 dx < s$$

Smoothing spline

- This is equivalent to

$$\sum_{i=1}^{n}(y_i - f(x_i))^2 + \lambda \int \ddot{f}(x)^2 dx \tag{7.11}$$

  where λ is the tuning parameter.

- The first term is loss; the second term is roughness penalty.
- The function f minimizing the above is called *smoothing spline.*
- The function that minimize that loss+roughness penalty is a **natural cubic spline** with data adaptive **knots** $x_1, ..., x_n$ (Exercise!).

Smoothing spline representation

- n knots $x_1, ..., x_n$ and n **Natural spline basis** N_j.
- Regression function: $f(x) = \sum_{j=1}^{n} \beta_j N_j$.
- Penalized least square

$$\sum_{i=1}^{n} (y_i - f(x_i))^2 + \lambda \int \ddot{f}(x)^2 dx = \|\mathbf{y} - \mathbf{N}\beta\|^2 + \lambda \beta^T \Omega_n \beta.$$

  where $[\Omega_n]_{jk} = \int \ddot{N}_j(t) \ddot{N}_k(t) dt$

- Solution is

$$\hat{\beta}_\lambda = (\mathbf{N}^T \mathbf{N} + \lambda \Omega_n)^{-1} \mathbf{N}^T \mathbf{y}$$

- Prediction

$$\hat{\mathbf{y}} = \mathbf{N}\hat{\beta}_\lambda = \mathbf{N}(\mathbf{N}^T \mathbf{N} + \lambda \Omega_n)^{-1} \mathbf{N}^T \mathbf{y} =: \mathbf{S}_\lambda \mathbf{y}$$

The tuning parameter

- λ controls the amount of roughness penalty
- $\lambda = 0$: no penalty, degree of freedom $= n$; overfit.

$$\hat{f}(x_i) = y_i$$

- $\lambda = \infty$: infinity penalty; f must be linear, degree of freedom $= 2$.

$$\hat{f}(x) = \hat{\beta}_0 + \hat{\beta}_1 x_i, \quad \text{the least squares estimate}$$

- What the degree of freedom when $\lambda > 0$ and is finite?
- We call it *effective degree of freedom*, denoted as df_λ.

Effective degree of freedom

- The df_λ is a measure of the flexibility of the smoothing splinethe higher it is, the more flexible (and the lower-bias but higher-variance) the smoothing spline
- Minimizing (7.11), let the fitted values be

$$\hat{\mathbf{y}} = \mathbf{S}_\lambda \mathbf{y} \tag{7.12}$$

  where $\hat{\mathbf{y}} = (y_1, ..., y_n)^T$ is an n-vector, representing the fitted values at $x_1, ..., x_n$; and the sensitivity matrix $\mathbf{S}_\lambda$ is an $n \times n$ matrix, depending only on covariates.
- (Reproducing Kernel Hilbert Spaces, Wahba 1990) It can be shown that the fitted values are linear functions of $\mathbf{y}$.
- Then, the effective degree of freedom is

$$df_\lambda = trace(\mathbf{S}_\lambda)$$

Choice of λ

- By cross validation.
- For leave-one-out cross-validation (LOOCV), it can be shown

$$\mathrm{RSS}_{cv}(\lambda) = \sum_{i=1}^{n} (y_i - \hat{f}_\lambda^{(-i)}(x_i))^2 = \sum_{i=1}^{n} \Big[\frac{y_i - \hat{f}_\lambda(x_i)}{1 - s_{\lambda,ii}} \Big]^2$$

  where $s_{\lambda,ii}$ is the i-th diagonal element of $\mathbf{S}_\lambda$.
- One fit does it all!
- Recall that this is the same as linear regression. In fact,

$$\mathbf{S}_\infty = \mathbf{H}$$

  where $\mathbf{H} = \mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T$ is the hat matrix in linear regression.

Fast computation of cross-validation I

- The leave-one-out cross-validation statistic is given by

  $$CV = \frac{1}{N}\sum_{i=1}^{N} e_{[i]}^2,$$

  where $e_{[i]} = y_i - \hat{y}_{[i]}$, the observations are given by $y_1, \dots, y_N$, and $\hat{y}_{[i]}$ is the predicted value obtained when the model is estimated with the ith case deleted.
- Suppose we have a linear regression model $\mathbf{Y} = \mathbf{X}\boldsymbol{\beta} + \mathbf{e}$. The $\hat{\boldsymbol{\beta}} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{Y}$ and $\mathbf{H} = \mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T$ is the hat matrix. It has this name because it is used to compute $\hat{\mathbf{Y}} = \mathbf{X}\hat{\boldsymbol{\beta}} = \mathbf{H}\mathbf{Y}$. If the diagonal values of $\mathbf{H}$ are denoted by $h_1, \dots, h_N$, then the leave-one-out cross-validation statistic can be computed using

  $$CV = \frac{1}{N}\sum_{i=1}^{N}[e_i/(1-h_i)]^2,$$

  where $e_i = y_i - \hat{y}_i$ is predicted value obtained when the model is estimated with all data included.

Fast computation of cross-validation II

Proof

- Let $\mathbf{X}_{[i]}$ and $\mathbf{Y}_{[i]}$ be similar to $\mathbf{X}$ and $\mathbf{Y}$ but with the ith row deleted in each case. Let $\mathbf{x}_i^T$ be the ith row of $\mathbf{X}$ and let

  $$\hat{\boldsymbol{\beta}}_{[i]} = (\mathbf{X}_{[i]}^T \mathbf{X}_{[i]})^{-1} \mathbf{X}_{[i]}^T \mathbf{Y}_{[i]}$$

  be the estimate of $\boldsymbol{\beta}$ without the ith case. Then $e_{[i]} = y_i - \mathbf{x}_i^T \hat{\boldsymbol{\beta}}_{[i]}$.
- Now $\mathbf{X}_{[i]}^T \mathbf{X}_{[i]} = (\mathbf{X}^T \mathbf{X} - \mathbf{x}_i \mathbf{x}_i^T)$ and $\mathbf{x}_i^T (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{x}_i = h_i$. So by the Sherman-Morrison-Woodbury formula,

  $$(\mathbf{X}_{[i]}^T \mathbf{X}_{[i]})^{-1} = (\mathbf{X}^T \mathbf{X})^{-1} + \frac{(\mathbf{X}^T \mathbf{X})^{-1} \mathbf{x}_i \mathbf{x}_i^T (\mathbf{X}^T \mathbf{X})^{-1}}{1 - h_i}.$$

Fast computation of cross-validation III

Proof

- Also note that $\mathbf{X}_{[i]}^T\mathbf{Y}_{[i]} = \mathbf{X}^T\mathbf{Y} - \mathbf{x}y_i$. Therefore

$$\begin{aligned}\hat{\boldsymbol{\beta}}_{[i]} &= \left[(\mathbf{X}^T\mathbf{X})^{-1} + \frac{(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{x}_i\mathbf{x}_i^T(\mathbf{X}^T\mathbf{X})^{-1}}{1-h_i}\right](\mathbf{X}^T\mathbf{Y} - \mathbf{x}_i y_i) \\ &= \hat{\boldsymbol{\beta}} - \left[\frac{(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{x}_i}{1-h_i}\right][y_i(1-h_i) - \mathbf{x}_i^T\hat{\boldsymbol{\beta}} + h_i y_i] \\ &= \hat{\boldsymbol{\beta}} - (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{x}_i e_i/(1-h_i)\end{aligned}$$

- Thus

$$\begin{aligned}e_{[i]} &= y_i - \mathbf{x}_i^T\hat{\boldsymbol{\beta}}_{[i]} \\ &= y_i - \mathbf{x}_i^T\left[\hat{\boldsymbol{\beta}} - (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{x}_i e_i/(1-h_i)\right] \\ &= e_i + h_i e_i/(1-h_i) = e_i/(1-h_i)\end{aligned}$$

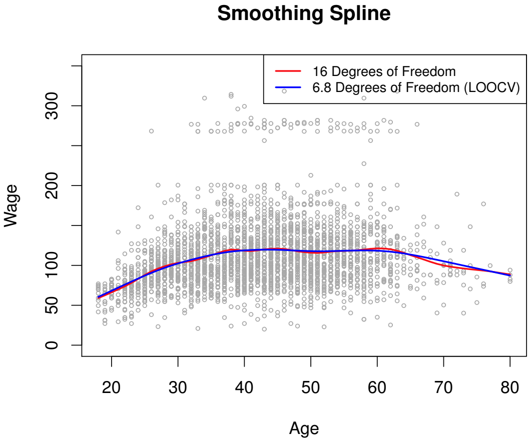


Figure: 7.8. Smoothing spline fits to the Wage data. The red curve results from specifying 16 effective degrees of freedom. For the blue curve, λ was found automatically by leave-one-out cross-validation, which resulted in 6.8 effective degrees of freedom.

Outline

Polynomial regression

Step functions

Regression splines

Smoothing spline

Local regression

Generalized additive model

Local view

- Rather than considering fitting a function f to the data, we just focus on a target point, say x_0, and try to estimate $f(x_0) = \beta_0$.
- Consider a weight function, often called kernel function, $k(t)$ which is nonnegative symmetric and becomes small when $|t|$ is large.

Typical choice of (Nadaraya-Watson) kernels

- Uniform kernel: $k(t) = 1/2I(|t| \leq 1)$.
- Triangle kernel: $k(t) = (1 - |t|)I(|t| \leq 1)$.
- Gaussian kernel: $k(t) = e^{-t^2/2}/\sqrt{2\pi}$
- Epanecknikov kernel: $k(t) = 3/4(1 - t^2)_+$
- Logistic kernel: $k(t) = 1/(e^t + e^{-t} + 2)$.
- Sigmoid kernel: $k(t) = 2/(\pi(e^t + e^{-t}))$.
- Note that this is **NOT reproducing kernel**!

Local view: Nadaraya-Watson Kernel Regression

- Use the kernel function to create *weights* on each observation so that those with x_i closer to x_0 gets more weights:
$$K_{i0} = \frac{1}{h} k(\frac{x_i - x_0}{h})$$
- These weights create the "Localness" surrounding x_0. h is the bandwidth that is usually small.
- We can consider minimization
$$\sum_{i=1}^{n} K_{i0}(y_i - \beta_0 - \beta_1(x_i - x_0))^2$$

  Then, $\hat{\beta}_0$ is the estimator of $f(x_0)$.
- This estimator is local linear estimator, since locally around x_0, we used linear function to approximate $f(x)$.
- One can certainly consider local polynomial estimation, by considering local polynomial approximation.

Remark.

- Local linear estimate is also a linear function of **y**, and there has expression of the form of (7.12).
- The degree of freedom controlled by the bandwidth.
- Small bandwidth results in small bias but high variance (and high effective degree of freedom).
- Can be difficult to implement with high dimension data, by the **curse of dimensionality**.

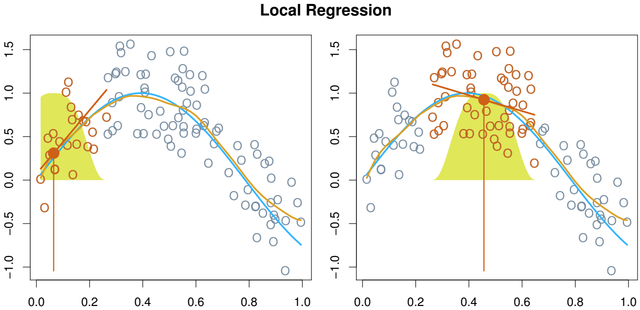


Figure: 7.9. Local regression illustrated on some simulated data, where the blue curve represents $f(x)$ from which the data were generated, and the light orange curve corresponds to the local regression estimate $\hat{f}(x)$. The orange colored points are local to the target point x_0, represented by the orange vertical line. The yellow bell-shape superimposed on the plot indicates weights assigned to each point, decreasing to zero with distance from the target point. The fit $\hat{f}(x_0)$ at x_0 is obtained by fitting a weighted linear regression (orange line segment), and using the fitted value at x_0 (orange solid dot) as the estimate $\hat{f}(x)$.

Outline

Polynomial regression

Step functions

Regression splines

Smoothing spline

Local regression

Generalized additive model

- With p inputs, the general model should be

$$y_i = f(x_{i1}, ..., x_{ip}) + \epsilon_i.$$

- Difficult to model multivariate nonlinear function.
- Restrict search space to

$$\{f(x_1, ..., x_p) : f_1(x_1) + f_2(x_2) + ...f_p(x_p)\}$$

- The multivariate function is simple sum of nonlinear function of each variable.
- This leads to the **generalized additive model (GAM)**.

The GAM

- The model:

$$y_i = f_1(x_{i1}) + f_2(x_{i2}) ... + f_p(x_{ip}) + \epsilon_i$$

- The statistical estimation of f_1, ..., f_p can be solved by taking advantage of
- 1. the methodologies for nonlinear model for single input case.
- 2. a backfit algorithm.

The backfitting algorithm

- Initialize the estimator of f_1, ..., f_p, denoted as $\hat{f}_1, ..., \hat{f}_p$.
- Given estimates $\hat{f}_1, .., \hat{f}_{k-1}$, $\hat{f}_{k+1}$, ..., $\hat{f}_p$, compute

$$\tilde{y}_i = y_i - \hat{f}_1(x_{i1}) - \hat{f}_{k-1}(x_{i,k-1}) - \hat{f}_{k+1}(x_{i,k+1}) - ... - \hat{f}_p(x_{ip})$$

- Run nonlinear regression with response $\tilde{y}_i$ and single input x_{ik}, to obtain the estimate of f_k. Update $\hat{f}_k$ by this estimate.
- Continue with the update of f_{k+1}. (If $k = p$ continue the update of f_1.)
- Repeat till convergence.

Example: Wage data

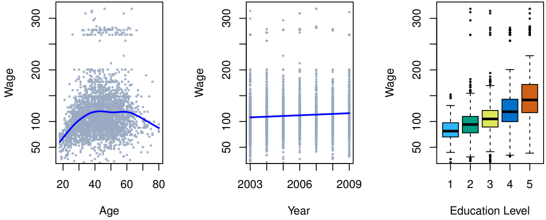


Figure: 1.1. Left: wage as a function of age. On average, wage increases with age until about 60 years of age, at which point it begins to decline. Center: wage as a function of year. There is a slow but steady increase of approximately $10,000$ in the average wage between 2003 and 2009. Right: Boxplots displaying wage as a function of education, with 1 indicating the lowest level (no high school diploma) and 5 the highest level (an advanced graduate degree).

Example: Wage data

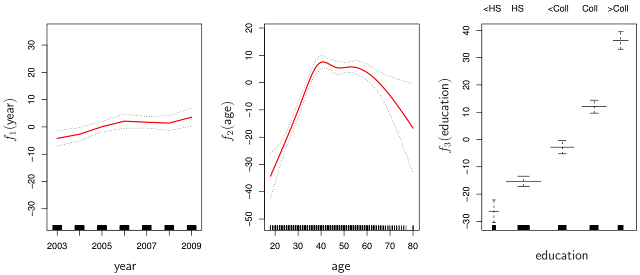


Figure: 7.11. For the Wage data, plots of the relationship between each feature and the response, wage, in the fitted model (7.16). Each plot displays the fitted function and pointwise standard errors. The first two functions are **natural splines** in year and age, with four and five degrees of freedom, respectively. The third function is a **step** function, fit to the qualitative variable education.

Example: Wage data

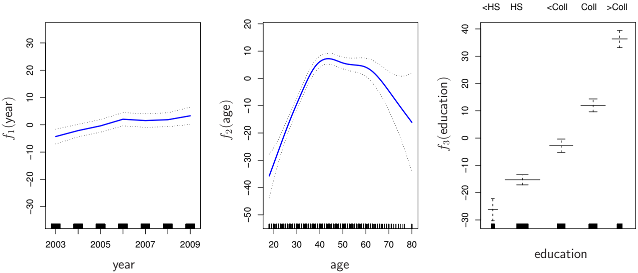


Figure: 7.12. Details are as in Figure 7.11, but now f_1 and f_2 are **smoothing splines** with four and five degrees of freedom, respectively.

Pros and Cons of GAM

- It is nonlinear (potentially more accurate than linear if linear relation is not true)
- Additivity:
  - examine the effect of each x_j on the response y while holding all of the other variables fixed;
  - inference is possible;
  - the smoothness of the function f_j for the variable X_j can be summarized via degrees of freedom.
- **Interactions are missed**: add low-dimensional interaction functions of the form $f_{jk}(X_j, X_k)$, or high order interactions.

GAM also work for generalized linear model

- In general we have

$$E(Y|X) = g(f_1(X_1) + ... + f_p(X_p))$$

  where g is known link function.
- For example, for logistic GAM:

$$P(Y = 1|X) = \frac{\exp(f_1(X_1) + ... + f_p(X_p))}{1 + \exp(f_1(X_1) + ... + f_p(X_p))}$$

Logistic GAM

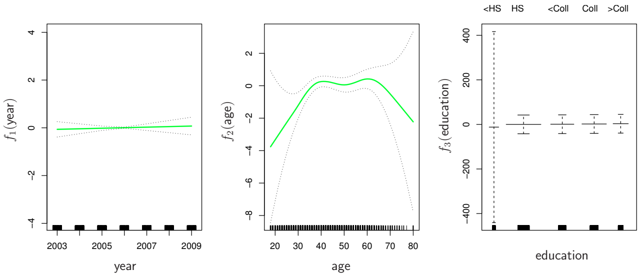


Figure: 7.13. For the Wage data, the logistic regression GAM given in (7.19) is fit to the binary response I(wage$>$ 250) Each plot displays the fitted function and pointwise standard errors. The first function is **linear** in year, the second function a **smoothing spline** with five degrees of freedom in age, and the third a **step** function for education. There are very wide standard errors for the first level $<$HS of education.