Moving Beyond Linearity

Jasmine. Hao[1]

[1]University of Hong Kong

ECON 3225: Big Data Economics

Outline

Outline

Polynomial Regression

- Historically, the standard way to extend linear regression to settings where the relationship between predictors and response is nonlinear has been to replace the standard linear model $y_i = \beta_0 + \beta_1 x_i + \epsilon_i$ with a *polynomial function*.
- $y_i = \beta_0 + \beta_1 x_i + \beta_2 x_i^2 + \beta_3 x_i^3 + \cdots + \beta_d x_i^d + \epsilon_i$, where ϵ_i is the error term.
- For a sufficiently large degree d, a polynomial regression allows us to produce an extremely *non-linear curve*.
- Generally, it is unusual to use d greater than 3 or 4 because for large values of d, the polynomial curve can become overly flexible and can take on very strange shapes.

Polynomial Illustration

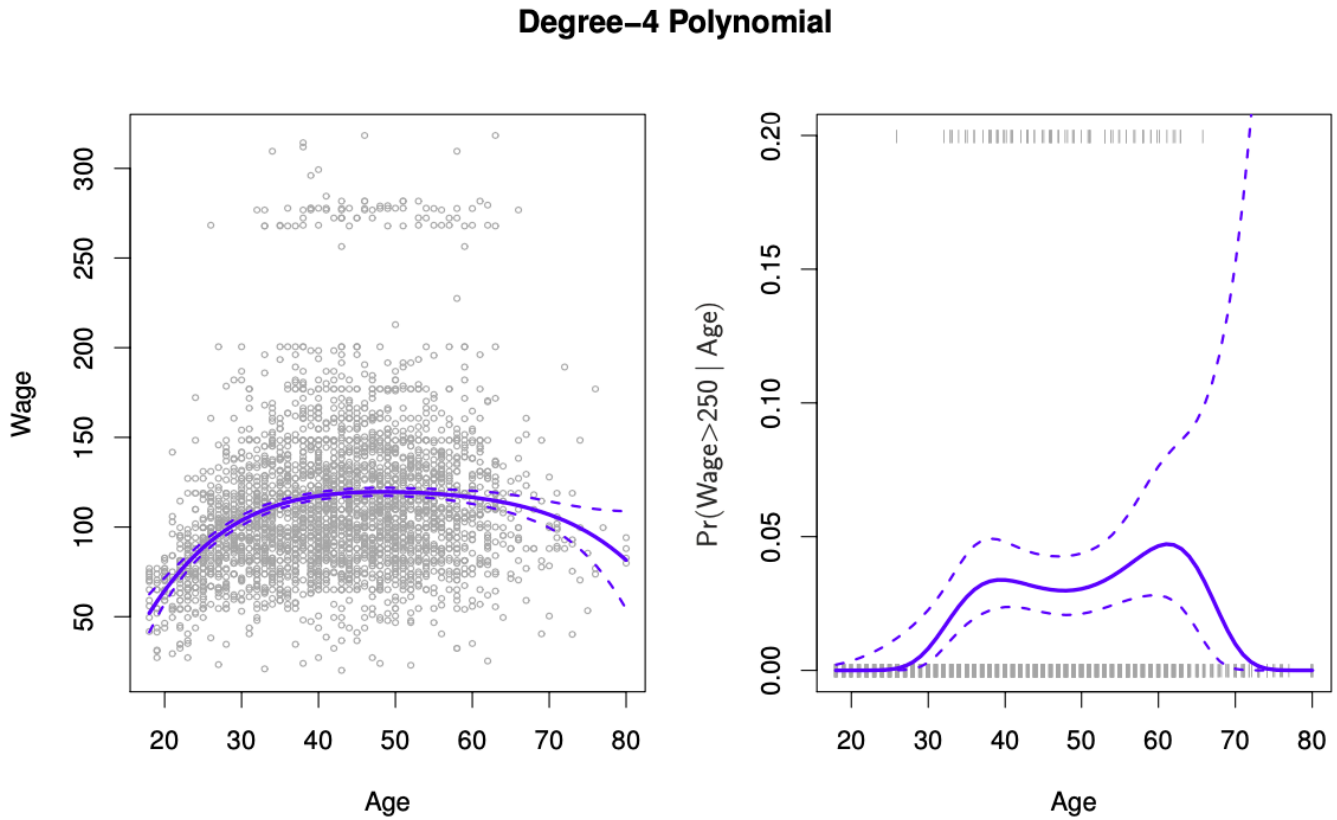


FIGURE 7.1. *The* `Wage` *data.* Left: *The solid blue curve is a degree-4 polynomial of* `wage` *(in thousands of dollars) as a function of* `age`*, fit by least squares. The dashed curves indicate an estimated 95 % confidence interval.* Right: *We model the binary event* `wage>250` *using logistic 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.*

Prediction and Variance

- Suppose we have computed the fit at a particular value of age,
  - $\hat{f}(x_0) = \hat{\beta}_0 + \hat{\beta}_1 x_0 + \hat{\beta}_2 x_0^2 + \hat{\beta}_3 x_0^3$
- What is the variance of the fit, i.e., $\text{Var}(\hat{f}(x_0))$?
- Least squares returns *variance estimates* for each of the fitted coefficients $\hat{\beta}_j$, as well as the covariances between pairs of coefficient estimates.
- These estimates can be used to compute the variance of $\hat{f}(x_0)$.

Polynomial with Logit: High and Low Earners

- Analysis reveals two distinct groups: *high earners* with income over $250,000 per annum and *low earners*.
- We can categorize wage into these binary groups for further analysis.
- Logistic regression is employed to predict this binary response, utilizing polynomial functions of age as predictors:
  - The model is given by:

$$\Pr(y_i > 250|x_i) = \frac{\exp(\beta_0 + \beta_1 x_i + \beta_2 x_i^2 + \beta_3 x_i^3)}{1 + \exp(\beta_0 + \beta_1 x_i + \beta_2 x_i^2 + \beta_3 x_i^3)}$$

Polynomial with Logit: Confidence Intervals

- The confidence intervals for the logistic regression model are particularly wide on the right-hand side.
- This is attributed to the following factors:
  - The substantial sample size (n = 3,000) contrasts with the limited number of high earners (only 79).
  - This discrepancy results in high variance in the estimated coefficients, leading to wide confidence intervals.

Step Functions

- Using polynomial functions of the features as predictors in a linear model imposes a global structure on the non-linear function of X.
- We can instead use *step functions* in order to avoid imposing such a global structure.
- Here we break the range of X into bins, and fit a different constant in each bin.
- This amounts to converting a continuous variable into an ordered categorical variable

Ordered Categorical Variable

- In greater detail, we create cutpoints $c_1, c_2, ..., c_K$ and then construct $K + 1$ new variables
-

$$
\begin{aligned}
C(X) &= I(X < c_1), \\
C_1(X) &= I(c_1 \leq X < c_2), \\
C_2(X) &= I(c_2 \leq X < c_3), \\
C_2(X) &= I(c_2 \leq X < c_3), \\
&\vdots \\
C_{K-1}(X) &= I(c_{K-1} X < c_K), \\
C_K(X) &= I(c_K \leq X)
\end{aligned}
$$

- where $I(\cdot)$ is an indicator function that returns 1 if the condition is true.
  - returns a 1 if the condition is true, and returns a 0 otherwise.
  - These are sometimes called dummy variables.

Fit Data

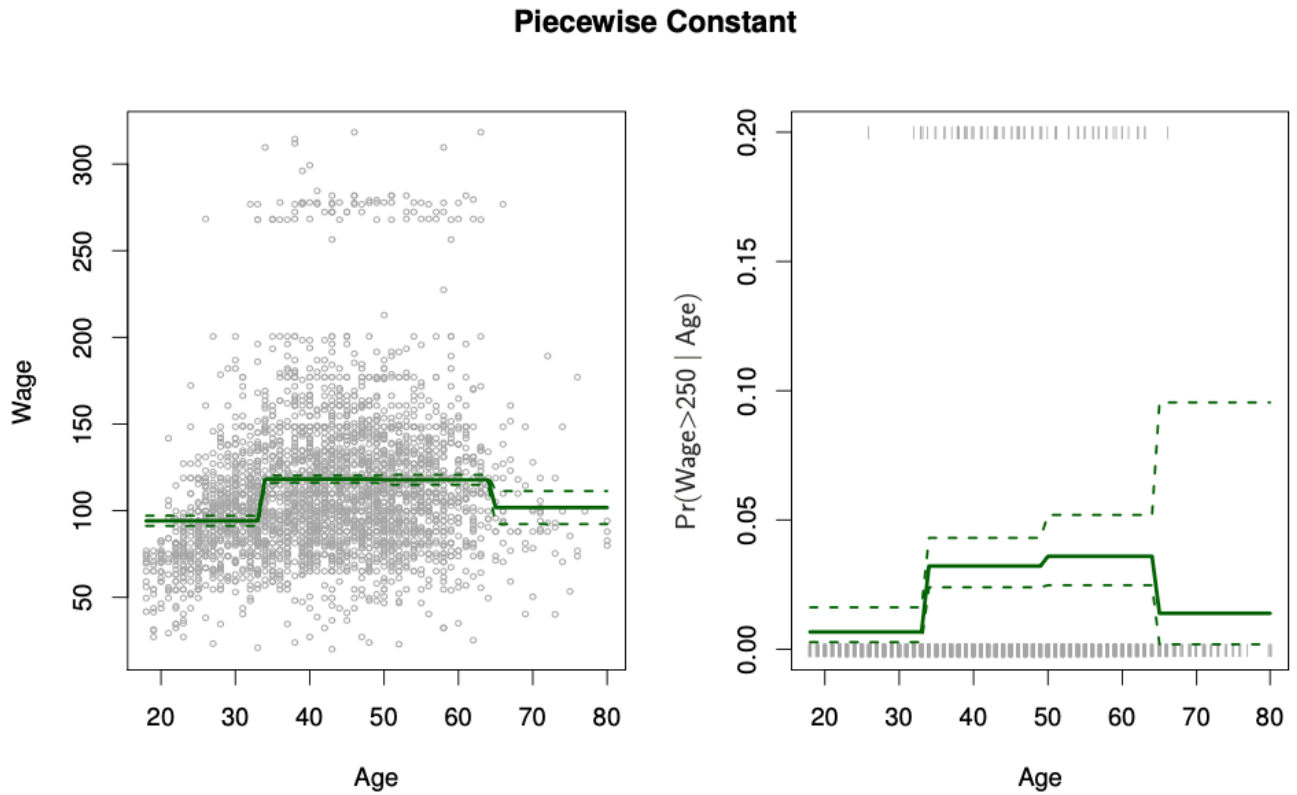


FIGURE 7.2. *The* `Wage` *data.* Left: *The solid curve displays the fitted value from a least squares regression of* `wage` *(in thousands of dollars) using step functions of* `age`*. The dashed curves indicate an estimated 95 % confidence interval.* Right: *We model the binary event* `wage>250` *using 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.*

Prediction Using Step Functions

- In step functions, any value of X leads to a partition of unity: $C_0(X) + C_1(X) + \cdots + C_K(X) = 1$.
- We can fit a linear model using these step functions as predictors:
  - $y_i = \beta_0 + \beta_1 C_1(x_i) + \beta_2 C_2(x_i) + \cdots + \beta_K C_K(x_i) + \varepsilon_i$.
- This is a special case of the more general *basis function approach*.

General Basis Function Approach

- In the basis function approach, we transform X using a family of functions $b_1(X), b_2(X), \ldots, b_K(X)$, and fit the model:
  - $y_i = \beta_0 + \beta_1 b_1(x_i) + \ldots + \beta_K b_K(x_i) + \epsilon_i$.
- The basis functions $b_1(\cdot), b_2(\cdot), \ldots, b_K(\cdot)$ are predetermined and known.
- For example, in polynomial regression, $b_j(x_i) = x_i^j$, and in step functions, $b_j(x_i) = I(c_j \leq x_i < c_{j+1})$.
- This framework allows for the use of all inference tools available for linear models, such as standard errors and F-statistics, in a more flexible modeling setting.

Other Types of Basis Functions

- Regression Splines
- Local Polynomial Regression
- Kernel Regression

Outline

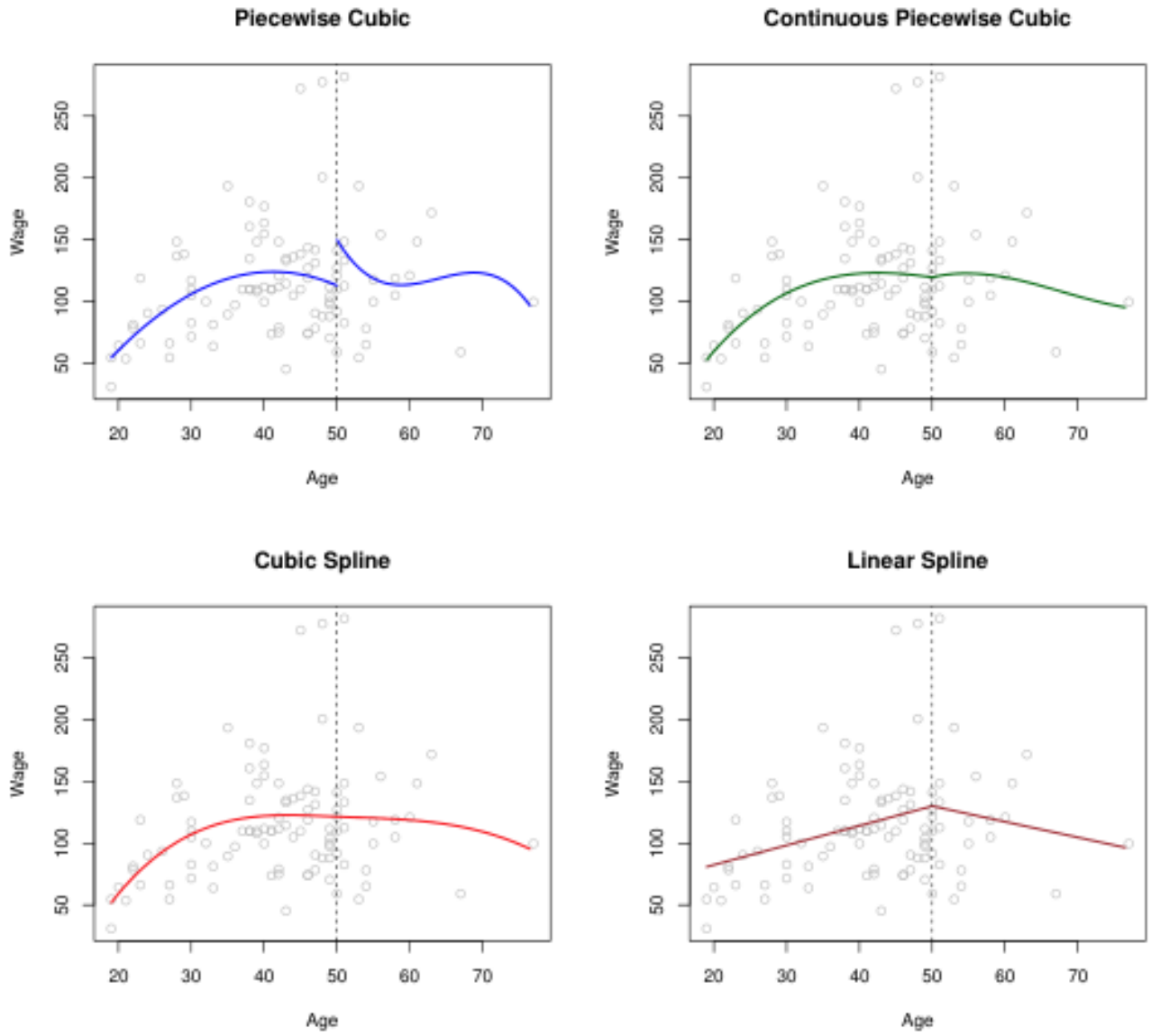
Piecewise Polynomials - Introduction

- Piecewise polynomial regression involves fitting separate low-degree polynomials over different regions of X.
- A typical example is the *piecewise cubic polynomial*, where we fit a cubic regression model:
  - $y_i = \beta_0 + \beta_1 x_i + \beta_2 x_i^2 + \beta_3 x_i^3 + \epsilon_i$,
  - The coefficients $\beta_0, \beta_1, \beta_2$, and β_3 vary in different parts of the range of X.
  - The points where the coefficients change, leading to a different polynomial being used, are called **knots**.

Piecewise Polynomials - Examples

- A *piecewise cubic polynomial* with no knots is equivalent to a standard cubic polynomial.
- With a single knot at a point c, the piecewise cubic polynomial takes the form:
  - $$y_i = \begin{cases} \beta_{01} + \beta_{11} x_i + \beta_{21} x_i^2 + \beta_{31} x_i^3 + \epsilon_i & \text{if } x_i < c, \\ \beta_{02} + \beta_{12} x_i + \beta_{22} x_i^2 + \beta_{32} x_i^3 + \epsilon_i & \text{if } x_i \geq c. \end{cases}$$
- The flexibility of the piecewise polynomial increases with the number of knots.
- In general, if we place K different knots throughout the range of X, we will end up fitting $K + 1$ different cubic polynomials.

Constraints and Splines - Visualization



Constraints and Splines - Explanation

- **Top Left:** The cubic polynomials are *unconstrained*. This means that there are no restrictions on the behavior of the polynomial at the knots, leading to potential discontinuities.
- **Top Right:** The cubic polynomials are constrained to be *continuous* at the knot (age=50). This ensures that the polynomial does not have any sudden jumps at the knot.
- **Bottom Left:** The cubic polynomials are further constrained to have *continuous first and second derivatives* at the knot. This results in a smoother transition at the knot, with no abrupt changes in slope or curvature.
- **Bottom Right:** A *linear spline* is shown, which is constrained to be continuous. Unlike the cubic splines, linear splines use straight-line segments, resulting in a piecewise linear function.

The Spline Basis Representation

- To fit a piecewise degree-d polynomial with continuity constraints, we use a *spline basis representation*.
- A cubic spline with K knots can be modeled as:

$$y_i = \beta_0 + \beta_1 b_1(x_i) + \beta_2 b_2(x_i) + \cdots + \beta_{K+3} b_{K+3}(x_i) + \epsilon_i$$

  with appropriate basis functions $b_1, b_2, \ldots, b_{K+3}$, which can be estimated using least squares.
- The most direct way to represent a cubic spline is to start off with a basis for a cubic polynomial and add one *truncated power basis function* per knot.

Truncated Power Basis

- A truncated power basis function is defined as:

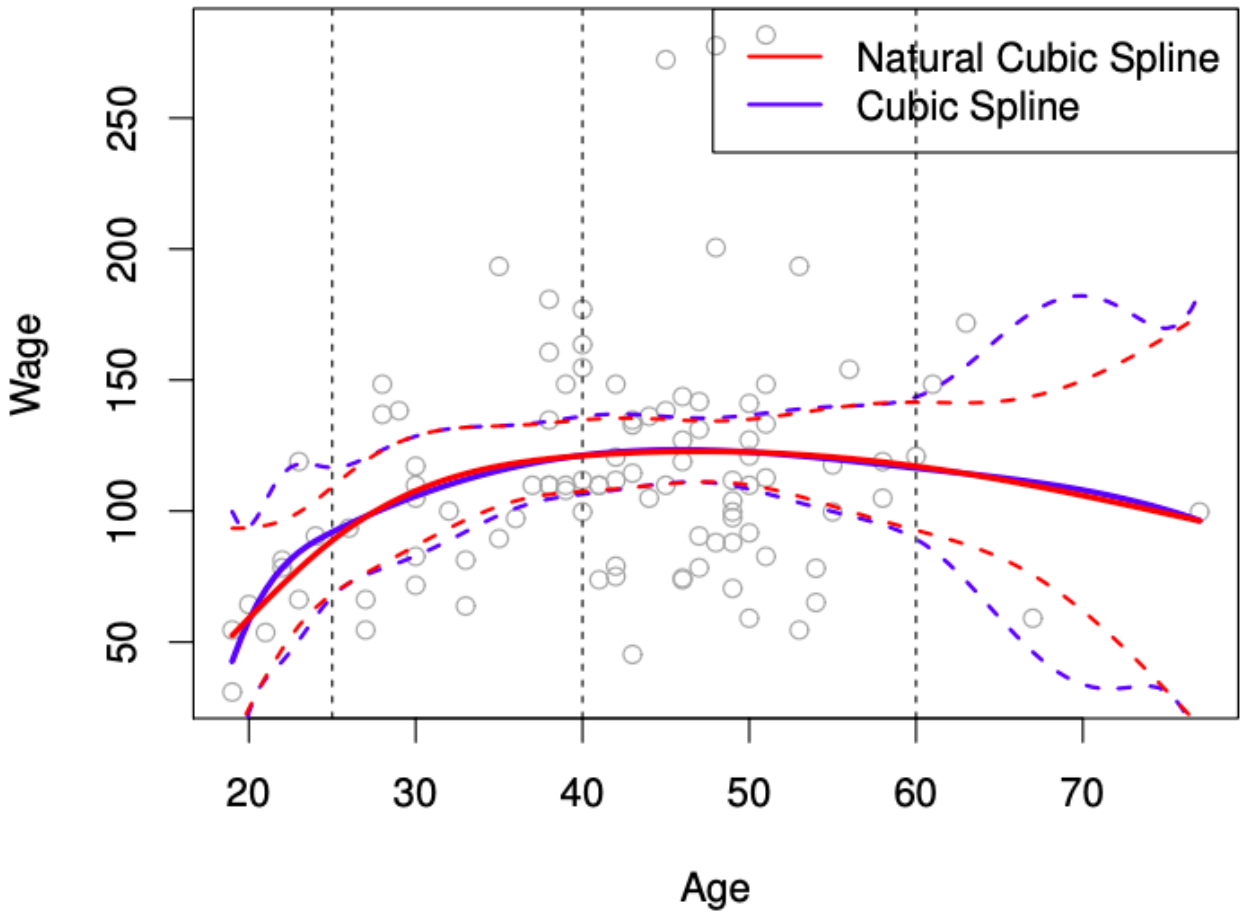
$$h(x, \xi) = (x - \xi)_+^3 = \begin{cases} (x - \xi)^3 & \text{if } x > \xi \\ 0 & \text{otherwise} \end{cases}$$

- Adding this term to a cubic polynomial introduces discontinuity only in the third derivative at ξ, ensuring the function remains continuous.
- The least squares fit is performed using the basis functions:

$$X, X^2, X^3, h(X, \xi_1), h(X, \xi_2), \ldots, h(X, \xi_K)$$

where $\xi_1, \ldots, \xi_K$ are the *knots*.

[Illustration] Natural Cubic Spline



Natural Spline

- Splines can exhibit high variance at the edges of the predictor range, especially when X is very small or very large.
- This is evident from the figure, where the confidence bands in the boundary region appear erratic.
- A **natural spline** is a type of regression spline with additional boundary constraints:
  - The spline function is required to be linear beyond the outermost knots. This means that the function behaves linearly where X is smaller than the smallest knot or larger than the largest knot.
- These boundary constraints generally lead to more stable estimates at the boundaries, reducing the variance observed with regular splines.

Choosing the Locations of the Knots

- The placement of knots is crucial as it determines the **flexibility** of the spline.
- A larger number of knots in a specific region allows the spline to **adapt more freely** to the data, leading to rapid changes in the polynomial coefficients.
- Conversely, fewer knots imply a **smoother, more stable** function.
- While it is tempting to place more knots in regions of rapid change and fewer knots in stable regions, this strategy requires **prior knowledge** of the data's behavior.
- In practice, a common approach is to distribute knots **uniformly** across the range of X, which provides a balance between flexibility and stability.

[Illustration]Fitted Model

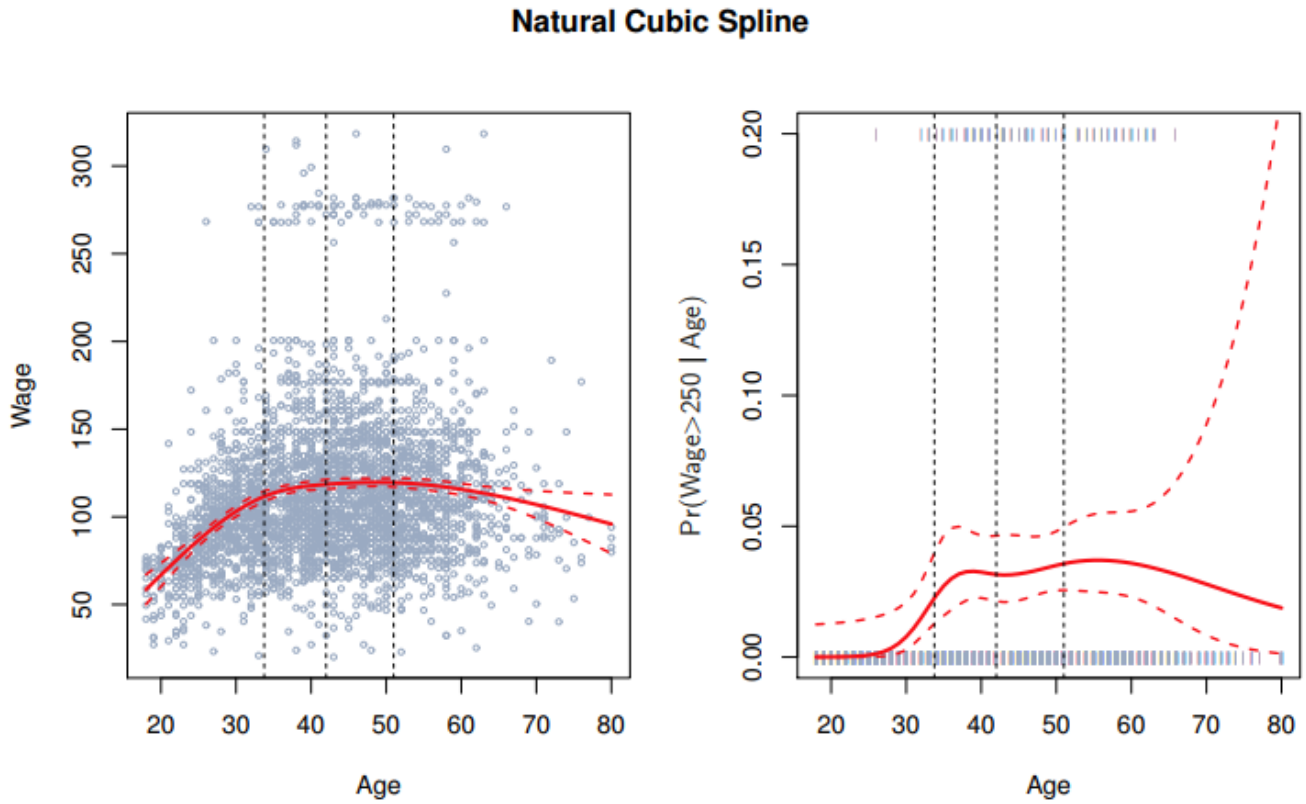


FIGURE 7.5. *A natural cubic spline function with four degrees of freedom is fit to the* `Wage` *data.* Left: *A spline is fit to* `wage` *(in thousands of dollars) as a function of* `age`. Right: *Logistic regression is used to model the binary event* `wage>250` *as a function of* `age`. *The fitted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

How many knots to use?

- How many knots should we use? Howmany degree of freedom should our spline contain?
  - One option is to try out different numbers of knots and see which produces the best looking curve.
  - Cross-validation:
    - we remove a portion of the data (say 10 %),
    - fit a spline with a certain number of knots to the remaining data,
    - use the spline to make predictions for the held-out portion.
    - repeat this process multiple times until each observation has been left out once,
    - compute the overall cross-validated RSS.
  - Choose K that give smallest RSS

Cross Validation

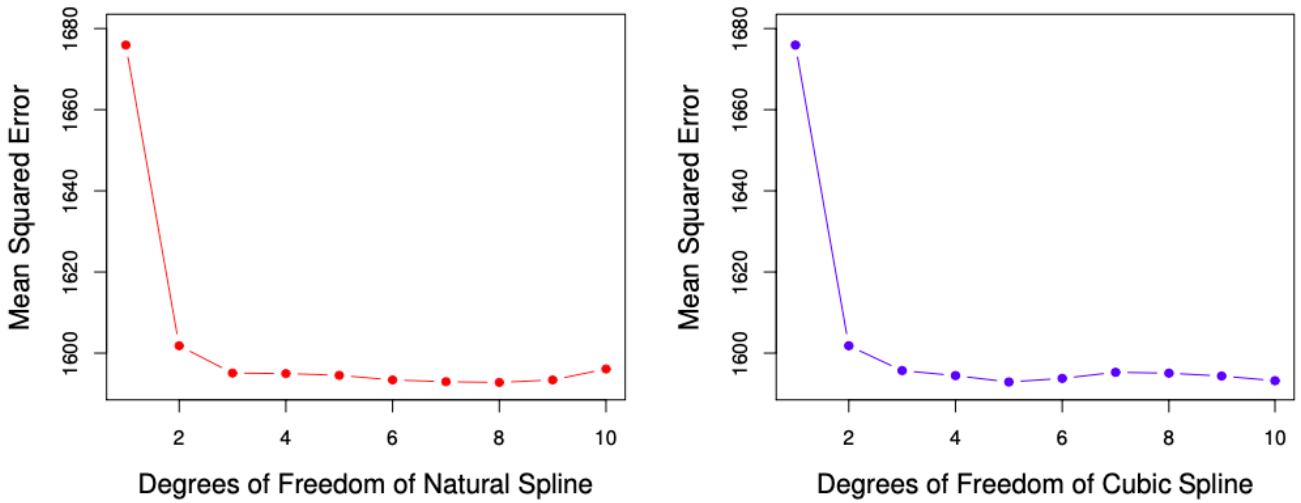


FIGURE 7.6. *Ten-fold cross-validated mean squared errors for selecting the degrees of freedom when fitting splines to the* `Wage` *data. The response is* `wage` *and the predictor* `age`. Left: *A natural cubic spline.* Right: *A cubic spline.*

Comparison - Visual Representation

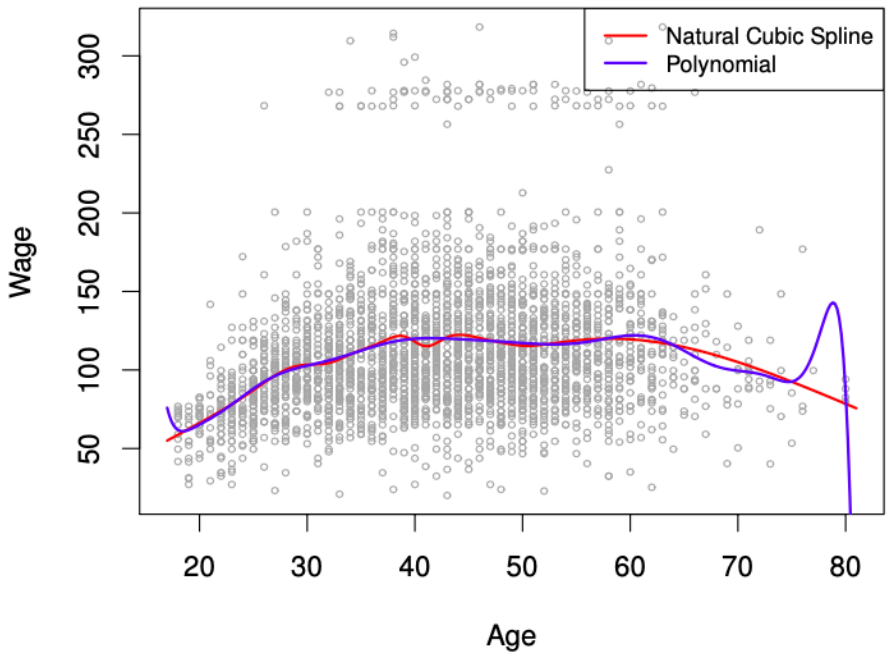


FIGURE 7.7. *On the* `Wage` *data set, a natural cubic spline with 15 degrees of freedom is compared to a degree-15 polynomial. Polynomials can show wild behavior, especially near the tails.*

Figure: Comparison of polynomial and spline fits.

Comparison - Analysis

- The extra flexibility in the polynomial produces **undesirable results at the boundaries**, while the natural cubic spline still provides a **reasonable fit** to the data.
- Regression splines often give **superior results** to polynomial regression:
  - Polynomials use a high degree to produce flexible fits, leading to **overfitting at the boundaries**.
  - Splines introduce flexibility by **increasing the number of knots** while keeping the degree fixed, resulting in **more stable estimates**.
  - Splines allow for **adaptive flexibility**, with more knots in regions where the function f is changing rapidly and fewer knots where f is more stable.

Outline

Smoothing Splines

- When fitting a smooth curve to a set of data, our goal is to find a function $g(x)$ that minimizes the **residual sum of squares (RSS)**:
  - $RSS = \sum_{i=1}^{n}(y_i - g(x_i))^2$
- However, without constraints, $g(x)$ could overfit the data. To prevent this, we use a **smoothing spline** g, which minimizes:
  - $\sum_{i=1}^{n}(y_i - g(x_i))^2 + \lambda \int g''(t)dt$
  - Here, λ is a non-negative tuning parameter that controls the trade-off between fitting the data and smoothness of the curve.

Loss + Penalty

- The **loss function** $\sum_{i=1}^{n}(y_i - g(x_i))^2$ encourages g to fit the data well.
- The **penalty term** $\lambda \int g''(t)dt$ penalizes the variability in g, with the second order derivative measuring the roughness of the curve.
- The **smoothing parameter** λ balances the trade-off between fitting the data closely (small λ) and having a smoother curve (large λ).

Effective Degree of Freedom

- The **degree of freedom** refers to the number of free parameters in a model.
- Although a smoothing spline has n parameters, these parameters are constrained, leading to a concept called **effective degrees of freedom** (df_λ).
- As the smoothing parameter λ increases from 0 to ∞, df_λ decreases from n to 2.
- The effective degrees of freedom can be defined as:
  - $\hat{\mathbf{g}}_\lambda = \mathbf{S}_\lambda \mathbf{y}$, where $\hat{\mathbf{g}}_\lambda$ is the solution to the smoothing spline minimization problem.
  - $\mathbf{S}_\lambda$ represents the matrix of fitted values for the smoothing spline.
  - $df_\lambda = \sum_{i=1}^{n} \{\mathbf{S}_\lambda\}_{ii}$, which captures the effective complexity of the model.

Leave-One-Out Cross-Validation Error (LOOCV)

- To choose the optimal smoothing parameter λ, we use the **Leave-One-Out Cross-Validation Error (LOOCV)**.
- The LOOCV error is given by:
  - $RSS_{cv}(\lambda) = \sum_{i=1}^{n} \left[y_i - \hat{g}_{\lambda}^{(-i)}(x_i) \right]^2$,
  - where $\hat{g}_{\lambda}^{(-i)}(x_i)$ is the fitted value for observation i using all data except the i-th observation.
- The optimal λ is the one that minimizes the LOOCV error.

[Illustration] Smoothing Splines

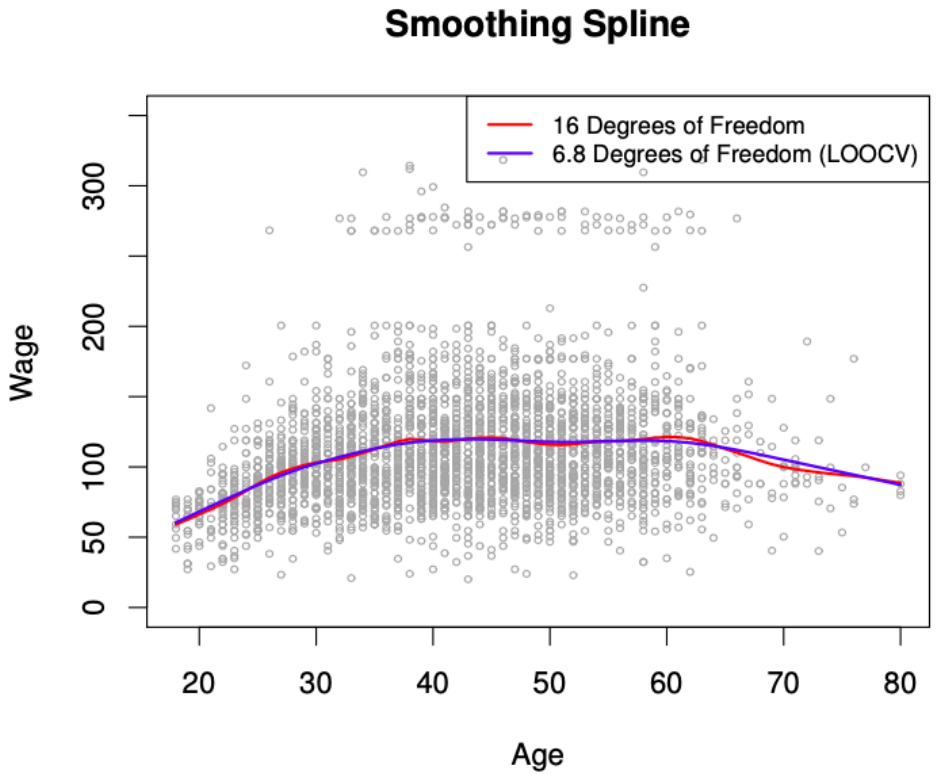


FIGURE 7.8. *Smoothing spline fits to the* `Wage` *data. The red curve results from specifying* 16 *effective degrees of freedom. For the blue curve,* λ *was found automatically by leave-one-out cross-validation, which resulted in* 6.8 *effective degrees of freedom.*

Figure: An example of a smoothing spline fit to data.

Outline

Local Regression

- **Local regression** is an approach for fitting flexible non-linear functions by computing the fit at a target point x_0 using *only the nearby training observations*.
- In local regression, we need to make several choices:
  - Define the **weighting function** K, which determines how much influence nearby points have on the fit at x_0.
  - Decide whether to fit a **linear**, **constant**, or **quadratic** regression model locally.
  - Choose the **span** s, which controls the size of the neighborhood around x_0 that is used for the fit:
    - A smaller value of s results in a more *local* and potentially *wiggly* fit, focusing on a smaller neighborhood.
    - A larger value of s leads to a more *global* fit, using a larger portion of the training observations.

[Illustration] Local Regression

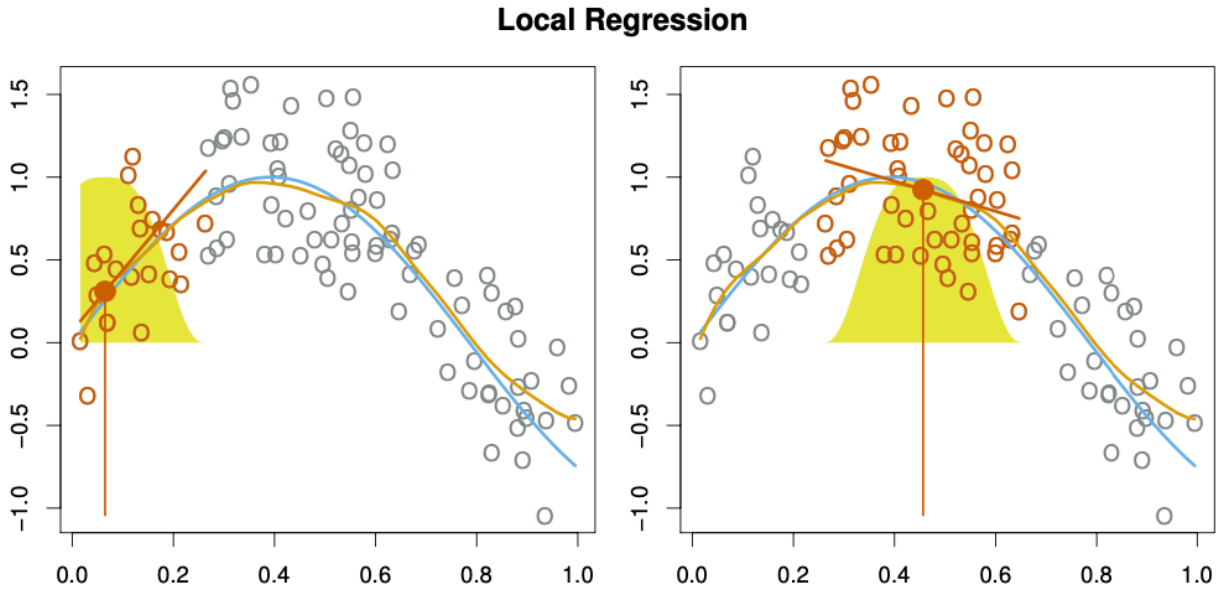


FIGURE 7.9. *Local regression illustrated on some simulated data, where the blue curve represents $f(x)$ from which the data were generated, and the light orange curve corresponds to the local regression estimate $\hat{f}(x)$. The orange colored points are local to the target point x_0, represented by 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 $\hat{f}(x_0)$.*

Algorithm

Algorithm 7.1 *Local Regression At* $X = x_0$

1. Gather the fraction $s = k/n$ of training points whose x_i are closest to x_0.

2. Assign a weight $K_{i0} = K(x_i, x_0)$ to each point in this neighborhood, so that the point furthest from x_0 has weight zero, and the closest has the highest weight. All but these k nearest neighbors get weight zero.

3. Fit a *weighted least squares regression* of the y_i on the x_i using the aforementioned weights, by finding $\hat{\beta}_0$ and $\hat{\beta}_1$ that minimize

$$\sum_{i=1}^{n} K_{i0}(y_i - \beta_0 - \beta_1 x_i)^2. \tag{7.14}$$

4. The fitted value at x_0 is given by $\hat{f}(x_0) = \hat{\beta}_0 + \hat{\beta}_1 x_0$.

Generalization

- **varying coefficient models**
  - global in some variables with multiple features, but local in another, such as time.
  - a useful way of adapting a model to the most recently gathered data.
- **two-dimensional neighborhoods**
  - use observations that are near each target point in two-dimensional space.
  - Theoretically the same approach can be implemented in higher dimensions,
  - However, local regression can perform poorly if p is much larger than about 3 or 4
  - very few training observations close to x_0.

Outline

Generalized Additive Models

- Generalized additive models (GAMs) provide a general framework for extending a standard linear model by allowing nonlinear functions of each of thevariables.
- Unlike traditional linear models, GAMs can capture nonlinear and non-monotonic relationships, as well as interactions between predictors.
- GAMs are constructed by fitting a separate smooth function to each predictor variable, and then combining these functions to form the final model.
- GAMs can handle a wide range of data types, including continuous, categorical, and ordinal variables, as well as missing data and outliers.
- GAMs can be used for both regression and classification problems, and can incorporate regularization techniques to prevent overfitting.

GAMs for Regression Problems

- A natural way to extend the multiple linear regression model
  - $y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ldots + \beta_p x_{ip} + \varepsilon_i$
- Replace the linear component with a non-linear component
  - $y_i = \beta_0 + f_1(x_{i1}) + f_2(x_{i2}) + \ldots + f_p(x_{ip}) + \varepsilon_i$

We fit the data with $wage = \beta_0 + f_1(year) + f_2(age) + f_3(education) + \varepsilon$

[Illustration] GAM with Regression

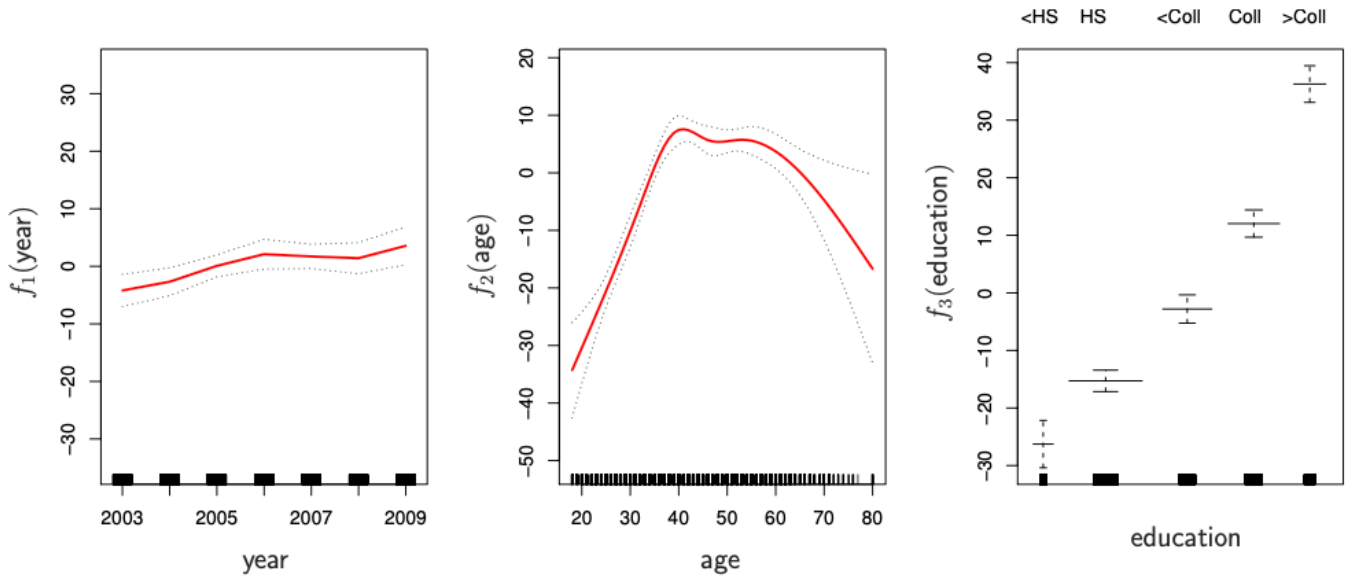


FIGURE 7.11. *For the* `Wage` *data, plots of the relationship between each feature and the response,* `wage`*, in the fitted model (7.16). Each plot displays the fitted function and pointwise standard errors. The first two functions are natural splines in* `year` *and* `age`*, with four and five degrees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

In Sections 7.1–7.6, we discuss many methods for fitting functions to a single variable. The beauty of GAMs is that we can use these methods as building blocks for fitting an additive model. In fact, for most of the methods that we have seen so far in this chapter, this can be done fairly trivially. Take, for example, natural splines, and consider the task of fitting the model

Pros and Cons of GAMs

- **Generalized Additive Models (GAMs)** allow for fitting a non-linear function f_j to each predictor X_j, enabling the modeling of **non-linear relationships** that standard linear regression may miss.
- These non-linear fits can potentially provide **more accurate predictions** for the response Y.
- GAMs allow for the examination of the effect of each X_j on Y **individually**, while holding other variables constant.
- The **smoothness** of the function f_j is described using **degrees of freedom**.
- One drawback is that with many variables, **important interactions** can be missed. These can be addressed by:
  - Manually adding interaction terms.
  - Using **two-dimensional splines** to capture interactions between variables.

GAMs for Classification Problems

- With $p(X) = Pr(Y = 1|X)$, the logistic regression model
  - $\log\left(\frac{p(X)}{1-p(X)}\right) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_p X_p$
- A natural way to extend
  - $\log\left(\frac{p(X)}{1-p(X)}\right) = \beta_0 + f_1(X_1) + f_1(X_2) + \ldots + f_p(X_p)$
- We fit the data wage with
  - $\log\left(\frac{p(X)}{1-p(X)}\right) = \beta_0 + f_1 \times year + f_1(wage) + f_3(education)$,
  - where $p(X) = \Pr(wage > 250|year, age, education)$

[Illustration] GAM with Regression

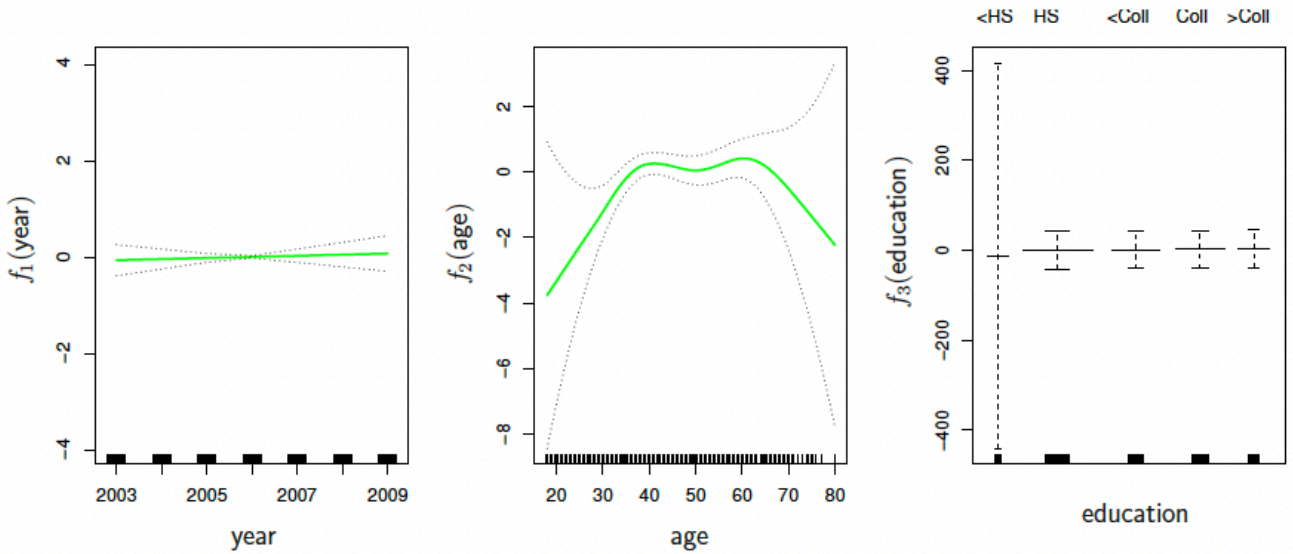


FIGURE 7.13. *For the* `Wage` *data, the logistic regression GAM given in (7.19) is fit to the binary response* `I(wage>250)`*. Each plot displays the fitted function and pointwise standard errors. The first function is linear in* `year`*, the second function a smoothing spline with five degrees of freedom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

Lab: Non-linear Modeling

- Dataset: The Wage Data
- Polynomial Regression and Step Functions
- Splines
- GAMs

For Further Reading I

James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). An introduction to statistical learning (Vol. 112, p. 18). New York: