Moving Beyond Linearity

Lu Haibo

Monday, Jun 1, 2015

Introduction

Linear models

- are easy to describe and implement
- have advantages over other approaches in terms of interpretation and inference
- but can have significant limitations in terms of predictive power
  - the linearity assumptios is almost always an approximation, and sometimes a poor one
  - the Ridge regression, lasso, principal components regression, and other techniques may improve the result by reducing the complexity of the linear model , and hence the variance of the estimates. But they are still using a linear model

Now we relax the linearity assumption while still attempting to maintain as much interpretability as possible.

- *Polynomial regression*
- *Step functions*
- *Regression splines*
- *Smoothing splines*
- *Local regression*
- *Generalized additive models*

Polynomial Regression

Historically, the standard way to extend linear regression is to replace the relationship between the predictors and the response by a polynomial function

$$y_i = \beta_0 + \beta_1 x_i + \beta_2 x_i^2 + \cdots + \beta_d x_i^d + \epsilon_i$$

- For large enough degree d, a polynomial regression allows to produce an extremely non-linear curve.
- 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 some very strange shapes. This is especially true near the boundary of the X variable.

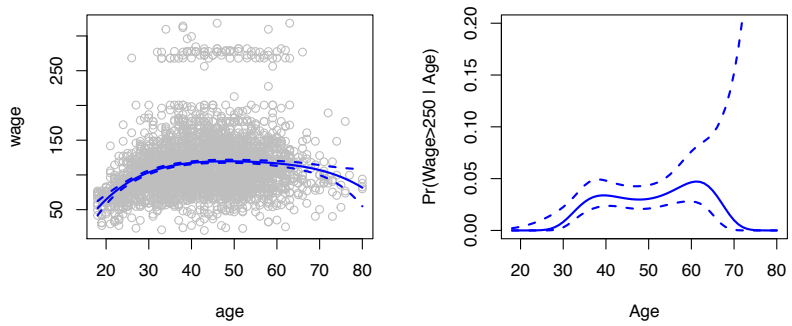

Figure 1: The Wage data. Left: The solid blue curve is a degree-4 polynomial of wage (in thousands of dollars) as a function of age, fit by least squares. The dotted curves indicate an estimated 95% confidence interval. Right: We model the binary event wage>250 using logistic regression, again with a degree-4 polynomial. The fitted posterior probability of wage exceeding $250,000 is shown in blue, along with an estimated 95% confidence interval.

Step Functions (piecewise constant functions)

- polynomial functions: a *global* structure on the non-linear function of X
- *step functions*: a *local* structure

Create cutpoints $c_1, c_2, \ldots, c_K$ in the range of X, and then construct $K+1$ new variables

$$\begin{aligned} C_0(X) &= I(X < c_1), \\ C_1(X) &= I(c_1 \leq X < c_2), \\ C_2(X) &= I(c_2 \leq X < c_3), \\ &\vdots \\ C_{K-1}(X) &= I(c_{K-1} \leq X < c_K), \\ C_K(X) &= I(c_K \leq X), \end{aligned}$$

where $I(\cdot)$ is an *indicator function* that returns a 1 if the condition is true, and return a 0 otherwise. These are sometimes called *dummy* variables. Notice that for any value of X, $C_0(X) + C_1(X) + \ldots + C_K(X) = 1$.

Basis Functions

Polynomial and piecewise-constant regression model are in fact special cases of a *basis function* approach.

Instead of fitting a linear model in X, we fit the model

$$y_i = \beta_0 + \beta_1 b_1(x_i) + \beta_2 b_2(x_i) + \ldots + \beta_K b_K(x_i) + \epsilon,$$

where the basis functions $b_1(\cdot), b_2(\cdot), \ldots, b_K(\cdot)$ are fixed and known. Hence, we can use least squares to estimate the unknown regression coefficients β_j.

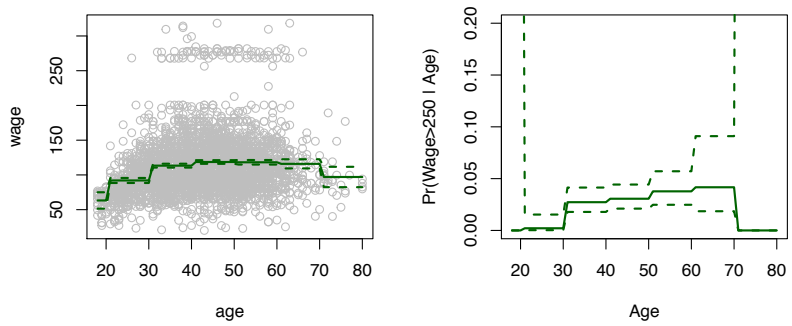

Figure 2: The Wage data. Left: The solid curve displays the fitted value from a least squares regression of wage (in thousands of dollars) using step functions of age. The dotted curves indicate an estimated 95% confidence interval. Right: We model the binary event wage>250 using logistic regression, again using step functions of age. The fitted posterior probability of wage exceeding $250,000 is shown, along with an estimated 95% confidence interval.

Commonly used basis functions

- Splines
- Fourier series
- Wavelets

Regression Splines

Piecewise Polynomials

Instead of fitting a high-degress polynomial over the extire range of X, *piecewise polynomial regression* involves fitting separate low-degree polynomials over different regions of X

For example, a piecewise cubic polynomial works by fitting a cubic regression model of the form

$$y_i = \begin{cases} \beta_{01} + \beta 11 x_i + \beta_{21} x_i^2 + \beta_{31} x_i^3 + \epsilon_i, & x_i < c, \\ \beta_{02} + \beta 12 x_i + \beta_{22} x_i^2 + \beta_{32} x_i^3 + \epsilon_i, & x_i \geq c. \end{cases}$$

- The points where the coefficients change are called *knots* - Using more knots leads to a more flexible piecewise polynomial - Tye piecewise constant functions are piecewise polynomials of degree 0

Constraints and Splines

The top left panel of Figure 3 looks wrong because the fitted curve is just too flexible (with a total of eight *degrees of freedom*). To remedy this problem, we can fit a piecewise polynomial under the *constraint* that the fitted curve must be continous. In other words, there cannot be a jump when age $= 50$. The top right plot in Figure 3 shows the resulting fit. This looks better than the top left plot, but the V-shaped

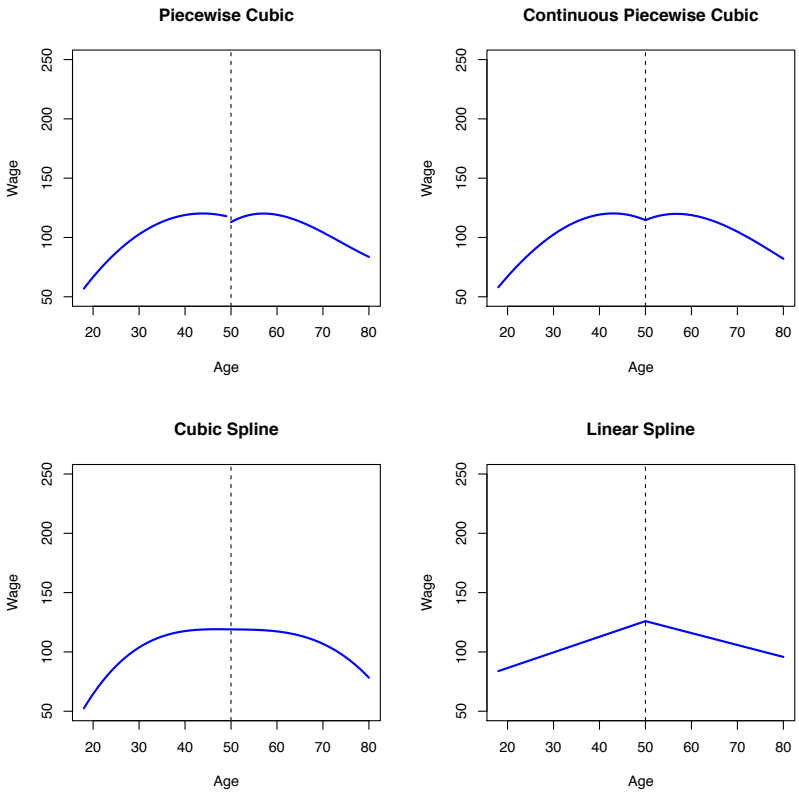

Figure 3: Various piecewise polynomials are fit to a subset of the Wage data, with a knot at age=50. Top Left: The cubic polynomials are unconstrained. Top Right: The cubic polynomials are constrained to be continuous at age=50. Bottom Left: The cubic polynomials are constrained to be continuous, and to have continuous first and second derivatives. Bottom Right: A linear spline is shown, which is constrained to be continuous.

join looks unnatural.

$$\begin{aligned} Wage =& \beta_0 + \beta_1 \times Age + \beta_2 \times Age^2 + \beta_3 \times Age^3 \\ &+ \beta_4 \times (Age - 50)^3_+ + \beta_5 \times (Age - 50)^2_+ + \beta_6 \times (Age - 50)_+ \end{aligned}$$

The Spline Basis Representation

We can add two additional constraint: both the first and second derivatives of the piecewise polynomials are continuous at age $= 50$. The curve in the bottom left plot is called a *cubic spline*[1].

[1] Cubic splines are popular because most human eyes cannot detect the discontinuity at the knots.

Each constraint that we impose on the piecewise cubic polynomials effectively frees up one degree of freedom, and reduces the complexity of the resulting piecewise polynomial fit.

$$\begin{aligned} Wage =& \beta_0 + \beta_1 \times Age + \beta_2 \times Age^2 + \beta_3 \times Age^3 \\ &+ \beta_4 \times (Age - 50)^3_+ \end{aligned}$$

In order to fit a cubic spline to a data set with K knots, we perform least squares regression with an intercept and $3 + K$ predictors, of the form

$$X, X^2, X^3, (X - \xi_1)^3_+, (X - \xi_2)^3_+, \ldots, (X - \xi_K)^3_+$$

where $\xi_1, \ldots, \xi_K$ are the knots. This amounts to estimating a total of $K + 4$ regression coefficients, so, fitting a cubic spline with K knots has $K + 4$ degrees of freedom.

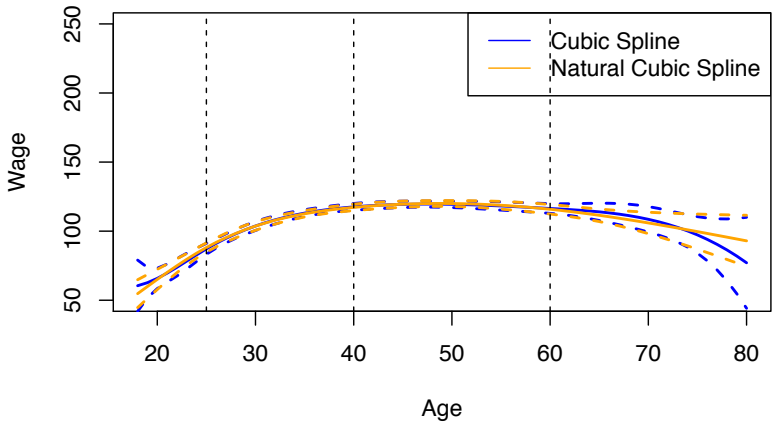

Figure 4: A cubic spline and a natural cubic spline, with three knots, fit to a subset of the Wage data. We see that the confidence bands in the boundary region appear fairly wild.

Natural Spline: a regression spline with additional *boundary constraints*

- The function is required to be linear at the boundary (in the region where X is smaller than the smallest knot, or larger than the largest knot).
- This additional constraint means that natural splines generally produce more stable estimates at the boundaries.

Choosing the Number and Locations of the Knots

The regression spline is most flexible in regions that contain a lot of knots. Hence, one option is to

- place more knots in places where we feel the function might vary most rapidly
- place fewer knots where it seems more stable

While in practice it is common to place knots in a uniform way:

- specify the desired degress of freedom
- let the software automatically place the knots at uniform quantiles of the data

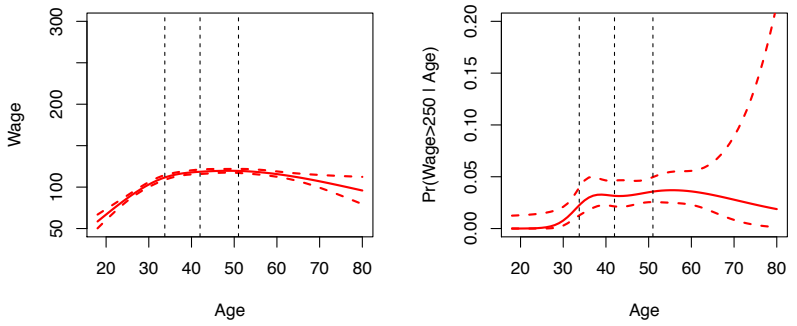

Figure 5: A natural cubic spline function with four degrees of freedom is fit to the Wage data. Left: A spline is fit to wage (in thousands of dollars) as a function of age. Right: Logistic regression is used to model the binary event wage>250 as a function of age. The fitted posterior probability of wage exceeding $250,000 is shown. (There are actually five knots, including the two boundary knots. A cubic spline with five knots would have nine degrees of freedom. But natural cubic splines have two additional natural constraints at each boundary to enforce linearity, resulting in $9 - 4 = 5$ degrees of freedom. Since this includes a constant, which is absorbed in the intercept, we count it as four degrees of freedom.)

We can use cross-validation to decide how many knots should use.

Comparison to Polynomial Regression

Regression splines often give superior results to polynomial regression. This is because:

- Unlike polynomials, which must use a high degree to produce flexible fit, splines introduce flexibility by increasing the number of knots but keeping the degree fixed. Generally, this approach produces more stable estimates

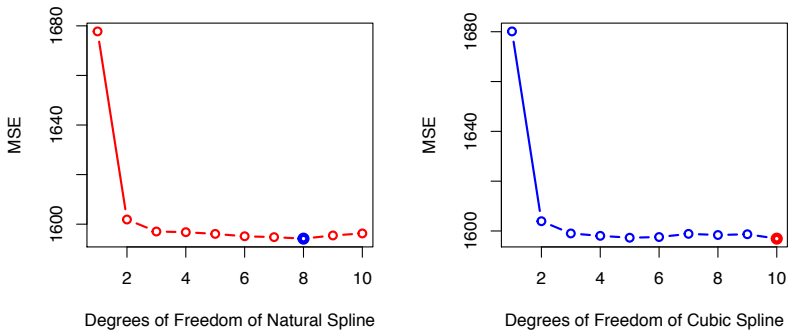

Figure 6: Ten-fold cross-validated mean squared errors for selecting the degrees of freedom when fitting splines to the Wage data. The response is wage and the predictor age. Left: A natural cubic spline. Right: A cubic spline.

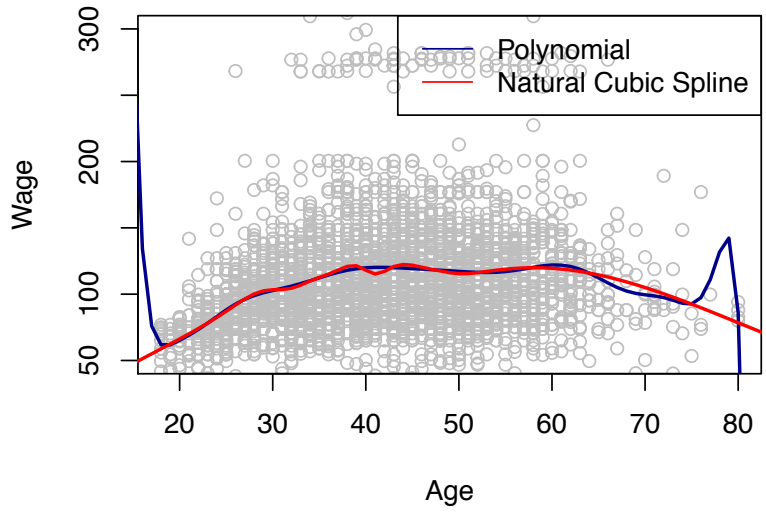

Figure 7: On the Wage data set, a natural cubic spline with 15 degrees of freedom is compared to a degree-15 polynomial. Polynomials can show wild behavior, especially near the tails.

Smoothing Splines

In regression splines, we create by specifying a set of knots, producing a sequence of basis functions, and then using least squares to estimate the spline coefficients.

In fitting a smooth curve to a set of data, what we really want to do is find some function, say $g(x)$, that fits the observed data well:

- We want $RSS - \sum_{i=1}^{n}(y_i - g(x_i))^2$ to be samll
- But if we don't put any constraints on $g(x)$, then we can always make RSS zero simply by choosing g such that is *interpolates* all of the y_i. Such a function would clearly overfit the data—it would be far too flexible
- What we really want is a function g that makes RSS small, but also *smooth*

A natural approach is to find the function g that minimizes

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

where λ is a nonnegative *turning parameter*. The function g that minimizes (1) is known as a *smoothing spline*.

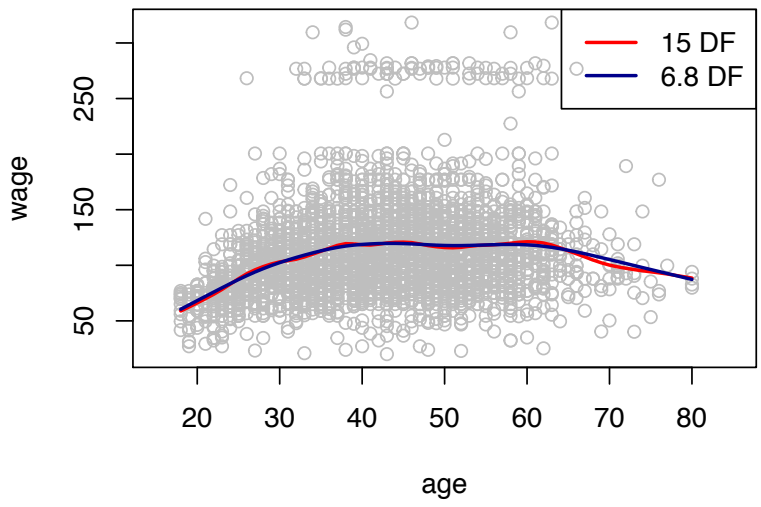

Figure 8: Smoothing spline fits to the Wage data. The red curve results from specifying 15 effective degrees of freedom. For the blue curve, λ was found automatically by leave-one-out cross-validation (LOOCV), which resulted in 6.8 effective degrees of freedom.

Theorem: The function $g(x)$ that minimizes (1) is a natural cubic spline with knots at $x_1, x_2, \ldots, x_n$. [2]

[2] However, it is not the same natural cubic spline that one would get if one applied the basis function approach with knots at $x_1, \ldots, x_n$ —— rather, it is a *shrunken* version of such a natural cubic spline, where the value of the tuning parameter λ in (1) controls the level of shrinkage. λ can be choosen by cross-validation.

Local Regression

Local regression is a different approach (*nonparametric method*) for fitting flexible non-linear functions, which involves computing the fit at a target point x_0 using only the nearby training observations.

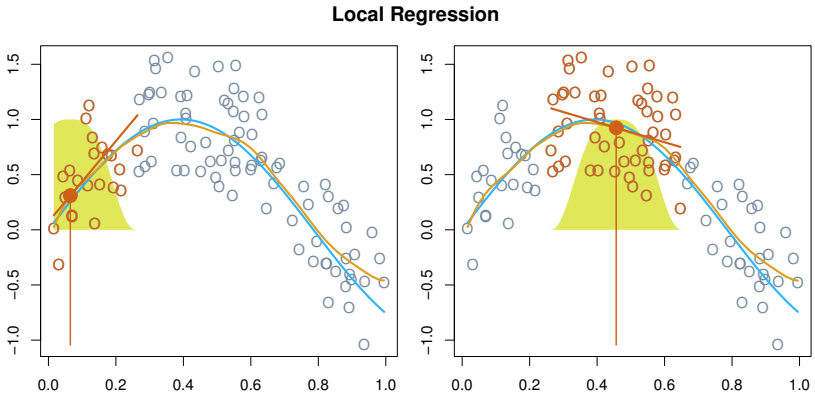

Figure 9: Local regression illustrated on some simulated data, where the blue curve represents $f(x)$ from which the data were generated, and the light orange curve corresponds to the local regression estimate $\hat{f}(x)$. The orange colored points are local to the target point x_0, represented by the 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: *Local Regression At* $X = x_0$

1. Gather the fractions $s = k/n$ of training points whose x_i are closest to x_0. (s is called the *span*)
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

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

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

In order to perform local regression, there are a number of choices to be made:

- the weighting function K
- whether to fit a constant, linear, or quadratic regression in Step 3 above
- and, the most important, span s

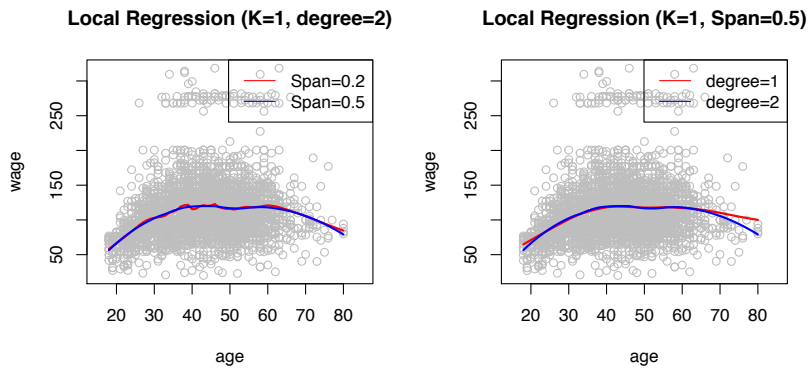

Figure 10: Local regression using spans of 0.2 and 0.5, that is , each neighborhood consists of 20% or 50% of the observations.

Kernel regression

$$K_h(x_i, x_0) = K(\frac{x_i - x_0}{h}),$$

where $K(\cdot)$ is the *kernel*, and h is the *bandwidth*.

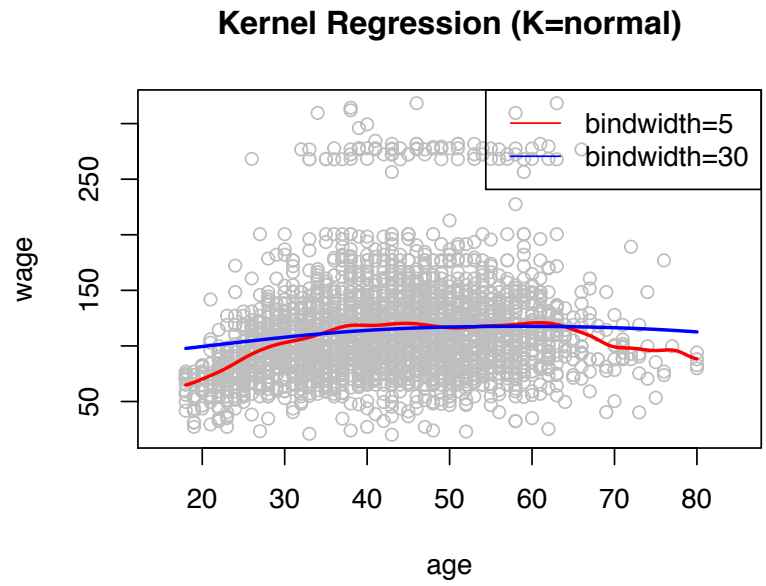

Figure 11: Kernel regression using the normal distribution with bandwidth 5 and 30

Pros and Cons of Local Regression

- Such *varying coefficient models* are a useful way of adapting a model to the most recently gathered data.

- Local regression can perform poorly if p is larger than about 3 or 4 because there will generally be very few training observations close to x_0.

Generalized Additive Models

Generalized additive models (GAMs) provide a general framework for extending a standard linear model by allowing non-linear functions of each of the variables, while maintaining *additivity*:

$$Y = \beta_0 + f_1(X_1) + f_2(X_2) + \cdots + f_p(X_p) + \epsilon.$$

The GAMs can be fitted via an approach known as *backfitting*, by repeatedly updating the fit for each predictor in turn, holding the others fixed.

```
library(gam)
par(mfrow = c(1, 3))
fit3 <- gam(wage ~ ns(year, 4) + ns(age, 5) +
    education, data = Wage)
plot(fit3, ylim = c(-30, 40), col = "darkblue",
    lwd = 2, se = TRUE)
```

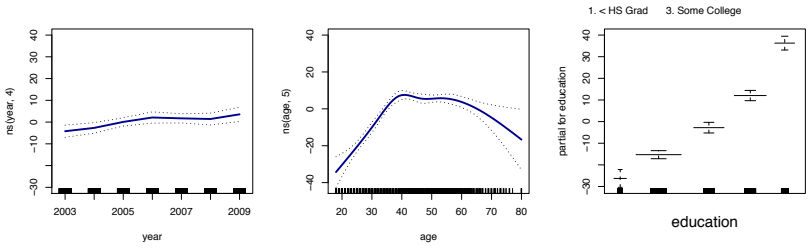

Figure 12: For the Wage data, plots of the relationship between each feature and the response, wage. Each plot displays the fitted function and pointwise standard errors. The first two functions are natural splines in year and age, with four and five degrees of freedom, respectively. The third function is a step function, fit to the qualitative variable education.

```
# we perform a series of ANOVA tests in order
# to determine which of these models is best:
# a GAM that excludes year, a GAM that uses a
# linear function of year, or a GAM that uses
# a spline function of year.
fit1 <- gam(wage ~ ns(age, 5) + education, data = Wage)
fit2 <- gam(wage ~ year + ns(age, 5) + education,
    data = Wage)
anova(fit1, fit2, fit3, test = "F")

## Analysis of Deviance Table
##
```

```
## Model 1: wage ~ ns(age, 5) + education
## Model 2: wage ~ year + ns(age, 5) + education
## Model 3: wage ~ ns(year, 4) + ns(age, 5) + education
##   Resid. Df Resid. Dev Df Deviance       F
## 1      2990    3712881
## 2      2989    3694885  1  17996.1 14.5551
## 3      2986    3691919  3   2966.4  0.7997
##      Pr(>F)
## 1
## 2 0.0001389 ***
## 3 0.4938916
## ---
## Signif. codes:
##   0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1  ' ' 1
```

```
par(mfrow = c(1, 3))
fit <- gam(I(wage > 250) ~ s(year, 5) + lo(age,
    span = 0.5, degree = 1) + education, data = Wage,
    family = "binomial")
plot(fit, ylim = c(-30, 40), col = "darkblue",
    lwd = 2, se = TRUE)
```

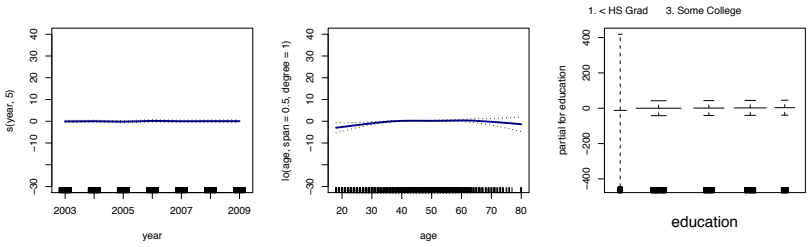

Figure 13: For the Wage data, the logistic regression GAM, is fit to the binary response $I(wage > 250)$. Each plot displays the fitted function and pointwise standard errors. The first function is a smoothing spline with 5 degrees of freedom in year, the second function a local linear regression with $span = 0.5$ in age, and the third a step function for education. There are very wide standard errors for the first level $< HS$ of education.

- `ploy()`: polynomials
- `bs()`: b-splines
- `ns()`: nature cubic splines
- `s()`: smoothing splines
- `lo()`: local regression fitting

Pros and Cons of GAMs

- GAMs allow 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 fit 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 is good.
- The smoothness of the function f_j for the variable X_j 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*.