Moving Beyond Linearity

1. Polynomial regression
2. Step functions
3. Regression splines
4. Smoothing spline
5. Local regression
6. Generalized additive Models

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, ...

- Data: $(\mathbf{x}_i, y_i), i = 1, ..., n$.
- The general model
$$y_i = f(\mathbf{x}_i) + \epsilon_i.$$
- We assumed f is linear before.
- Linearity assumption is almost always an approximation, and sometimes a poor one.
- What kind of functions of $f(\cdot)$ we should assume?

- Data: $(\mathbf{x}_i, y_i), i = 1, ..., n$.
- The general model

$$y_i = f(\mathbf{x}_i) + \epsilon_i.$$

- We assumed f is linear before.
- Linearity assumption is almost always an approximation, and sometimes a poor one.
- What kind of functions of $f(\cdot)$ we should assume?
- Cannot search for arbitrary function $f(\cdot)$.
- Limit the search space.
- by Polynomial functions, or step functions, or certain basis functions,...

x_i is 1-dimensional data

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

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

x_i is 1-dimensional data

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

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

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

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

x_i is 1-dimensional data

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

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

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

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

- This is a multiple linear regression model with d inputs: $(x_i, x_i^2, ..., x_i^d)$.
- All linear regression results apply.

x_i is 1-dimensional data

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

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

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

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

- This is a multiple linear regression model with d inputs: $(x_i, x_i^2, ..., x_i^d)$.
- All linear regression results apply.
- Problem: how to determine the appropriate degree d.
- it is unusual to use d greater than 3 or 4.

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.

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

Logistic model with polynomial regression

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

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

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

Basis functions

- Let $b_1(x), ..., b_p(x)$ 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:

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

Basis functions

- Let $b_1(x), ..., b_p(x)$ 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:

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

- The model is

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

Basis functions

- Let $b_1(x), ..., b_p(x)$ 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:

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

- The model is

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

- Again a multiple linear regression model.
- The polynomial functions are special cases of basis functions approach.

Step functions (piecewise constant functions)

- Step functions are piece-wise constants.

Step functions (piecewise constant functions)

- Step functions are piece-wise constants.
- Continuous functions can be well approximated by step functions. A function can be approximated by step functions.

Step functions (piecewise constant functions)

- Step functions are piece-wise constants.
- Continuous functions can be well approximated by step functions. A function can be approximated by step functions.
- Create the cutpoints

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

- The entire real line is cut into $p + 1$ intervals.
- Set $c_k(x) = I(c_k \leq x < c_{k+1})$, for $k = 0, ..., p$.
- Any x must be in exactly one of the $p + 1$ intervals.

$$c_0(x) + \cdots + c_p(x) = 1, \ \forall \ x.$$

Regression model based on step functions

- Use linear combination of $c_k(x)$ to approximate functions.

$$f(x) = \beta_0 + \beta_1 c_1(x) + \cdots + \beta_p c_p(x).$$

$$y_i = \beta_0 + \beta_1 c_1(x_i) + \ldots + \beta_p c_p(x_i) + \epsilon_i, \ i = 1, \cdots, n.$$

Regression model based on step functions

- Use linear combination of $c_k(x)$ to approximate functions.

$$f(x) = \beta_0 + \beta_1 c_1(x) + \cdots + \beta_p c_p(x).$$

$$y_i = \beta_0 + \beta_1 c_1(x_i) + \ldots + \beta_p c_p(x_i) + \epsilon_i, \ i = 1, \cdots, n.$$

- Again a multiple linear regression model.

Regression model based on step functions

- Use linear combination of $c_k(x)$ to approximate functions.

$$f(x) = \beta_0 + \beta_1 c_1(x) + \cdots + \beta_p c_p(x).$$

$$y_i = \beta_0 + \beta_1 c_1(x_i) + \ldots + \beta_p c_p(x_i) + \epsilon_i, \ i = 1, \cdots, n.$$

- Again a multiple linear regression model.
- Same extension works for generalized linear model

$$f(x) = g(\beta_0 + \beta_1 c_1(x) + \cdots + \beta_p c_p(x)).$$

Piecewise polynomial functions

1. Cut the entire real line (or the range of values of covariates) into sub-intervals same as step function approach.
   Create the cutpoints

$$-\infty = c_0 < c_1 < ... < c_d < c_{d+1} = \infty.$$

   The entire real line is cut into $d + 1$ intervals.
2. Use a polynomial function on each sub-interval.

$$p_k(x) = \begin{cases} \text{a polynomial}, & x \in [c_k, c_{k+1}), \\ 0, & \text{others.} \end{cases}$$

Piecewise polynomial functions

1. Cut the entire real line (or the range of values of covariates) into sub-intervals same as step function approach.
   Create the cutpoints

$$-\infty = c_0 < c_1 < \ldots < c_d < c_{d+1} = \infty.$$

   The entire real line is cut into $d + 1$ intervals.
2. Use a polynomial function on each sub-interval.

$$p_k(x) = \begin{cases} \text{a polynomial}, & x \in [c_k, c_{k+1}), \\ 0, & \text{others.} \end{cases}$$

- These cutpoints are called *knots*.
- Step function approach is a special case of piecewise polynomial of degree 0.

- $f(x) = \beta_0 p_0(x) + \cdots \beta_d p_d(x)$.
- The model is

$$y_i = \beta_0 p(x_i) + \beta_1 p_1(x_i) + ... + \beta_d p_d(x_i) + \epsilon_i.$$

- $f(x) = \beta_0 p_0(x) + \cdots \beta_d p_d(x)$.
- The model is

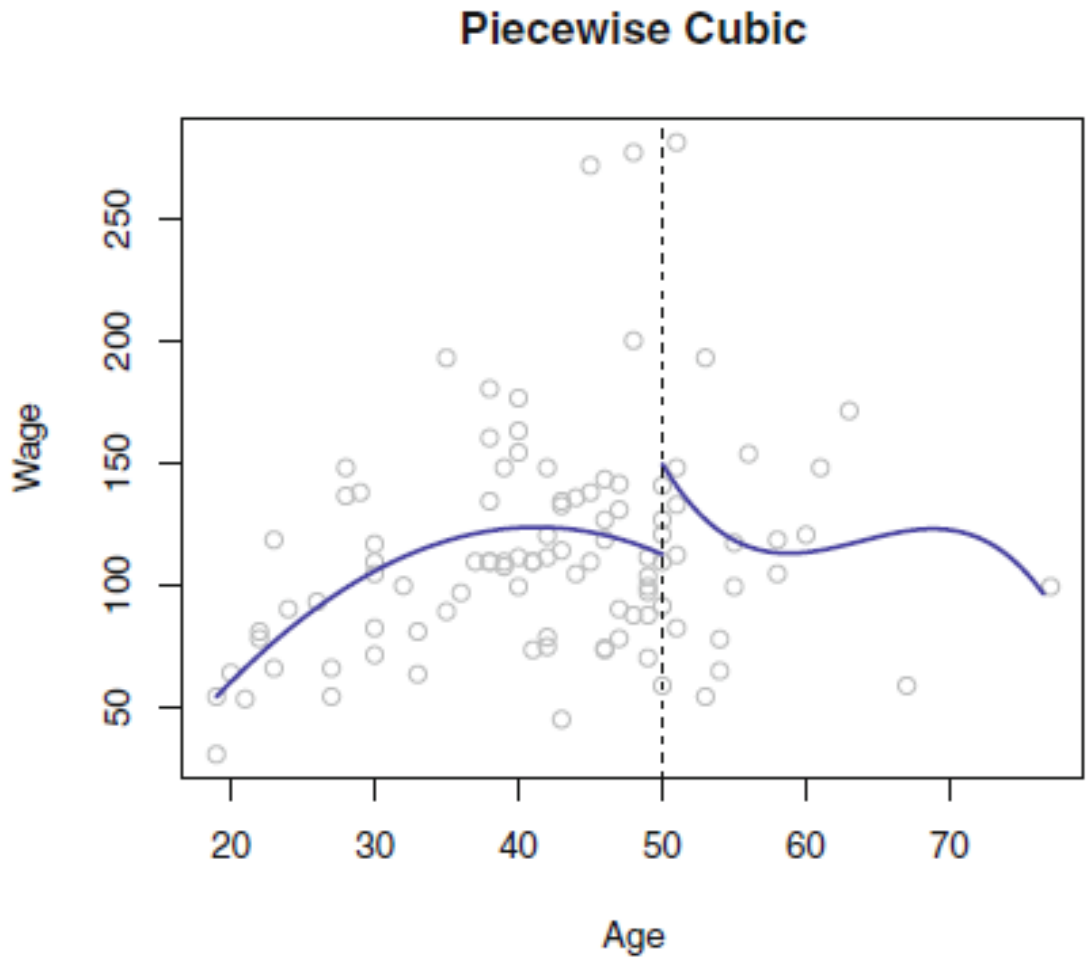
$$y_i = \beta_0 p(x_i) + \beta_1 p_1(x_i) + ... + \beta_d p_d(x_i) + \epsilon_i.$$

- Still a multiple linear regression model.

- $f(x) = \beta_0 p_0(x) + \cdots \beta_d p_d(x)$.
- The model is

$$y_i = \beta_0 p(x_i) + \beta_1 p_1(x_i) + ... + \beta_d p_d(x_i) + \epsilon_i.$$

- Still a multiple linear regression model.
- Difficulty in creating the number and locations of cutpoints
- Drawback: non-smooth, not even continuous.



- Advantage:
  - capture local variation;
  - the degree of polynomial is generally low.
- disadvantage:
  - dis-continuity at knots.

Constraining the piecewise polynomial

When fit the least squares, one can add constriants to the least squares minimization.

Constraining the piecewise polynomial

When fit the least squares, one can add constriants to the least squares minimization.

- The constraints can be such that the piecewise polynomial is forced to be continuous at knots.

Constraining the piecewise polynomial

When fit the least squares, one can add constriants 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 differentiale at knots with continuous first derivatives.

Constraining the piecewise polynomial

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

Constraining the piecewise polynomial

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

The effect of constraints

- Each constraint can be expressed as on 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,
  - have continuous derivatives up to order $d-1$ at knots.

Spline functions

- *Spline functions* of degree d are
  - piecewise polynomial functions of degree d,
  - have continuous derivatives up to order $d-1$ at knots.
- Cubic spline: piecewise cubic polynomials but are continuous and have continous 1st and 2nd derivatives at knots.

Spline functions

- *Spline functions* of degree d are
  - piecewise polynomial functions of degree d,
  - have continuous derivatives up to order $d-1$ at knots.
- Cubic spline: piecewise cubic polynomials but are continuous and have continous 1st and 2nd 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 $\xi_1, ..., \xi_K$.

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

$$y_i = \beta_0 + \eta_1 b_1(x_i) + ... + \beta_K b_{K+3}(x_i) + \epsilon_i.$$

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

$$y_i = \beta_0 + \eta_1 b_1(x_i) + ... + \beta_K b_{K+3}(x_i) + \epsilon_i.$$

- How to find these basis functions $b_k(x)$?
- **Require**: each must be a polynomial of order 3 and must be continuous, continous 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

- The behavior of the cubic spline at boundary can be quite unstable.
- *Natural cubic spline* is cubic spline but require the function to be *linear* on $(-\infty, \xi_1]$ and $[\xi_K, \infty)$.

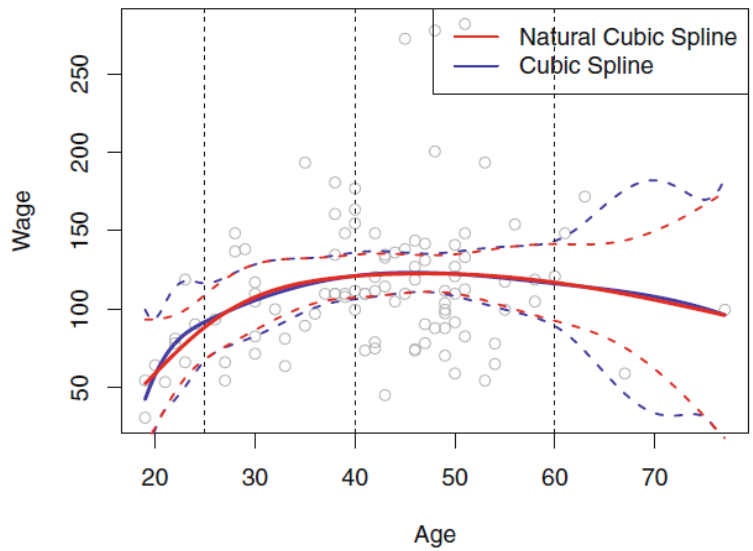


Figure: 7.4. A cubic spline and a natural cubic spline, with three knots, fit to a subset of the Wage data. Natural spline has narrower confidence intervals near boundary

- With further restriction near boundary, natural spline regression generally behaves better than cubic spline regression.

- With further restriction near boundary, natural spline regression generally behaves better than cubic spline regression.
- Degree of freedom of natural spline with K knots is $K + 4 - 4 = K$, but excluding the constant (absorbed in intercept), we usually call it $K - 1$ degree of freedom.

- With further restriction near boundary, natural spline regression generally behaves better than cubic spline regression.
- Degree of freedom of natural spline with K knots is $K + 4 - 4 = K$, 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.

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.

Comparison with Polynomial Regression

- Regression splines often give superior results to polynomial regression.

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.

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 flexibity by increased order of power function, which can be dangerously inapproximate for moderately large or small X in absolute value.

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 flexibity 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.

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 flexibity 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.

Smoothing spline

Review of regression splines

- specifying a set of knots,
- producing a sequence of basis functions,
- using least squares to estimate the spline coefficients.

Smoothing spline

Review of regression splines

- specifying a set of knots,
- producing a sequence of basis functions,
- using least squares to estimate the spline coefficients.

Find some function, say $f(x)$, that fits the observed data well (x_i, y_i).

- With to minimize

$$\text{RSS} = \sum_{i=1}^{n} (y_i - f(x_i))^2.$$

  - No constrains on $f(\cdot)$, interpolation will give 0 RSS.

Smoothing spline

Review of regression splines

- specifying a set of knots,
- producing a sequence of basis functions,
- using least squares to estimate the spline coefficients.

Find some function, say $f(x)$, that fits the observed data well (x_i, y_i).

- With to minimize

$$\text{RSS} = \sum_{i=1}^{n} (y_i - f(x_i))^2.$$

- No constrains on $f(\cdot)$, interpolation will give 0 RSS.
- We need certain smoothness constraints on $f(\cdot)$.

Smoothing spline

Review of regression splines

- specifying a set of knots,
- producing a sequence of basis functions,
- using least squares to estimate the spline coefficients.

Find some function, say $f(x)$, that fits the observed data well (x_i, y_i).

- With to minimize

$$\text{RSS} = \sum_{i=1}^{n} (y_i - f(x_i))^2.$$

  - No constrains on $f(\cdot)$, interpolation will give 0 RSS.
  - We need certain smoothness constraints on $f(\cdot)$.
- The most common constraint is f'', the second derivative do not vary much.

Smoothing spline

- The first derivative $f'(t)$ measures the slope of a function at t,

Smoothing spline

- The first derivative $f'(t)$ measures the slope of a function at t,
- 2nd derivative corresponds to amount that the slope is changing.

Smoothing spline

- The first derivative $f'(t)$ measures the slope of a function at t,
- 2nd derivative corresponds to amount that the slope is changing.
- the second derivative of a function is a measure of its roughness:

Smoothing spline

- The first derivative $f'(t)$ measures the slope of a function at t,
- 2nd derivative corresponds to amount that the slope is changing.
- the second derivative of a function is a measure of its roughness:
  - it is large in absolute value if $f(t)$ is very wiggly near t,
  - it is close to zero otherwise.

Smoothing spline

- The first derivative $f'(t)$ measures the slope of a function at t,
- 2nd derivative corresponds to amount that the slope is changing.
- the second derivative of a function is a measure of its roughness:
  - it is large in absolute value if $f(t)$ is very wiggly near t,
  - it is close to zero otherwise.
- A natural choice is: minimizng

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

Smoothing spline

- The first derivative $f'(t)$ measures the slope of a function at t,
- 2nd derivative corresponds to amount that the slope is changing.
- the second derivative of a function is a measure of its roughness:
  - it is large in absolute value if $f(t)$ is very wiggly near t,
  - it is close to zero otherwise.
- A natural choice is: minimizng

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

- This is equivalent to with λ being the tuning parameter

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

- The function f minimizing the above is called *smoothing spline.*
- The function that minimize that loss+roughness penalty is *skrunken* version of a natural cubic spline with knots $x_1, ..., x_n$.

The tuning parameter

λ controls the amount of roughness penalty.

- $\lambda = 0$: no penalty, exactly interpolate the training observations, degree of freedom $= n$; maybe overfit.

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

The tuning parameter

λ controls the amount of roughness penalty.

- $\lambda = 0$: no penalty, exactly interpolate the training observations, degree of freedom $= n$; maybe overfit.

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

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

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

The tuning parameter

λ controls the amount of roughness penalty.

- $\lambda = 0$: no penalty, exactly interpolate the training observations, degree of freedom $= n$; maybe 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?

The tuning parameter

λ controls the amount of roughness penalty.

- $\lambda = 0$: no penalty, exactly interpolate the training observations, degree of freedom $= n$; maybe 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 spline
  - the higher it is, the more flexible.
  - the lower-bias but higher-variance.

Effective degree of freedom

- The df_λ is a measure of the flexibility of the smoothing spline
  - the higher it is, the more flexible.
  - the lower-bias but higher-variance.
- 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.
- Then, the effective degree of freedom is

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

Choice of λ

- By cross validation.

Choice of λ

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

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

  where $s_{\lambda,ii}$ is the i-th diagonal element of $\mathbf{S}_\lambda$.
- $\hat{f}_\lambda^{(-i)}(x_i)$ – fitted value for this smoothing spline evaluated at x_i, where the fit uses all of the training observations except for the ith observation (x_i, y_i).
- $\hat{f}_\lambda(x_i)$ – the smoothing spline function fit to all of the training observations and evaluated at x_i.

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 $y_1, \ldots, y_N$, 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 $\hat{\mathbf{Y}} = \mathbf{X}\hat{\boldsymbol{\beta}} = \mathbf{HY}$. If the diagonal values of $\mathbf{H}$ are denoted by $h_1, \ldots, h_N$, then the leave-one-oout 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

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

Local regression

- 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, $k(t)$.
  - $k(t)$ is nonnegative symmetric,
  - becomes small when $|t|$ is large.

Typical choice of kernels

- Uniform kernel: $k(t) = 1/2 I(|t| \leq 1)$.
- 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: $k(t) = 2/(\pi(e^t + e^{-t}))$.

Local view

- 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 $\mathbf{y}$, and there has expression of the form of (7.12).
- The degree of freedom controled 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.

Generalized additive Models (GAMs)

- 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 varible.
- This leads to the generalized additive model (GAM).

The GAMs

- 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

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

- 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.

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 $f_{jk}(X_j, X_k)$.

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:

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