STA 314: Statistical Methods for Machine Learning I

Lecture 5 - More on regularized linear regression and gradient descent

Xin Bing

Department of Statistical Sciences
University of Toronto

Review: why consider alternatives to the OLS estimator?

Recall the linear model is

$$Y = \beta_0 + \beta_1 X_1 + \ldots + \beta_p X_p + \epsilon.$$

Alternative fitting procedures to OLS could yield **better prediction accuracy** and **model interpretability**.

- Prediction: OLS estimator has large variance when p is large. Especially, if $p > n$, then OLS estimator is not unique and its variance is very large.
- Interpretability: By removing irrelevant features – that is, by setting some coefficient estimates to zero – we can obtain a model that is more parsimonious hence more interpretable.

Review

- Best subset selection
  - Great! But computationally unaffordable (choose from 2^p models)!
- Stepwise subset selection
  - Forward stepwise selection
  - Backward stepwise selection
  - Computationally affordable, but greedy approaches
- Are there better alternatives?
  - Shrinkage Methods! In particular, the Lasso.

Magic of the Lasso

Why does the lasso, unlike ridge regression, yield coefficient estimates that have exact zero?

Another Formulation for Ridge Regression and Lasso

The lasso and ridge regression coefficient estimates solve the problems

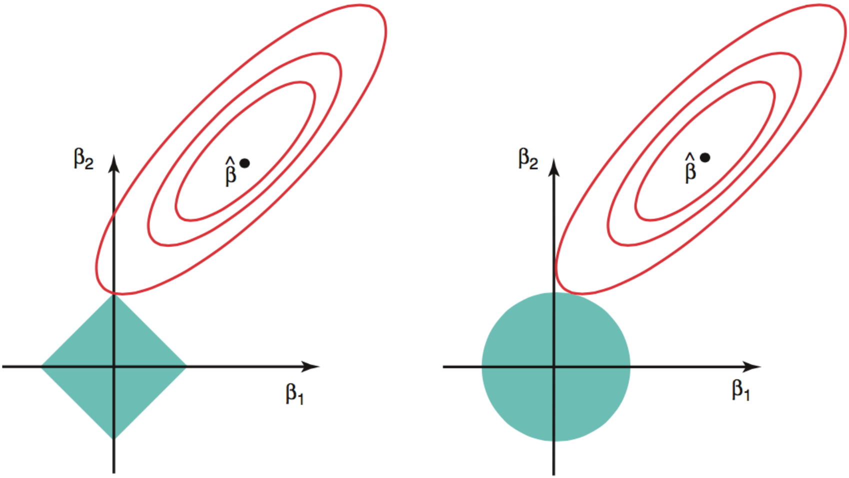
$$\underset{\beta}{\text{minimize}} \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2 \quad \text{subject to} \quad \sum_{j=1}^{p} |\beta_j| \leq s$$

and

$$\underset{\beta}{\text{minimize}} \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2 \quad \text{subject to} \quad \sum_{j=1}^{p} \beta_j^2 \leq s,$$

Here $s \geq 0$ is some regularization parameter (connected with the original λ).

Understand why the Lasso yields zero estimates

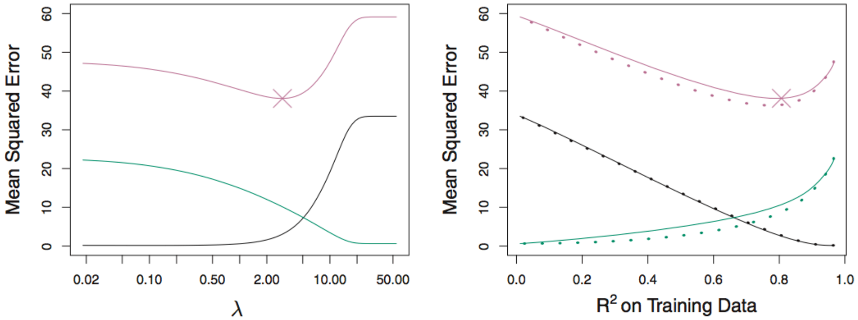


The solid areas are the constraint regions, $|\beta_1| + |\beta_2| \leq s$ and $\beta_1^2 + \beta_2^2 \leq s$, while the red ellipses are the contours of the RSS.

Lasso vs Ridge

- The ability of yielding a **sparse** model is a huge advantage of Lasso comparing to Ridge.
- A more sparse model means more interpretability!
- What about their prediction performance?

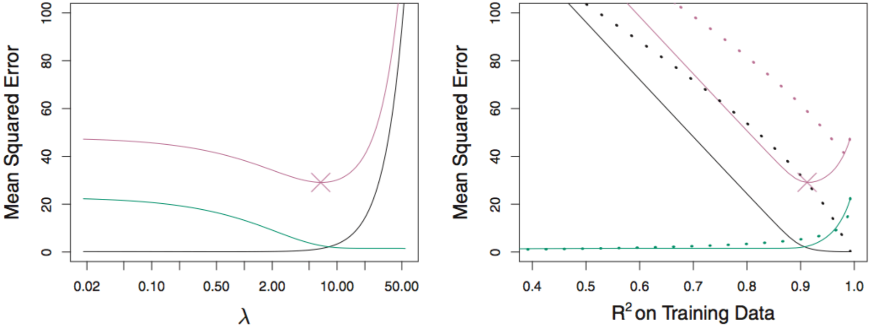
Comparing the MSE of Lasso and Ridge



Left: Plots of squared bias (black), variance (green), and test MSE (purple) for the lasso on a simulated data set.
Right: Comparison of squared bias, variance and test MSE between lasso (solid) and ridge (dotted). Both are plotted against their R^2 on the training data, as a common form of indexing. The crosses in both plots indicate the lasso model for which the MSE is smallest.

- *When the true coefficients are non-sparse, ridge and lasso have the same bias but ridge has a smaller variance hence a smaller MSE.*

Another Case



- *When the true coefficients are sparse, Lasso outperforms ridge regression of having both a smaller bias and a smaller variance.*

Conclusions on Lasso relative to Ridge

- These two examples illustrate that neither ridge regression nor the lasso will universally dominate the other.
- In general, one might expect the lasso to perform better when the response is only related with a relatively small number of predictors.
- As the ridge regression, when the OLS estimates have excessively high variance, the lasso solution can yield a reduction in variance at the expense of a small increase in bias, and consequently can lead to more accurate predictions.
- Unlike ridge regression, the lasso performs variable selection, and hence yields models that are easier to interpret.

A simple example of the shrinkage effects of ridge and lasso

- Assume that $n = p$ and $\mathbf{X} = \mathbf{I}_n$. We force the intercept term $\beta_0 = 0$.
- The OLS approach is to find $\beta_1, \ldots, \beta_p$ that minimize

$$\sum_{j=1}^{p} (y_j - \beta_j)^2.$$

This gives the OLS estimator

$$\hat{\beta}_j = y_j, \qquad \forall j \in \{1, \ldots, p\}.$$

The ridge estimator

- The ridge regression is to find $\beta_1, \dots, \beta_p$ that minimize

$$\sum_{j=1}^{p} (y_j - \beta_j)^2 + \lambda \sum_{j=1}^{p} \beta_j^2.$$

This leads to the ridge estimator

$$\hat{\beta}_j^R = \frac{y_j}{1+\lambda}, \qquad \forall j \in \{1, \dots, p\}.$$

Since $\lambda \geq 0$, the magnitude of each estimated coefficient is shrinked toward 0.

The lasso estimator

- The lasso is to find $\beta_1, \ldots, \beta_p$ that minimize

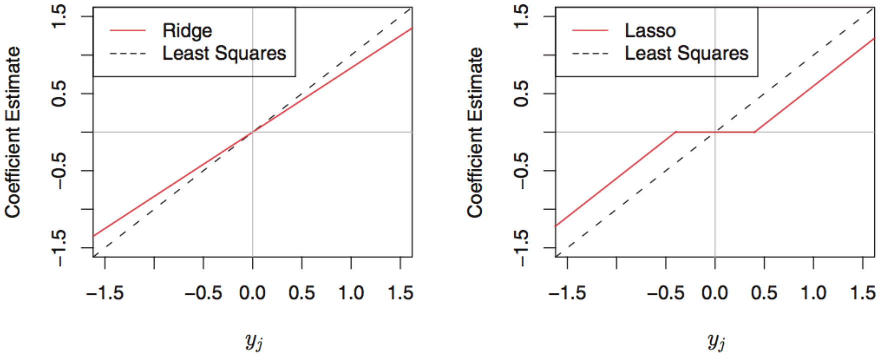
$$\sum_{j=1}^{p} (y_j - \beta_j)^2 + \lambda \sum_{j=1}^{p} |\beta_j|.$$

This gives estimator

$$\hat{\beta}_j^L = \begin{cases} y_j - \lambda/2 & \text{if } y_j > \lambda/2; \\ y_j + \lambda/2 & \text{if } y_j < -\lambda/2; \\ 0 & \text{if } |y_j| \leq \lambda/2. \end{cases}$$

The estimated coefficients from Lasso are also shrinked. The above shrinkage is known as the **soft-thresholding**.

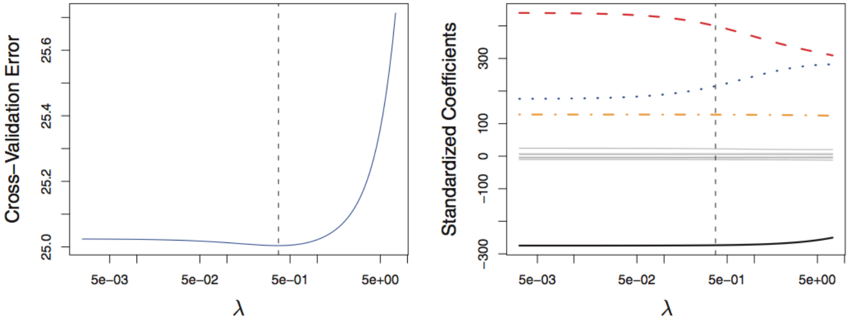
An illustrative figure



Selecting the Tuning Parameter

- Similar as the subset selection, for ridge and lasso, we require a systematic way of choosing the best model under a sequence of fitted models (from different choices of λ)
- Equivalently, we require a method to select the optimal value of the tuning parameter λ or equivalently, the value of the constraint s.
- Cross-validation provides a simple way to tackle this problem. We choose a grid of λ, and compute the cross-validation error rate for each value of λ.
- We then select the value of tuning parameter for which the cross-validation error is smallest.
- Finally, the model is re-fitted by using all of the available observations and the selected value of the tuning parameter.

Credit Card Data Example



Cross-validation errors that result from applying ridge regression to the Credit data set for various choices of λ.

More choices of penalties

- There are many other penalties in addition to the ℓ_2 and ℓ_1 norms used by ridge and lasso.
  - the elastic net:

$$\underset{\boldsymbol{\beta}}{\operatorname{argmin}} \; ||\mathbf{y} - \mathbf{X}\beta||_2^2 + \lambda\left[(1-\alpha)||\boldsymbol{\beta}||_1 + \alpha||\boldsymbol{\beta}||_2\right]$$

    for some tuning parameters $\lambda \geq 0$ and $\alpha \in [0, 1]$. Ridge corresponds to $\alpha = 1$ while lasso corresponds to $\alpha = 0$.

The group lasso

- If we suspect the model is nonlinear in X_1 or X_2, we can add quadratic terms, say

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_1^2 + \beta_3 X_2 + \beta_4 X_2^2 + \epsilon.$$

The **group lasso** estimator minimizes

$$RSS + \lambda \left(\sqrt{\beta_1^2 + \beta_2^2} + \sqrt{\beta_3^2 + \beta_4^2} \right).$$

In this penalty, we view β_1 and β_2 (coefficient of X_1 and X_1^2) as if they belong to the same group. The group Lasso can shrink the parameters in the same group (both β_1 and β_2) exactly to 0 simultaneously.

- There are a lot more penalties out there ......

Regularization in more general settings

- The ridge and lasso regressions are not restricted to the linear models.
- The idea of penalization is generally applicable to almost all parametric models.

$$\hat{\boldsymbol{\beta}} = \underset{\boldsymbol{\beta}}{\operatorname{argmin}} \; \underbrace{L(\boldsymbol{\beta}, \mathcal{D}^{train}) + Pen(\boldsymbol{\beta})}_{g(\boldsymbol{\beta};\mathcal{D}^{train})}.$$

  - OLS: $L(\boldsymbol{\beta}, \mathcal{D}^{train}) = ||\mathbf{y} - \mathbf{X}\beta||_2^2$, $Pen(\boldsymbol{\beta}) = 0$.
  - Ridge: $L(\boldsymbol{\beta}, \mathcal{D}^{train}) = ||\mathbf{y} - \mathbf{X}\beta||_2^2$, $Pen(\boldsymbol{\beta}) = ||\beta||_2^2$.
  - Lasso: $L(\boldsymbol{\beta}, \mathcal{D}^{train}) = ||\mathbf{y} - \mathbf{X}\beta||_2^2$, $Pen(\boldsymbol{\beta}) = ||\beta||_1$.
  - In general,
    - L can be any loss function, i.e. negative likelihood, 0-1 loss.
    - *Pen* could be any penalty function.

Moving Beyond Linearity

The linearity assumption in the feature space is almost always an approximation, and sometimes a poor one.

We consider the following extensions to relax the linearity assumption.

- Univariate case ($p = 1$):
  - Polynomial regression
  - Step functions
  - Regression splines
- Multivariate case ($p > 1$):
  - Local regression
  - Generalized additive models

Polynomial Regression

- The **polynomial regression**

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

  where ϵ_i is the error term and $x_i \in \mathbb{R}$.

- Can be fitted by the OLS approach.
- Coefficients themselves are not interpretable; we are more interested in the trend of the fitted function

$$\hat{f}(x_0) = \hat{\beta}_0 + \hat{\beta}_1 x_0 + \hat{\beta}_2 x_0^2 + \cdots + \hat{\beta}_d x_0^d.$$

- The degree d in practice is typically no greater than 4, and can be chosen via cross-validation.

Polynomial Regression

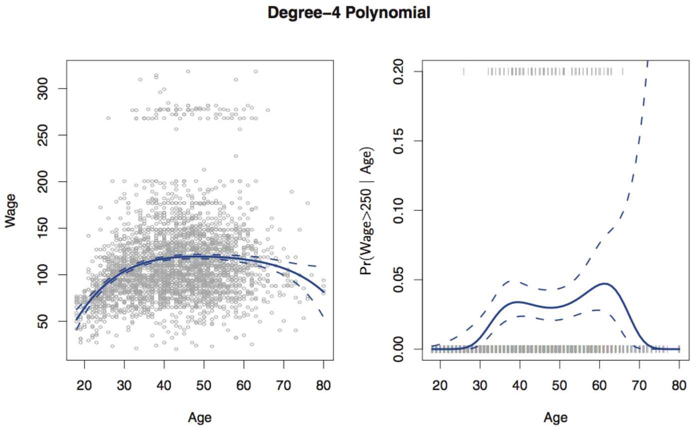
- The polynomial regression can be used for classification as well.
  - For instance, in the logistic regression,

$$\operatorname{logit}\left(\mathbb{P}(Y_i = 1 \mid X_i = x_i)\right) = \beta_0 + \beta_1 x_i + \beta_2 x_i^2 + \cdots + \beta_d x_i^d.$$

  - Can be fit by maximizing the likelihood.

- However, polynomials have notorious tail behavior – very bad for extrapolation.

The Wage Data



Left: The solid blue curve is a degree-4 polynomial of wage as a function of age, fit by the OLS. The dotted curves are estimated 95 % confidence intervals.

Right: We model the binary event 1{wage > 250} using logistic regression, with a degree-4 polynomial.

Step Functions

- The polynomial regression imposes a global structure on the non-linearity of X.
- The **step function** approach avoids such a global structure by breaking the range of X into bins.
- For pre-specified K cut points $c_1, \ldots, c_K$, define

$$
\begin{aligned}
C_0(X) &= 1\{X < c_1\}, \\
C_1(X) &= 1\{c_1 \leq X < c_2\}, \\
&\vdots \\
C_K(X) &= 1\{c_K \leq X\}.
\end{aligned}
$$

$C_0(X), \ldots, C_K(X)$ are in fact $(K+1)$ dummy variables, and they sum up to 1.

Step Functions

- Step function approach assumes

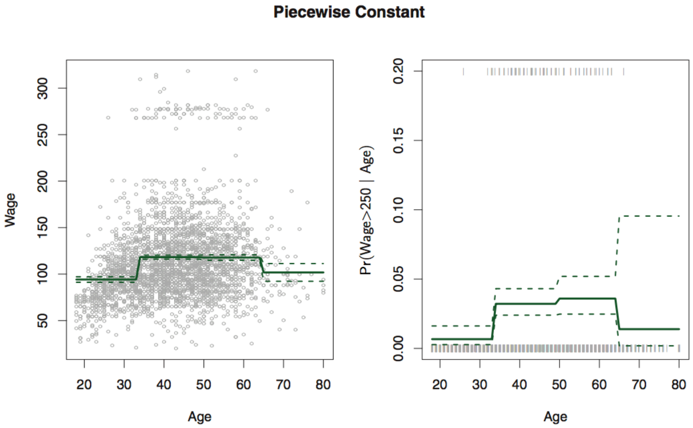
$$y_i = \beta_0 + \beta_1 C_1(x_i) + \beta_2 C_2(x_i) + ... + \beta_K C_K(x_i) + \epsilon_i,$$

  where ϵ_i is the error term. (Note we don't need $C_0(x_i)$ in the model when we also have the intercept term β_0.)

- Can be fitted by the OLS.

- β_j represents the average change in the response Y for $c_j \leq X < c_{j+1}$ relative to $X < c_1$.

The Wage Data



Left: The solid blue curve is a step function of wage 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, with the step function.

Pros and Cons of Step Function

- The step function approach is widely used in biostatistics and epidemiology among other areas, because the model is easy to fit and the regression coefficient has a natural interpretation.
- However, piecewise-constant functions can miss the trend of the true relationship between Y and X. The choice of cut points can be difficult to specify.
- How about combining polynomial and step function?

Piecewise Polynomials

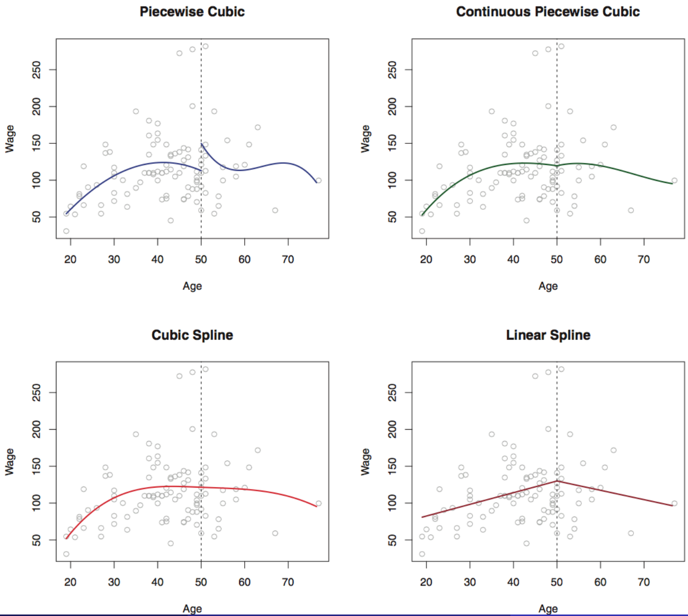
- Instead of a single polynomial in X over its whole domain, we can use different polynomials in different regions:

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

- The cut point c is called **knot**. Using more knots leads to a more flexible piecewise polynomial.

- In general, if we place K different knots throughout the range of X, then we will end up fitting $(K+1)$ different cubic polynomials.

The Wage Data



Regression splines

- Better to add constraints to polynomials at the knots for:
  - continuity: equal function values
  - smoothness: equal first and second order derivatives
  - higher order derivatives
- The constrained polynomials are called **splines**. A degree-d spline contains piecewise degree-d polynomials, with continuity in derivatives up to degree $(d-1)$ at each knot.
- How can we construct the degree-d spline?

Linear Splines

- A **linear spline** has piecewise linear functions continuous at each knot. That is, with knots at $\xi_1 < \xi_2 < \cdots < \xi_K$,

$$y_i = \beta_0 + \beta_1 x_i + \beta_2 (x_i - \xi_1)_+ \cdots + \beta_{K+1}(x_i - \xi_K)_+ + \epsilon_i,$$

  where

$$(x_i - \xi_k)_+ = \begin{cases} x_i - \xi_k & \text{if } x_i > \xi_k \\ 0 & \text{otherwise} \end{cases}.$$

- A basis representation:

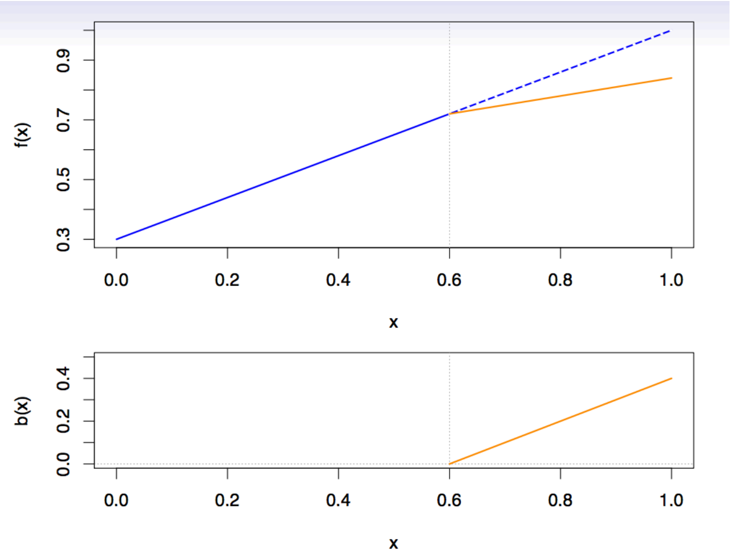
$$y_i = \beta_0 + \beta_1 b_1(x_i) + \beta_2 b_2(x_i) + \cdots + \beta_{K+1} b_{K+1}(x_i) + \epsilon_i,$$

  where b_k are basis functions

$$b_1(x_i) = x_i, \quad b_{k+1}(x_i) = (x_i - \xi_k)_+, \quad k = 1, \ldots, K,$$

- Interpretation of β_1: the averaged increase of Y associated with one unit of X for $X < \xi_1$.

Linear Splines



Cubic Splines

- A **cubic spline** has piecewise cubic polynomials with continuous derivatives up to order 2 at each knot. That is, with K knots at $\xi_1 < \xi_2 < \cdots < \xi_K$,

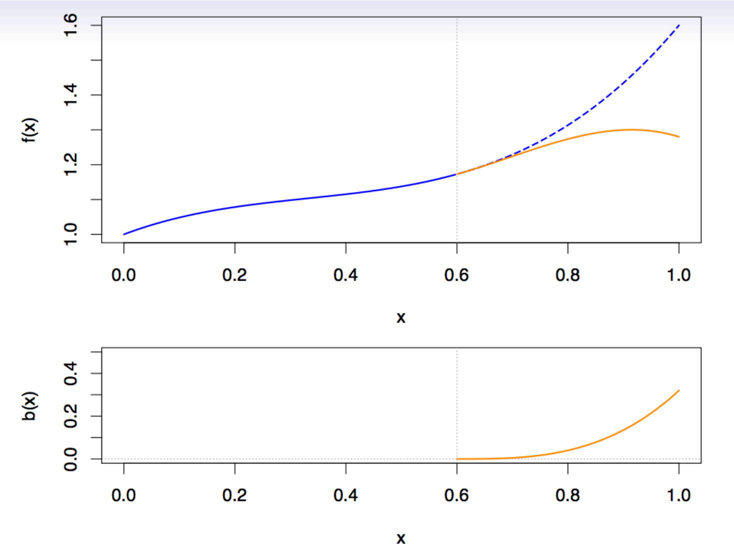
$$y_i = \beta_0 + \beta_1 b_1(x_i) + \beta_2 b_2(x_i) + \ldots + \beta_{K+3} b_{K+3}(x_i) + \epsilon_i,$$

where b_k are basis functions

$$b_1(x_i) = x_i, \quad b_2(x_i) = x_i^2, \quad b_3(x_i) = x_i^3,$$

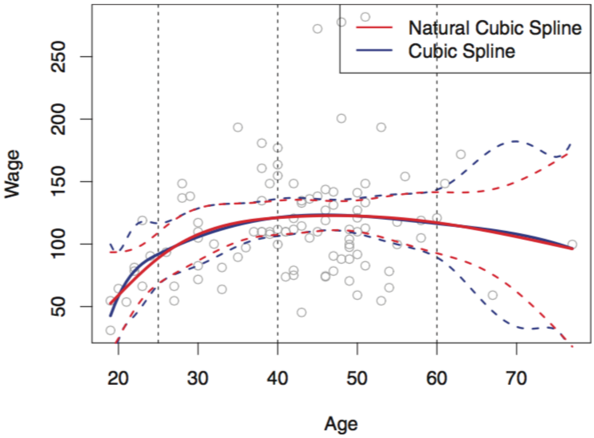
$$b_{k+3}(x_i) = (x_i - \xi_k)_+^3, \quad k = 1, \ldots, K.$$

Cubic Splines



Natural Splines

A natural spline is a regression spline with additional boundary constraints: the function is required to be linear at the boundary.



More on splines

- Choosing the number and locations of the knots
  - Typically, we place K knots at certain quantiles of the data or place on the range of X with equal space. Oftentimes, the placement of knots is not very crucial.
  - We use cross-validation to choose K.
- Polynomial regressions and step functions are special cases of splines.
- Another variant: smoothing spline (ISLR 7.5).

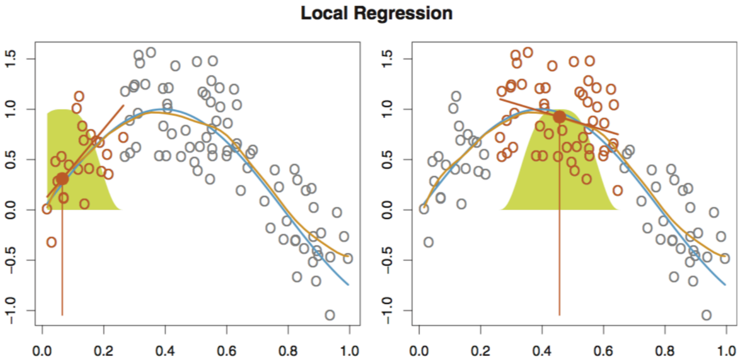
Local Regression

Local regression predicts at a target point x_0 using only the nearby training observations.

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 $\hat{f}(x_0) = \hat{\beta}_0 + \hat{\beta}_1 x_0$.

Simulated Example



The blue curve is true $f(x)$, and the light orange curve is the local regression $\hat{f}(x)$. The orange points are local to the target point x_0, represented by the orange vertical line. The yellow bell-shape indicates weights assigned to each 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 $\hat{f}(x_0)$.

Local Regression

- The size of the neighborhood (fraction s of training data) is a tuning parameter, which can be chosen by cross-validation.
- The weight of each point in the neighborhood needs to be specified.
- When we have two dimensional predictors X_1 and X_2, we can simply use 2-dimensional neighborhoods, and fit bivariate linear regression models using the observations that are near each target point in 2-dimensional space.
- However, local regression can perform poorly if p is much larger than about 3 or 4 (the curse of dimensionality).
- k-Nearest Neighbour is one of the most common local regression approaches. It corresponds to $K(x_i, x_0) = \frac{1}{k}$ and $\beta_1 = 0$.

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,

$$y_i = \beta_0 + f_1(x_{i1}) + f_2(x_{i2}) + \cdots + f_p(x_{ip}) + \epsilon_i.$$

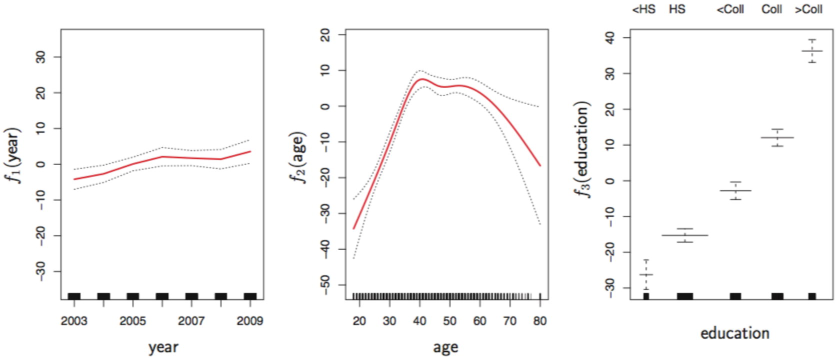
- Each f_k can be linear, polynomials, step function, splines and local regression.
- Can be applied to classification problems.
  - Logistic regression:

$$\text{logit}\left(\mathbb{P}(Y_i = 1 \mid X_i = x_i)\right) = \beta_0 + f_1(x_{i1}) + f_2(x_{i2}) + \cdots + f_p(x_{ip}).$$

Wage Data

Consider the wage data

$$\text{wage} = \beta_0 + f_1(\text{year}) + f_2(\text{age}) + f_3(\text{education}) + \epsilon.$$



The first two functions are natural splines in year and age. The third function is a step function, fit to the qualitative variable education.

Pros and Cons of GAMs

- GAMs allow us to fit a non-linear f_j to each X_j, so that we can automatically model non-linear relationships that standard linear regression won't be able to capture.
- The non-linear fit can potentially improve prediction accuracy.
- 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.
- It avoids the curse of dimensionality by assuming additivity.
- However, GAMs fail to incorporate the interaction of variables.