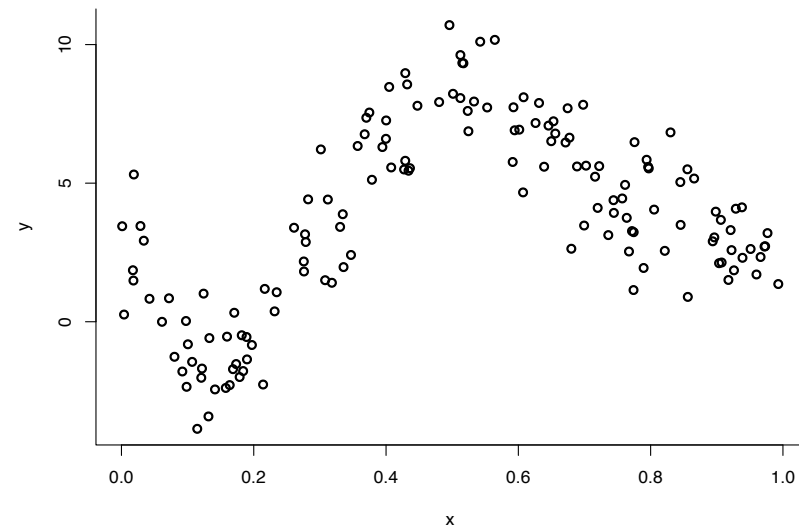


Advanced Data Analysis

Whiteboard Lecture

Automatic Nonparametric Regression



Recall the **nonparametric regression** setting where we observe ‘scatterplot’ data:

$$(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n).$$

The nonparametric regression model is:

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

where f is ‘smooth’.

Linear Smoothers

Let

$$\mathbf{y} = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix}.$$

Linear smoothers are those of the form

$$\hat{\mathbf{y}} = \mathbf{L}\mathbf{y}$$

where $\hat{\mathbf{y}}$ is the $n \times 1$ vector of fitted values.

\mathbf{L} ($n \times n$) is called the **smoother matrix**.

Example 1 Polynomial Regression.

$$y_i = \beta_0 + \beta_1 x_i + \beta_2 x_i^2 + \dots + \beta_p x_i^p + \varepsilon_i$$

Let

$$\mathbf{X}_p = \begin{bmatrix} 1 & x_1 & \dots & x_1^p \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_n & \dots & x_n^p \end{bmatrix}$$

and

$$\mathbf{P}_p = \mathbf{X}_p(\mathbf{X}_p^T \mathbf{X}_p)^{-1} \mathbf{X}_p^T.$$

Then

$$\hat{\mathbf{y}} = \mathbf{P}_p \mathbf{y}.$$

So polynomial regression is linear smoothing with $\mathbf{L} = \mathbf{P}_p$.

Example 2 Penalised Splines.

In the previous whiteboard lecture we had

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

where e.g.

$$\mathbf{S}_\lambda = \mathbf{X}_T (\mathbf{X}_T^T \mathbf{X}_T + \lambda \mathbf{D})^{-1} \mathbf{X}_T^T$$

with

$$\mathbf{X}_T = \begin{bmatrix} 1 & x_1 & (x_1 - \kappa_1)_+ & \dots & (x_1 - \kappa_K)_+ \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & x_n & (x_n - \kappa_1)_+ & \dots & (x_n - \kappa_K)_+ \end{bmatrix}.$$

These are linear smoothers with $L = \mathbf{S}_\lambda$.

("Effective") Degrees of Freedom

Consider a general linear smoother

$$\hat{\mathbf{y}} = \mathbf{L} \mathbf{y}$$

A widely accepted definition as a meaningful measure of effective degrees of freedom is

$$\text{df} = \text{tr}(\mathbf{L}).$$

For p th degree polynomials:

$$\begin{aligned} \text{df} &= \text{tr}(\mathbf{P}_p) \\ &= \text{tr}\{\mathbf{X}_p (\mathbf{X}_p^T \mathbf{X}_p)^{-1} \mathbf{X}_p^T\} \\ &= \text{tr}\{\mathbf{X}_p^T \mathbf{X}_p (\mathbf{X}_p^T \mathbf{X}_p)^{-1}\} \\ &= \text{tr}(\mathbf{I}_{p+1}) \\ &= p + 1 \end{aligned}$$

So

linear regression has $df=2$.

quadratic regression has $df=3$.

cubic regression has $df=4$.

For penalised splines

$$\begin{aligned}df_\lambda &= \text{tr}\{X_T(X_T^T X_T + \lambda D)^{-1} X_T^T\} \\ &= \text{tr}\{(X_T^T X_T + \lambda D)^{-1} X_T^T X_T\} \\ &= \text{monotonic function of } \lambda\end{aligned}$$

When using penalised splines in practice, we need to choose:

(1) spline basis functions,

(2) knots $(\kappa_1, \dots, \kappa_K)$,

(3) λ (or df_λ) (by far the most important!).

The general consensus on their choices are:

spline basis	cubic B-splines
knots	for most f can get by with $K \simeq 30$ and $\kappa_1, \dots, \kappa_K \simeq$ the quantiles of the x_i s
λ (df_λ)	Popular choices are: <ul style="list-style-type: none">• CV• GCV• REML• MCMC in Bayesian framework

Cross-Validation (CV) and Generalised Cross-Validation (GCV)

Data: $(x_1, y_1), \dots, (x_n, y_n)$

Model: $y_i = f(x_i) + \varepsilon_i$.

Family of estimators: $\{\hat{f}_\lambda(\cdot; \lambda) : \lambda > 0\}$

We first explain CV.

Let $n = 6$ to keep things simple.

i.e. Data are

$$(x_1, y_1), (x_2, y_2), \dots, (x_6, y_6).$$

(1) Set the value of λ . e.g. $\lambda = 0.05$.

(2) Fit estimator to data with (x_1, y_1) omitted.

Call this fit $\hat{f}_{-1}(\cdot; 0.05)$.

(3) Repeat (2) but omitting (x_2, y_2) , giving $\hat{f}_{-2}(\cdot; 0.05)$.

(4) Continue in this way until we get

$$\hat{f}_{-1}(\cdot; 0.05), \hat{f}_{-2}(\cdot; 0.05), \dots, \hat{f}_{-6}(\cdot; 0.05)$$

known as **leave-one-out** estimators.

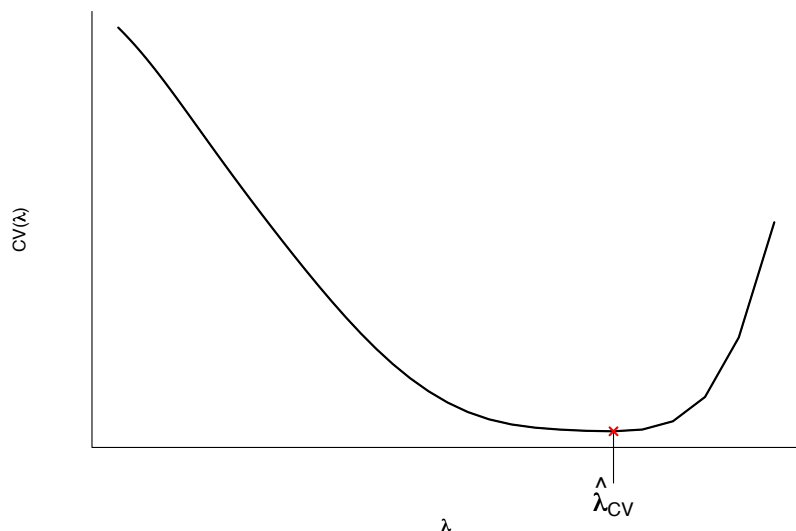
(5) The CV function at $\lambda = 0.05$ is

$$CV(0.05) = \sum_{i=1}^6 \{y_i - \hat{f}_{-i}(x_i; 0.05)\}^2.$$

The idea is to use each (x_i, y_i) to **cross-validate** the estimate with that datum omitted.

(6) Repeat all of the above over a grid of λ values.

This gives:



Question What is GCV?

Answer

GCV was invented in the early 1970s to avoid perceived computing problems with CV.

The genesis is the expression (not derived here) for CV:

$$CV(\lambda) = \sum_{i=1}^n \left(\frac{y_i - \hat{f}(x_i; \lambda)}{1 - S_{\lambda,ii}} \right)^2$$

where $S_{\lambda,ii}$ is the (i, i) entry of S_{λ} .

GCV says:

Replace $S_{\lambda,ii}$ with

$$\frac{1}{n} \sum_{i=1}^n S_{\lambda,ii} = \text{tr}(S_{\lambda})/n = df_{\lambda}/n.$$

This gives

$$\text{GCV}(\lambda) = \frac{\text{RSS}(\lambda)}{(1 - \text{df}_\lambda/n)^2}$$

By a Taylor series argument

$$\frac{1}{1 - \varepsilon} = 1 + \varepsilon + \varepsilon^2 + \dots$$

For large n this leads to the approximation

$$\text{GCV}(\lambda) \simeq \text{RSS}(\lambda)(1 + 2\text{df}_\lambda/n).$$

We see, from this expression, that minimising $\text{GCV}(\lambda)$ involves a **trade-off** between **over-fitting** (RSS(λ) small, df_λ big)

and

under-fitting (RSS(λ) big, df_λ small)

We can also use GCV in the polynomial regression context to choose the degree of polynomial.

