Quant III

Lab 11: Nonparametric Regression

Junlong Aaron Zhou

December 03, 2020

Nonparametric Method

- $Y_i = f(X_i) + \epsilon_i$
- Parametric case: $f(X_i) = X'\beta$, where X_i can contains non-linear transformation
- Some non-parametric approach: take average over bins $f(x) = \sum y_i \frac{\mathbb{I}(x \in B_j)}{\sum \mathbb{I}(x_i \in B_j)}$
- Improvement: local average $f(x) = \frac{1}{K} \sum_{x_i \in B_K} y_i$ where $B_K = \{|x_i - x| < M\} s.t. |B_K| = K$
- Kernel method: use kernel to weight each observation
- Intuition: X_i close to X should be put more weight

Kernel

- Kernel function: $K(Z)$: Gaussian Kernel, uniform kernel, triangular kernel, etc
- Z is a distance measure
- Kernel as weight: $w_i = K(\frac{x - x_i}{h})$
- Kernel: $f(x) = \frac{\sum y_i w_i}{\sum w_i}$
- We can do better than calculating weighted mean

Local Polynomial Regression

- One idea: given a data point x, use the data point around x to run a linear regression (similar to local average)
- Cleveland (1979) first proposed the local regression smoother, using kernel.
- Local linear regression and local polynomial regression are similar

$$\arg\min_{\{\alpha_m\}_{m=1}^{M}} \sum_{i=1}^{N} \left(y_i - \alpha_0 - \sum_{m=1}^{M} \alpha_m x_i^m \right)^2 K\left(\frac{x - x_i}{h} \right)$$

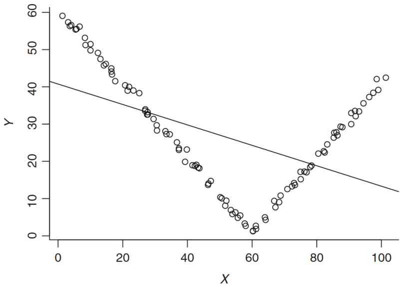
- For example, RDD uses local linear regression in the same manner.
- Easy to implement in R (later)
- Bandwidth selection is an issue (plug in or cross-validation)

Beyond LPR

- Why not just LPR?
- Splines have an analytic foundation that is superior to that of local regression, as one can prove that a spline smoother will provide the best mean squared error fit.
- One type of spline, the smoothing spline, is designed to prevent overfitting, a prominent concern with nonparametric smoothers.
- There have been a number of advances in the methods used to estimate splines, while advances in local regression has been fairly static

Spline

- Splines are piecewise regression functions we constrain to join at points called knots



- We are fitting $y = \alpha + \beta_1 x + \beta_2 (x - c)_+ + \epsilon$, where $(x - c)_+ = \mathbb{I}(x - c \geq 0) \times (x - c)$

Terms

- Basic function: a transformation of a single predictor
- What did we do in previous example?

$$\begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \vdots \\ 1 & x_k \\ \vdots & \vdots \\ 1 & x_n \end{bmatrix} \to \begin{bmatrix} 1 & x_1 & 0 \\ 1 & x_2 & 0 \\ \vdots & \vdots & \vdots \\ 1 & x_k & x_k - c \\ \vdots & \vdots & \vdots \\ 1 & x_n & x_n - c \end{bmatrix}$$

Cubic Spline

- Basic function is a cubic function

$$f(x) = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3 + \beta_4 (x - \eta_1)^3_+ + ... + \beta_{K+3} (x - \eta_K)^3_+$$

$$\begin{bmatrix} 1 & x_1 & x_1^2 & x_1^3 \\ 1 & x_2 & x_2^2 & x_2^3 \\ \vdots & \vdots & \vdots & \vdots \\ 1 & x_k & x_k^2 & x_k^3 \\ \vdots & \vdots & \vdots & \vdots \\ 1 & x_n & x_n^2 & x_n^3 \end{bmatrix} \rightarrow \begin{bmatrix} 1 & x_1 & x_1^2 & x_1^3 & 0 & 0 \\ 1 & x_2 & x_2^2 & x_2^3 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & x_k & x_k^2 & x_k^3 & (x_k - \eta_1)^3 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & x_n & x_n^2 & x_n^3 & (x_n - \eta_1)^3 & (x_n - \eta_2)^3 \end{bmatrix}$$

- How many parameters do we have?

Some restriction

- There are a lot of different ways to write a cubic spline
- Notice that: to make function smooth, we (implicitly) impose some constraints:
- First and second order derivative at knots exist
- effective degree of freedom: intuitively, we put more constraint, then we have less degree of freedom to choose parameters
- Mathematically, trace of our prediction matrix

Natural cubic spline

- We only have data point within $[x_1, x_n]$
- Because spline is essentially a piece-wise polynomial regression, we don't have data point to estimate the behavior outside the range
- Natural cubic splines add two knots to the fit at the minimum and maximum values of x and fit a linear function between the additional knots at the boundary and the interior knots.

Smoothing spline

- Spline may over-fit the data
- Put penalty on "roughness"
- Previously, spline minimizes $SS(f) = \sum(y - f(x))^2$
- Penalty on "roughness":$\lambda \int_{x_1}^{x_n} f''(x)^2 dx$
- New target: minimize $SS(f) = \sum(y - f(x))^2 + \lambda \int_{x_1}^{x_n} f''(x)^2 dx$
- Fact: the minimizer will be a natural spline function $\hat{f}$ s.t. $f(\hat{x_i}) = f(x_i)$
- Why? What are the knots?

Penalized cubic spline

- What `mgcv::gam` does for cubic spline
- Instead of putting knots to all observation, we set number of knots

Cross-validation

- Two parameters: λ and K
- How to choose them?
- Cross-validation!
- A lot of metric can be used: AIC, BIC, MSE
- Usually, people do LOOCV: $CV = \frac{1}{N}\sum_i (f(x_i) - f_{-i}(x_i))^2$
- `mgcv::gam` provide GCV (generalized cross-validation), which is an approximation of LOOCV when dataset is large
- Tend to overfit however