

STK2100 - Machine Learning and Statistical Methods for Prediction and Classification

Spring 2026

Kernel smoothing methods

Covers Sections 6.1-6.3

Ingrid Hobæk Haff
Department of Mathematics
University of Oslo

Local regression models

- Assume that we want to predict Y , with

$$Y = f(\mathbf{x}) + \varepsilon, \quad E[\varepsilon] = 0$$

- So far, we have used methods that use all data to estimate $f(\mathbf{x})$ for all $\mathbf{x}$ s simultaneously
- An alternative is to use only the observations close to the point $\mathbf{x}_0$ of interest
- The main assumption we make is that $f(\mathbf{x})$ is smooth in $\mathbb{R}^p$
- One such method is ***k*-nearest neighbour** methods
- Other alternatives are ***kernel smoothers*** and ***local linear regression*** models

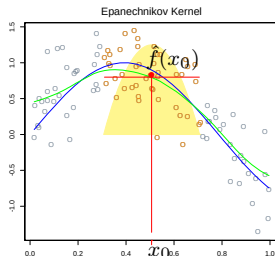
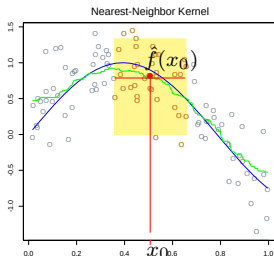
One-dimensional kernel smoothers

- The k -nearest-neighbour estimate is given by

$$E(Y|X = x_0) \approx \hat{f}(x_0) = \frac{1}{k} \sum_{\{i: x_i \in N_k(x_0)\}} y_i = \frac{\sum_{i=1}^N I(x_i \in N_k(x_0)) y_i}{\sum_{i=1}^N I(x_i \in N_k(x_0))}$$

where $N_k(x_0)$ are the k nearest neighbours of x_0

- A weakness of this estimate is that it is rough, with a binary decision boundary that depends on whether a point is inside or outside the neighbourhood $N_k(x_0)$
- A smoother estimate is obtained by instead weighting the data points according to their distance to x_0



One-dimensional kernel smoothers

- The weighting of the data points is done by substituting $I(x_i \in N_k(x_0))$ with a **kernel** $K_\lambda(x_0, x_i)$:

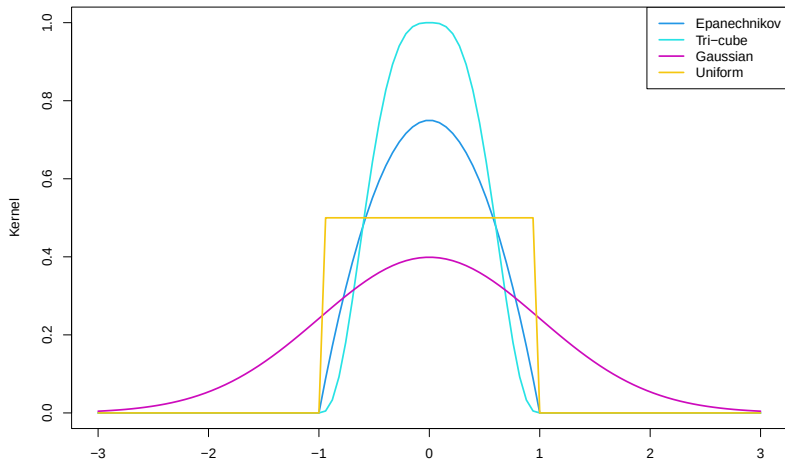
$$\hat{f}(x_0) = \frac{\sum_{i=1}^N K_\lambda(x_0, x_i) y_i}{\sum_{i=1}^N K_\lambda(x_0, x_i)} \quad K_\lambda(x_0, x_i) = D\left(\frac{|x_i - x_0|}{\lambda}\right)$$

- Note that $K_\lambda(x_0, x_i) = 0$ for $|x_i - x_0| > \lambda$
- A common choice is the **Epanechnikov quadratic kernel**

$$D(t) = \begin{cases} \frac{3}{4}(1 - t^2) & \text{if } |t| < 1; \\ 0 & \text{otherwise} \end{cases}$$

- The resulting estimate is called the **Nadaraya-Watson** kernel smoother
- Alternatives kernels are
 - the **Gaussian kernel**: $D(t) = \exp(-0.5t^2)$
 - the **tri-cube kernel**: $D(t) = (1 - |t|^3)^3 I(|t| \leq 1)$
 - the **uniform kernel**: $D(t) = I(|t| \leq 1)$

Examples of kernels



Properties of the kernel smoother

- The expectation is given by

$$\begin{aligned} E(\hat{f}(x_0)) &= E\left(\frac{\sum_{i=1}^N K_\lambda(x_0, x_i) Y_i}{\sum_{i=1}^N K_\lambda(x_0, x_i)}\right) = \frac{\sum_{i=1}^N K_\lambda(x_0, x_i) E(Y_i)}{\sum_{i=1}^N K_\lambda(x_0, x_i)} \\ &= \frac{\sum_{i=1}^N K_\lambda(x_0, x_i) f(x_i)}{\sum_{i=1}^N K_\lambda(x_0, x_i)} \\ &= f(x_0) + \frac{\sum_{i=1}^N K_\lambda(x_0, x_i) (f(x_i) - f(x_0))}{\sum_{i=1}^N K_\lambda(x_0, x_i)} \end{aligned}$$

- The variance is given by

$$\begin{aligned} \text{Var}(\hat{f}(x_0)) &= \text{Var}\left(\frac{\sum_{i=1}^N K_\lambda(x_0, x_i) Y_i}{\sum_{i=1}^N K_\lambda(x_0, x_i)}\right) = \frac{\sum_{i=1}^N K_\lambda(x_0, x_i)^2 \text{Var}(Y_i)}{(\sum_{i=1}^N K_\lambda(x_0, x_i))^2} \\ &= \sum_{i=1}^N \underbrace{\frac{(K_\lambda(x_0, x_i))^2}{(\sum_{j=1}^N K_\lambda(x_0, x_j))^2}}_{=w_{i,0}^2} \sigma^2 = \sigma^2 \sum_{i=1}^N w_{i,0}^2 \end{aligned}$$

Properties of the kernel smoother

- The bias $f(x_0) - E(\hat{f}(x_0))$ is small if $f(x_i) \approx f(x_0)$ when $K_\lambda(x_0, x_i)$ large, i.e. when $|x_i - x_0|$ is small
- This is obtained when λ is small
- On the other hand, the variance $\text{Var}(\hat{f}(x_0))$ is minimised by minimising $\sum_{i=1}^N w_{i,0}^2$ subject to $\sum_{i=1}^N w_{i,0} = 1$
- The minimum then occurs at $w_{i,0} = K_\lambda(x_0, x_i) / \sum_{j=1}^N K_\lambda(x_0, x_j) = 1/N$
- This corresponds to a very large λ
- Hence the choice of the **smoothing parameter** λ is a tradeoff between bias and variance

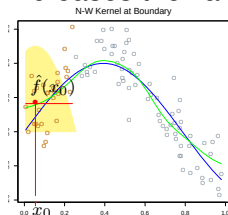
Further properties of kernel smoothers

- We can generalise to **adaptive neighbourhoods**

$$K_{\lambda}(x_0, x_i) = D \left(\frac{|x - x_0|}{h_{\lambda}(x_0)} \right),$$

where h_{λ} is a width function

- There are fewer points available at the boundaries, which increases the variance



- Kernel smoothers can also give high bias, particularly at the boundary, though bias-correction methods exists
- Note that $\hat{\mathbf{y}}$ is linear in $\mathbf{y}$, i.e. $\hat{\mathbf{y}} = \mathbf{S}\mathbf{y}$, so that the model complexity/degrees of freedom can be computed as $df = \text{tr}(\mathbf{S})$

Kernel smoothers as minimisation problems

- Assume that $f(x)$ is approximately constant around x_0
- Then a reasonable criterion is

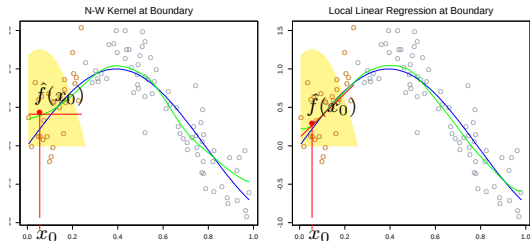
$$\min_{\alpha(x_0)} \sum_{i=1}^N K_{\lambda}(x_0, x_i) (y_i - \alpha(x_0))^2$$

- The solution is then the kernel smoother

$$\hat{\alpha}(x_0) = \frac{\sum_{i=1}^N K_{\lambda}(x_0, x_i) y_i}{\sum_{i=1}^N K_{\lambda}(x_0, x_i)}$$

Local linear regression

- Kernel smoothers assume that $f(x_0)$ is nearly constant around x_0
- Assuming instead that $f(x_0)$ is nearly linear around x_0 , we obtain **local linear regression**



- One then solves a separate weighted least squares problem at each target point x_0

$$\min_{\alpha(x_0), \beta(x_0)} \sum_{i=1}^N K_{\lambda}(x_0, x_i) (y_i - \alpha(x_0) - \beta(x_0)x_i)^2$$

- The estimate of $f(x_0)$ is then $\hat{f}(x_0) = \hat{\alpha}(x_0) + \hat{\beta}(x_0)x_0$

Local linear regression

- Define

$$\mathbf{B} = \begin{pmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \\ 1 & x_N \end{pmatrix} \quad \mathbf{W}(x_0) = \begin{pmatrix} K_\lambda(x_0, x_1) & 0 & & \\ 0 & K_\lambda(x_0, x_2) & 0 & \\ & \vdots & \ddots & \\ & & 0 & K_\lambda(x_0, x_N) \end{pmatrix}$$

and $\theta = (\alpha(x_0), \beta(x_0))$

- On vector/matrix form, the least squares problem is to find the θ that minimises

$$D(\theta) = (\mathbf{y} - \mathbf{B}\theta)^T \mathbf{W}(x_0) (\mathbf{y} - \mathbf{B}\theta)$$

- We have

$$\frac{\partial D(\theta)}{\partial \theta} = -2\mathbf{B}^T \mathbf{W}(x_0) \mathbf{y} + 2\mathbf{B}^T \mathbf{W}(x_0) \mathbf{B} \theta,$$

Local linear regression

- Solving $\partial D(\boldsymbol{\theta})/\partial \boldsymbol{\theta} = \mathbf{0}$ for $\boldsymbol{\theta}$ gives

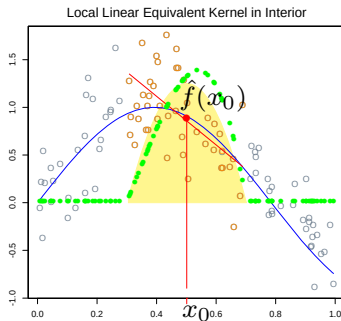
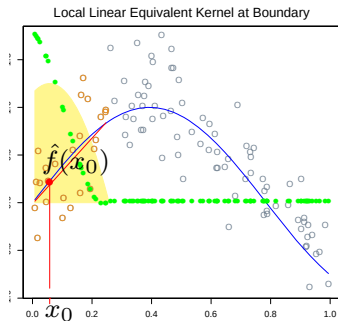
$$\hat{\boldsymbol{\theta}} = \left(\mathbf{B}^T \mathbf{W}(x_0) \mathbf{B} \right)^{-1} \mathbf{B}^T \mathbf{W}(x_0) \mathbf{y}$$

- The resulting estimate of $f(x_0)$ is

$$\begin{aligned} \hat{f}(x_0) &= (1 \quad x_0) \hat{\boldsymbol{\theta}} \\ &= (1 \quad x_0) (\mathbf{B}^T \mathbf{W}(x_0) \mathbf{B})^{-1} \mathbf{B}^T \mathbf{W}(x_0) \mathbf{y} = \sum_{i=1}^N l_i(x_0) y_i \end{aligned}$$

- This is again a linear function of $\mathbf{y}$
- The $l_i(x_0)$ s are sometimes called **equivalence kernels**

Local equivalence kernel at the boundary and in the interior



Properties of local linear regression

- Using the Taylor expansion of $f(x_i)$ around x_0 , we obtain

$$\begin{aligned} E(\hat{f}(x_0)) &= E\left(\sum_{i=1}^N l_i(x_0) Y_i\right) = \sum_{i=1}^N l_i(x_0) E(Y_i) = \sum_{i=1}^N l_i(x_0) f(x_i) \\ &\approx f(x_0) \sum_{i=1}^N l_i(x_0) + f'(x_0) \sum_{i=1}^N l_i(x_0)(x_i - x_0) \\ &\quad + \frac{f''(x_0)}{2} \sum_{i=1}^N l_i(x_0)(x_i - x_0)^2 \end{aligned}$$

- It may be shown that $\sum_{i=1}^N l_i(x_0) = 1$ and $\sum_{i=1}^N (x_i - x_0) l_i(x_0) = 0$, so that

$$E[\hat{f}(x_0)] \approx f(x_0) + \frac{f''(x_0)}{2} \sum_{i=1}^N l_i(x_0)(x_i - x_0)^2$$

- Hence, local linear regression typically gives low bias

Local polynomial regression

- Local regression can be expanded to **local polynomial regression**

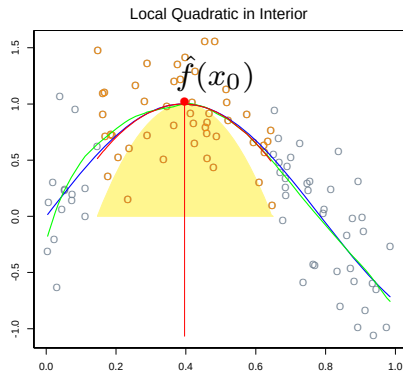
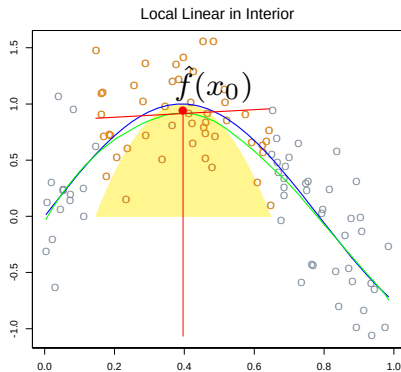
$$\min_{\alpha(x_0), \beta_j(x_0)} \sum_{i=1}^N K_\lambda(x_0, x_i) \left(y_i - \alpha(x_0) - \sum_{j=1}^d \beta_j(x_0) x_i^j \right)^2$$

- The estimate of $f(x_0)$ is then

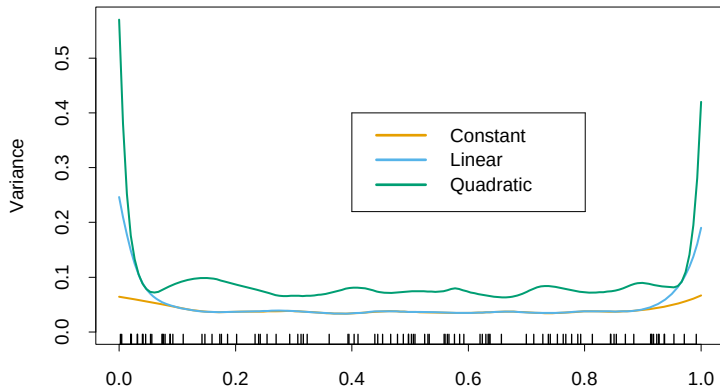
$$\hat{f}(x_0) = \hat{\alpha}(x_0) + \sum_{j=1}^d \hat{\beta}_j(x_0) x_0^j$$

- It may be shown that the bias will only have components of $x_i - x_0$ of degree $d + 1$ and higher
- Thus, the bias is reduced compared to local linear regression, but the variance is typically higher

Local polynomial regression - bias



Local polynomial regression - variance

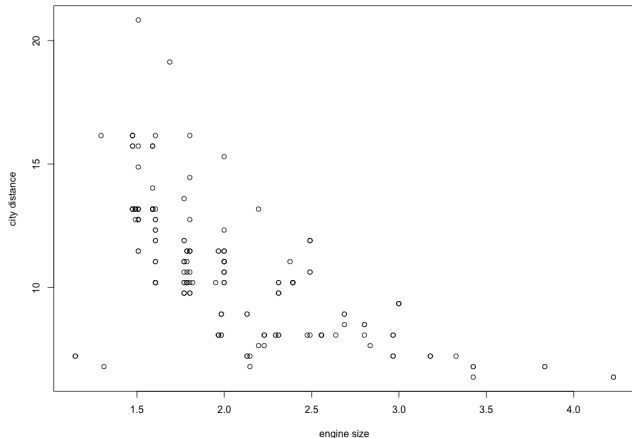


Selection of λ

- All methods have a parameter λ to be specified
- This is a tradeoff between bias and variance: a small λ gives a small bias, but a large variance and vice versa
- λ may be selected directly, or indirectly through the degrees of freedom
- The degrees of freedom are easier to compare between different methods, e.g between local linear regression and smoothing splines
- λ can also be selected automatically, using cross-validation
- In particular, efficient computation of leave-one-out cross-validation can be derived

Example: Auto data

- Response: city distance covered (km/l) (`city.distance`)
- Explanatory variable:
 - engine size in l (`engine.size`)
- We want to predict the distance covered by fuel unit of a car based on the size of its engine



Local regression in $\mathbb{R}^p$

- All kernel methods can (in principle) be easily generalised to higher dimension
- Define the multi-dimensional kernel
$$K_\lambda(\mathbf{x}_0, \mathbf{x}_i) = D(\lambda^{-1} \|\mathbf{x}_i - \mathbf{x}_0\|)$$
- Further, let $\mathbf{b} = (1, x_1, \dots, x_p)^T$
- The local multiple linear regression model is then obtained by solving

$$\min_{\beta(\mathbf{x}_0)} \sum_{i=1}^N K_\lambda(\mathbf{x}_0, \mathbf{x}_i) (y_i - \mathbf{b}_i^T \beta(\mathbf{x}_0))^2$$

- This is still a linear predictor, i.e we may write
$$\hat{f}(\mathbf{x}_0) = \sum_{i=1}^N l_i(\mathbf{x}_0) y_i$$

Local regression in $\mathbb{R}^p$

- However, the boundary problems are much more serious in higher dimension, as the fraction of points close to the boundary is higher
- In practice, local regression becomes less useful in dimensions much higher than two or three, above which it becomes impossible to obtain both low bias and low variance
- Also, visualisation of $\hat{f}(\mathbf{x}_0)$ becomes difficult in higher dimensions
- An alternative is a trellis display, where one plots the response as a function of one of the covariates, conditioning on the remaining covariates being within a certain interval

Illustration of trellis display

Predicting ozone from wind, temperature and radiation

