MSE = $\text{bias}^2 + \text{Variance}$

$\text{bias} = \mathbb{E}(\hat{r}(x)) - r(x)$

(Non-Parametric Regression) In machine learning this is known as "learning a function".

$Y_i = r(x_i) + \epsilon_i \leftarrow \text{model}$

We will construct $\hat{r}(x) \leftarrow \text{estimand of } r$, or smoother.

Assume that $\text{Var}(\epsilon_i) = \sigma^2$, independent of $x_i$.

(Linear Smoother) (not linear regression exactly).

Def: $\hat{r}$ is a linear smoother if for each $x_i$ there exists a vector $\hat{\ell}(x) = \begin{pmatrix} l_0(x) \\ l_1(x) \\ \vdots \\ l_n(x) \end{pmatrix}$ such that

$\hat{r}(x) = \sum_{i=1}^{n} l_i(x) Y_i$

$\hat{r} = \begin{pmatrix} \hat{r}(x_1) \\ \vdots \\ \hat{r}(x_n) \end{pmatrix}$ fitted values

$\hat{Y} = \begin{pmatrix} \hat{Y}_1 \\ \vdots \\ \hat{Y}_n \end{pmatrix}$
\[ \hat{y} = L Y, \quad \text{where} \quad L_{ij} = l_j(x_i) \]

- The matrix \( L \) is called the smoothing matrix, or the "hat" matrix.
- The \( i \)th row is the "effective kernel" for the estimator of \( y(x_i) \).
- The effective degrees of freedom = \( v = \text{tr}(L) \).

**Exercise**

Reinterpret linear least squares in this matrix vector formulation. (I.e., find \( L, Y \)).

It will turn out that most linear smoothers have the property that for all \( x \), \( \sum l_i(x) = 1 \). If this is true, then if \( Y_i = c \) for all \( i \), then \( \hat{Y}(x) = \sum l_i(x) Y_i = c \). (The smoother preserves constants.)

**Example**

**Regressogram**

Let \( x \in (a,b) \), and compute \( m \) bins on this interval:

\[ a \quad \underbrace{B_1 \cdots B_{m-1}}_{\text{Decide on } m}, \quad B_m \quad b \]

For \( x \in B_j \), define \( \hat{r}(x) = \frac{1}{l_j} \sum_{i : x_i \in B_j} Y_i \), \( l_j = \# x_i \in B_j \)

computing the average of \( Y_i \) over \( \text{bin } B_j \).
For $x \in B_j$, set $l_i(x) = \begin{cases} \frac{1}{k_j} & \text{if } x_i \in B_j \\ 0 & \text{otherwise} \end{cases}$

$\Rightarrow \hat{y}(x) = \sum l_i(x) y_i$ for $x \in B_j$.

and $l(x) = \begin{pmatrix} 0 \\ \vdots \\ y_{x_j} \\ \vdots \\ y_{x_0} \\ \vdots \\ 0 \end{pmatrix} i$ such that $x_i \in B_j$.

If $n = 9$, $m = 3$, and $k_1 = k_2 = k_3 = 3$.

$\Rightarrow L = \begin{pmatrix} y_3 & y_3 & y_3 \\ y_3 & y_3 & y_3 \\ y_3 & y_3 & y_3 \\ & & \\ & & \\ & & \\ & & \\ 0 & 0 & 0 \end{pmatrix}$

$\Rightarrow \text{tr}(L) = 3$

Ex: Local Smoothing / Averaging

For $h > 0$, set $B_x = \{ i : |x_i - x| \leq h \}$

$\Rightarrow \hat{y}(x) = \frac{1}{n_x} \sum_{i \in B_x} y_i$ average our points within a distance of $h$ from $x$. 

$3$
The key question in both of these examples is how to choose $h$?

- if $h$ is big, we get a very smooth $\hat{r}$
- if $h$ is small, then $\hat{r}$ looks a lot like $Y_i$

$h$ is generally referred to as the **BANDWIDTH**

### Choosing the smoothing parameter

To recall the risk: $R(h) = E\left(\frac{1}{n} \sum_{i=1}^{n} (\hat{r}(x_i) - r(x_i))^2\right)$

- this depends on the unknown $r$.

Instead, choose $h$ to minimize an estimate of $R(h)$. 

**Idea**: Use the "training error": $\frac{1}{n} \sum_{i=1}^{n} (Y_i - \hat{r}(x_i))^2$

- Poor estimate of $R(h)$
- Tends to lead to overfitting since outliers are given equal weight as other data points.

### A better option

**Def**: "Leave-one-out cross validation"

$$CV = \hat{R}(h) = \frac{1}{n} \sum_{i=1}^{n} (Y_i - \hat{r}_{(\hat{c})}(x_i))^2$$

$\hat{c}$ obtained by ignoring the $i$th data point.
For linear smoothers, \( \hat{r}(x) = \sum Y_i \ell_i(x) \)

\[ \hat{r}_{(-i)}(x) = \text{leave one out estimator} \]

\[ = \sum_{j \neq i} Y_j \ell_{j,(-i)}(x) \]

where \( \ell_{j,(-i)}(x) = \begin{cases} 0 & \text{if } j = i \\ \frac{\ell_j(x)}{\sum \ell_{k+1}(x)} & \text{if } j \neq i \end{cases} \)

re-weighting of \( \ell_j \)

General idea: compute \( \hat{r} \) with only part of the data and then check.

If we use this as \( \hat{R}(n) \), then what can we say?

\[ E \left( Y_i - \hat{r}_{(-i)}(x_i) \right)^2 = E \left( (Y_i - r(x_i)) + (r(x_i) - \hat{r}_{(-i)}(x_i)) \right)^2 \]

\[ = E \left( (Y_i - r(x_i))^2 + 2(Y_i - r(x_i))(r(x_i) - \hat{r}_{(-i)}(x_i)) \right) \]

model: \( Y_i = r(x_i) + \xi_i \)

\[ = \sigma^2 + E \left( (r(x_i) - \hat{r}_{(-i)}(x_i))^2 \right) \]

for large \( n \) \( \approx \sigma^2 + E \left( (r(x_i) - \hat{r}(x_i))^2 \right) \)

\[ \Rightarrow E(\hat{R}) \approx \sigma^2 + \beta \]

\( = \text{predictive risk} \Rightarrow \text{"nearly unbiased" since presumably } \beta \gg \sigma^2. \)

Is this expensive to do for all \( i \)?
Thus, for a linear smoother, the CV estimator can be written as:

\[ \hat{\sigma}^2(h) = \frac{1}{n} \sum_{i=1}^{n} \left( Y_i - \hat{\tau}(x_i) \right)^2 \]

Then, \( h \) can be chosen by minimizing \( \hat{\sigma}^2(h) \).

**Local Regression**

**Idea:** Give more weight to \( x_i, y_i \) that are near to where you want to evaluate \( \hat{\tau} \).

**Definition**

A **kernel** is a function \( K = K(x) \) such that:

\[
\int K(x) \, dx = 1 \\
\int xK(x) \, dx = 0 \\
\int x^2 K(x) \, dx = \sigma_{kk}^2 > 0 \\
< \infty
\]

**Ex:**

\[
\frac{1}{2} \mathbb{1}(x) = \begin{cases} 1 & \text{if } |x| \leq 1 \\ 0 & \text{else} \end{cases} \\
\frac{1}{\sqrt{2\pi}} e^{-x^2/2} \\
\frac{3}{4}(1-x^2) \mathbb{1}(x)
\]

**Note that:** For \( h > 0 \), and some \( x' \), we have that

\[ K_n(x-x') = \frac{1}{h} K \left( \frac{x-x'}{h} \right) \] is also a kernel centered at \( x' \).
Use these "kernels" to smooth out the noisy data \((x_i, y_i)\).

**Definition** For \(h > 0\), the Nadaray-Watson kernel estimator is:

\[
\hat{f}(x) = \frac{1}{n} \sum_{i=1}^{n} K\left(\frac{x-x_i}{h}\right) y_i
\]

where

\[
K\left(\frac{x-x_i}{h}\right) = \frac{1}{h} K\left(\frac{x-x_i}{h}\right)
\]

This form is similar to barycentric interpolation formula, but it is not an interpolant.

**Goal:**

1. Choose \(h\) via cross-validation.
2. The choice of \(K\) is less important, especially when there is a lot of data.

In order to choose \(h\), we need to be able to estimate the risk. For our purposes, assume that the \(x_i\)'s are randomly drawn from some density \(f\).

The risk can then be written as:

\[
\text{Risk} (\hat{f}, f) = \frac{h^4}{4} \left( \int x^2 K(x) \, dx \right)^2 \left( \int \left( f'' + 2f' \frac{f'}{f} \right) \, dx \right)^2 + \frac{\sigma^2}{nh} \int \left( \int K(x) \, dx \right) \left( \int \frac{1}{f} \, dx \right) + o\left( \frac{h^4}{n} \right) + o(h^4)
\]

When we have assumed that \(h \to 0\) and \(nh \to \infty\),

\[
\text{bias}^2 \sim h^4
\]

\[
\text{variance} \sim \frac{1}{nh}
\]
Since this depends on \( f \), the bias is sensitive to the distribution of the \( X_j \)'s.

1. To minimize \( R \), set \( \frac{dR}{dn} = 0 \) and solve.

\[
\Rightarrow h_x = \Theta \left( \frac{1}{n^{1/2}} \right)
\]

\[
\Rightarrow R = \Theta \left( \frac{1}{n^{1/4}} \right). \quad (\text{vs. MLE for least squares, } R \sim \frac{1}{n})
\]

Note

**Homoscedasticity** vs. **Heteroscedasticity**

\[
y = r(x) + \varepsilon
\]

\[
\text{Var}(\varepsilon) = \sigma^2
\]

and \( \varepsilon \) is independent of \( x \) (\( \varepsilon | x = 0 \)).

Reformulate the model to separate \( \varepsilon \) from \( r \).

\[
y - r = \varepsilon
\]

\[
(y - r)^2 = \sigma^2 \varepsilon^2
\]

\[
\log (y - r)^2 = \log \sigma^2 + \log \varepsilon^2
\]

\[
\Rightarrow 2\varepsilon = \log \sigma^2(x) + \delta_i
\]

Then our goal is to estimate \( \log \sigma^2 \) of \( x \).
There is a two-step procedure for doing this:

1. Estimate \( r \) in \( Y_i = r(x_i) + \tilde{e}_i \)
   to get \( \hat{r} \).

2. Compute \( \tilde{e}_i = \log \left( Y_i - \hat{r}(x_i) \right)^2 \).

3. Regress \( \tilde{e}_i \) on \( x_i \) again to get an estimator
   \( \hat{q} = \log(\sigma(x)) \), set \( \hat{\sigma}^2(x) = e^q > 0 \).

Density Estimation

Setup: Observe some data \( X_1, \ldots, X_n \sim F \), and
therefore the density is \( f = F' \).

Goal: Estimate \( f \) using as few assumptions as possible.

Still a smoothing problem:

\[
\begin{align*}
\text{unsmoothed} & \\
\text{oversmoothed} & \\
\text{undersmoothed} & \end{align*}
\]

Let \( \hat{f} \) be our estimate
of \( f \).

One way to measure the error:

\[
L = \int (f(x) - \hat{f}(x))^2.
\]