Prediction Error:
The Bias-Variance Trade-Off

Aldo Solari

Mean squared error

- The mean squared error for the training data

$$\mathrm{MSE}_{\mathrm{Tr}} = \frac{1}{n}\sum_{i=1}^{n}[y_i - \hat{f}(x_i)]^2$$

is not a good measure of performance

- We would like to have a good performance

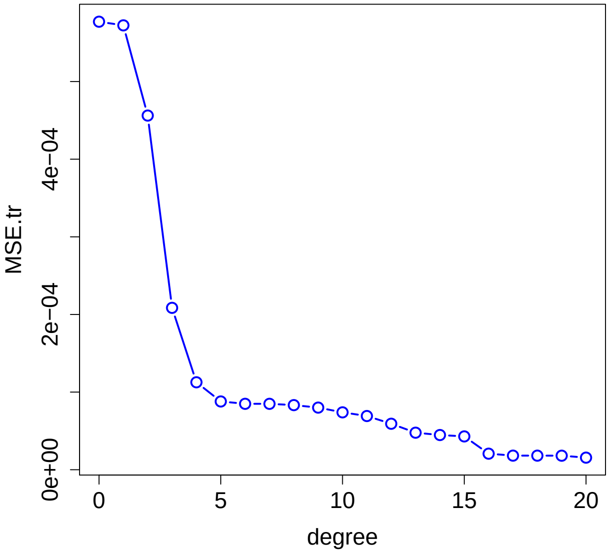
$$\mathrm{MSE}_{\mathrm{Te}} = \frac{1}{m}\sum_{i=1}^{m}[y_i^* - \hat{f}(x_i^*)]^2$$

on the test data

$$(x_1^*, y_1^*), (x_2^*, y_2^*), \ldots, (x_m^*, y_m^*)$$

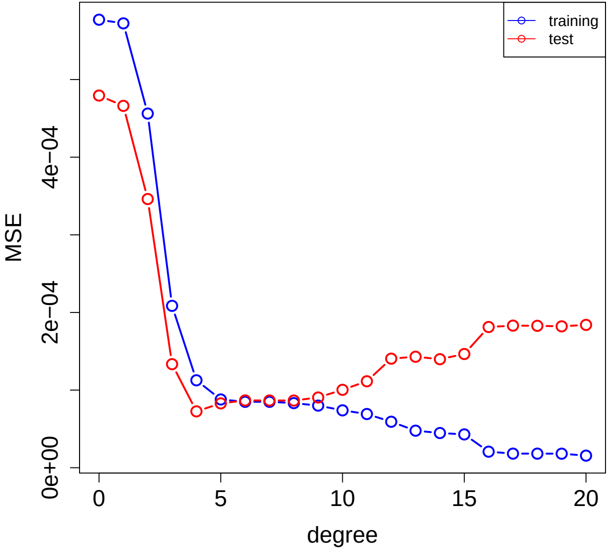
```
load("poly.Rdata")

n <- nrow(train)
ds = 0:20

fun <- function(d) if (d==0) lm(y~1, train)
          else lm(y~poly(x,degree=d), train)
fits <- sapply(ds, fun)

MSEs.tr <- unlist( lapply(fits, deviance) )/n
plot(ds, MSEs.tr, type="b")
```

Training data MSE

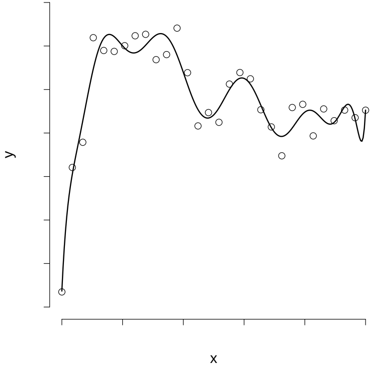


```
yhats <- lapply(fits, predict)
MSEs.te <- unlist( lapply(yhats,
           function(yhat) mean((test$y - yhat)^2)) )

plot(ds, MSEs.tr, type="b", col=4)
lines(ds, MSEs.te, type="b", col=2)
```

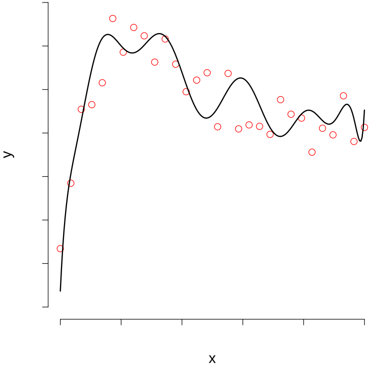
Test data MSE



Overfitting

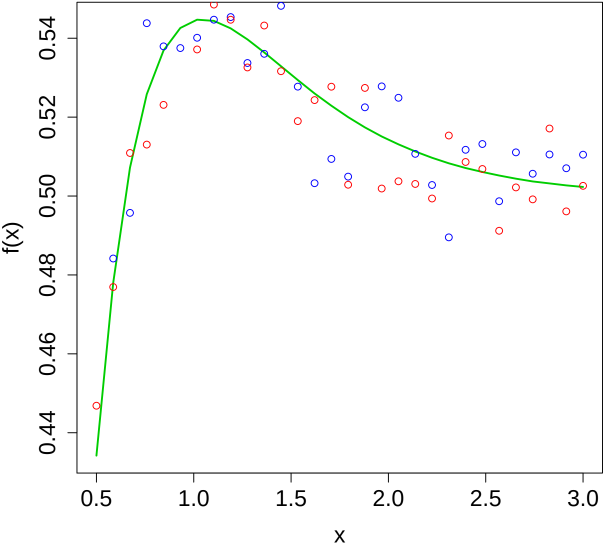


Yesterday data

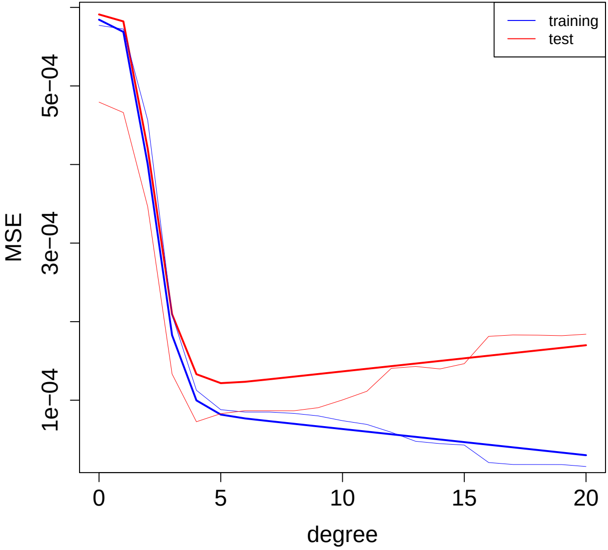


Tomorrow data

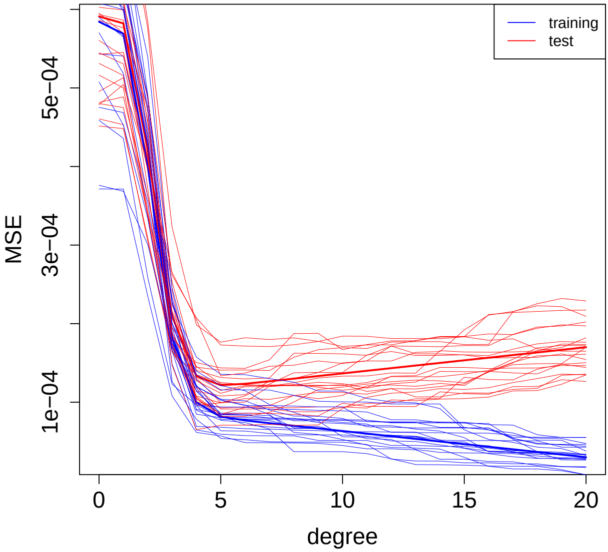
True function



Expected MSE



Expected MSE



Outline

Regression function

- (X, Y) have some unknown joint distribution
- We want to predict Y from X
- Over all functions f, the expected test error, measured in terms of squared error loss,

$$\mathbb{E}[(Y - f(X))^2] \tag{1}$$

  is minimized at

$$f(x) = \mathbb{E}(Y|X = x) \tag{2}$$

- $f(x)$ is called the (true) regression function
- **Homework**: prove that (1) is minimized at (2)

Example of regression function

- Joint distribution:

$$\begin{pmatrix} Y \\ X \end{pmatrix} \sim \mathcal{N}\left[\begin{pmatrix} \mu_y \\ \mu_x \end{pmatrix}, \begin{pmatrix} \sigma_y^2 & \rho\sigma_x\sigma_y \\ \rho\sigma_x\sigma_y & \sigma_x^2 \end{pmatrix}\right]$$

- Conditional distribution:

$$(Y|X=x) \sim \mathcal{N}\Big(\mu_y + \rho\frac{\sigma_y}{\sigma_x}(x-\mu_x), \sigma_y^2(1-\rho^2)\Big)$$

- Regression function:

$$f(x) = \mathrm{E}(Y|X=x) = \left(\mu_y - \rho\frac{\sigma_y}{\sigma_x}\mu_x\right) + \left(\rho\frac{\sigma_y}{\sigma_x}\right)x$$

The general model

$$Y = f(X) + \varepsilon \tag{3}$$

- Y : continuous response
- $\underset{p\times 1}{X} = (X_1, \ldots, X_p)^\mathsf{T}$: predictors
- $f(X) = \mathbb{E}(Y|X)$: regression function
- ε : error term, assumed with

$$\mathbb{E}(\varepsilon) = 0, \quad \mathbb{V}\mathrm{ar}(\varepsilon) = \sigma^2$$

  and independent of X

Expected test error

- The true $f(x) = \mathbb{E}(Y|X = x)$ is unknown
- We can use the training data

$$(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$$

  to get an estimate $\hat{f}(x)$ of $f(x)$
- The expected test error of $\hat{f}$ is

$$\mathbb{E}[(Y - \hat{f}(X))^2] \tag{4}$$

  where the expectation is over all that is random, namely, the training data $(X_i, Y_i), i = 1, \ldots, n$, and the test point (X, Y)

Sources of error

Irreducible error
Can we ever predict Y from X with zero error? No. Even the true regression function f cannot do this: $\mathbb{E}[(Y - f(X))^2] = \sigma^2$

Estimation bias
What happens if our fitted function $\hat{f}$ belongs to a model class that is far from the true f? E.g. we choose to fit a linear model in a setting where the true relationship is far from linear?

Estimation variance
What happens if our fitted (random) function $\hat{f}$ is itself quite variable? In other words, over different copies of the training data, we end up constructing substantially different functions $\hat{f}$?

Source: Tibshirani R. (2015) Statistical Machine Learning

The bias-variance decomposition

The expected test error, conditional on $X = x$, decomposes into

$$\mathbb{E}[\big(Y - \hat{f}(x)\big)^2 | X = x] = \sigma^2 + \mathbb{E}[\big(f(x) - \hat{f}(x)\big)^2] \tag{5}$$

where σ^2 is the irreducible error. The reducible error

$$\begin{aligned}
\mathbb{E}[(f(x) - \hat{f}(x))^2] &= \big(f(x) - \mathbb{E}(\hat{f}(x))\big)^2 + \mathbb{E}\big[\big(\hat{f}(x) - \mathbb{E}(\hat{f}(x)\big)^2\big] \\
&= \Big\{\mathbb{B}\mathrm{ias}\big(\hat{f}(x)\big)\Big\}^2 + \mathbb{V}\mathrm{ar}\big(\hat{f}(x)\big) \qquad (6)
\end{aligned}$$

decomposes into the squared (estimation) bias and the (estimation) variance

Homework: prove (5) and (6)

The bias-variance decomposition

- Unconditionally over X, the bias-variance decomposition is

$$\mathbb{E}[\big(Y-\hat{f}(X)\big)^2] = \sigma^2 + \int \mathbb{B}\mathrm{ias}^2\big(\hat{f}(x)\big)\mathrm{P}_X(dx) + \int \mathbb{V}\mathrm{ar}\big(\hat{f}(x)\big)\mathrm{P}_X(dx)$$

  where P_X is the distribution of X

- Note that

$$\begin{aligned}
\mathbb{E}(\mathrm{MSE}_{\mathrm{Te}}) &= \sigma^2 + \frac{1}{m}\sum_{i=1}^{m} \mathbb{B}\mathrm{ias}^2\big(\hat{f}(x_i^*)\big) + \frac{1}{m}\sum_{i=1}^{m} \mathbb{V}\mathrm{ar}\big(\hat{f}(x_i^*)\big) \\
&\approx \mathbb{E}[\big(Y - \hat{f}(X)\big)^2]
\end{aligned}$$

  with equality if $X_i^* = X_i$ are constants (X is not random)

- In words, the expected test error is the irreducible error + average (squared) bias + average variance

The bias-variance trade-off

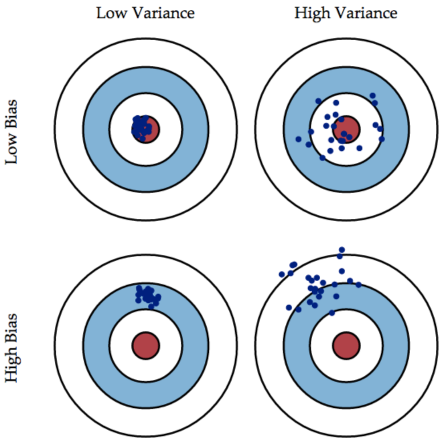
$$\text{Reducible Error} = \text{Bias}^2 + \text{Variance}$$

- Models $\hat{f}$ with low bias tend to have high variance
- Models $\hat{f}$ with low variance tend to have high bias

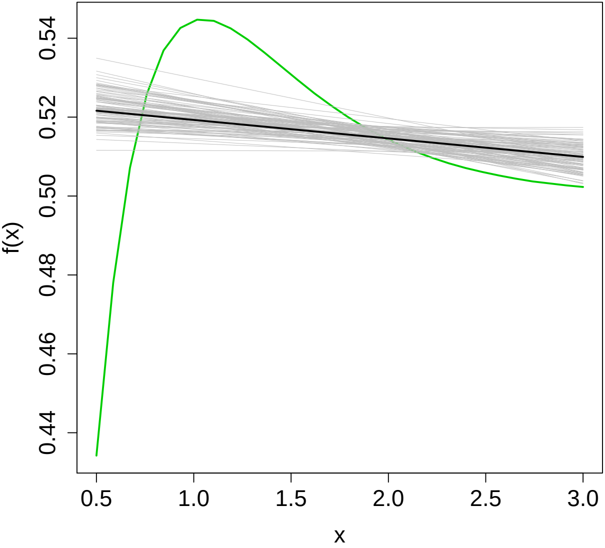
We can see that even if our prediction is unbiased, i.e. $\mathbb{E}(\hat{f}(x)) = f(x)$, we can still incur a large error if it is highly variable. On the other hand, $\hat{f}(x) \equiv 0$ has 0 variance but will be terribly biased

To predict well, we need to balance the bias and the variance

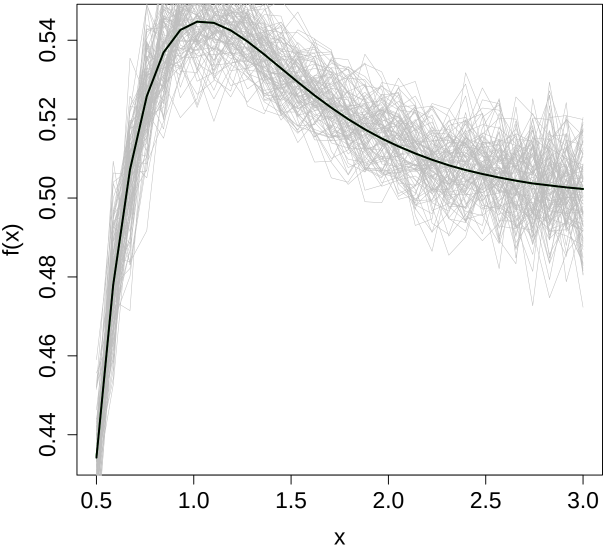
The bias-variance trade-off



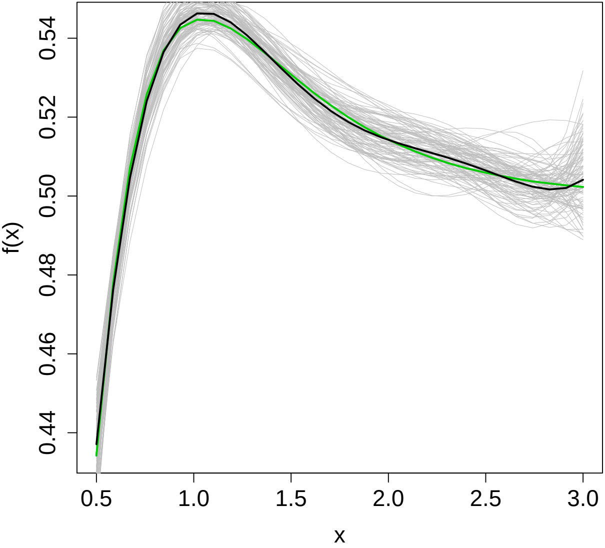
$\hat{f}(x)$ = regression line



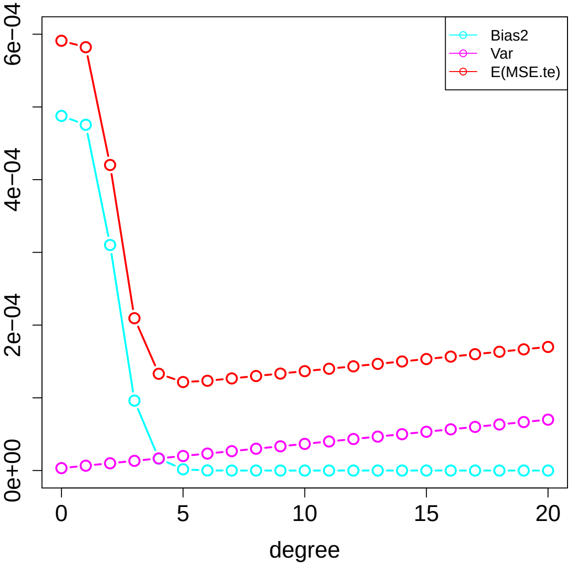
$\hat{f}(x)$ = high-degree polynomial



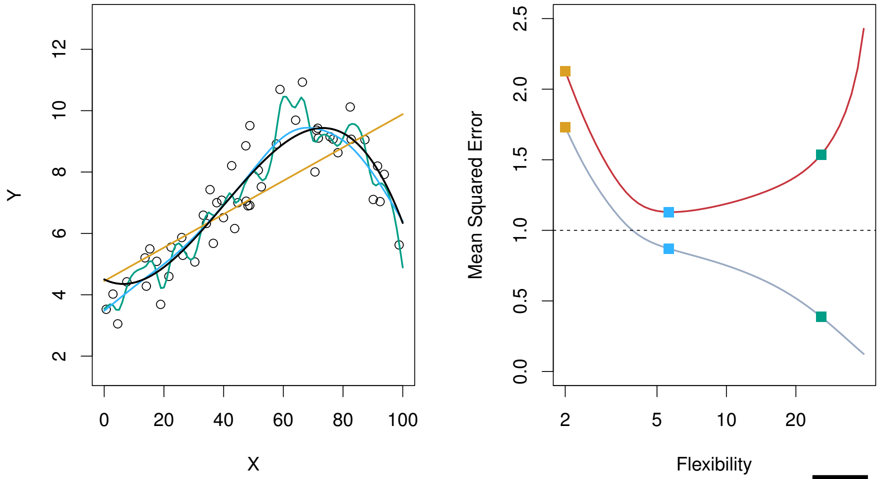
$\hat{f}(x)$ = optimal-degree polynomial



The bias-variance decomposition

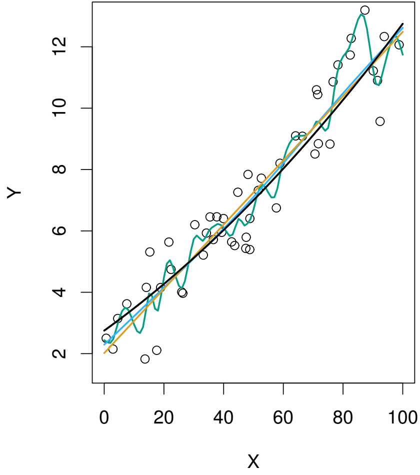


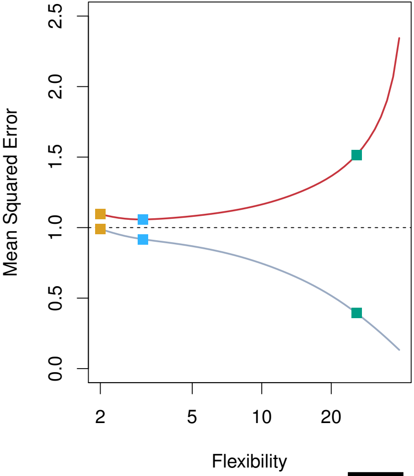
The bias-variance decomposition



Source: Gareth et al. (2013) p. 31

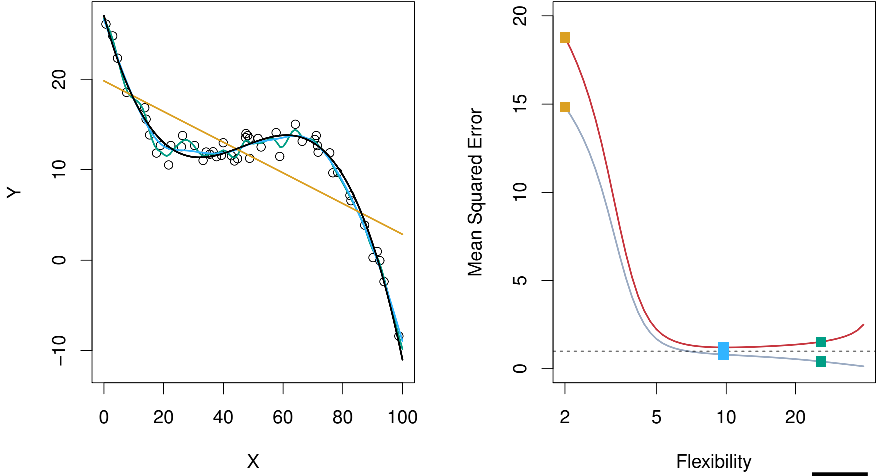
The bias-variance decomposition





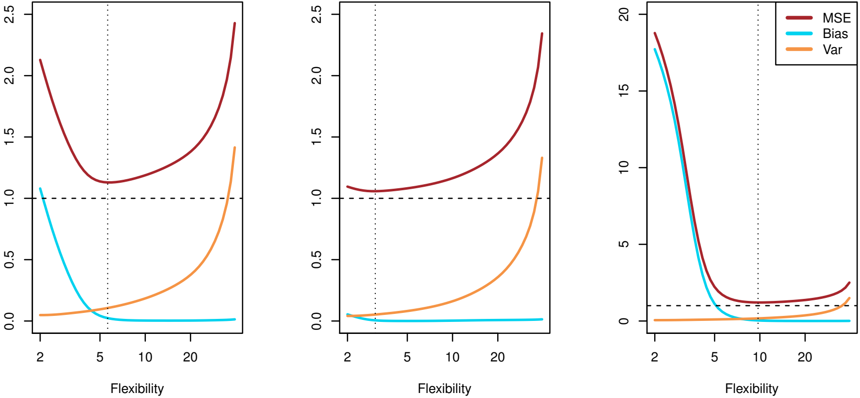
Source: Gareth et al. (2013) p. 33

The bias-variance decomposition



Source: Gareth et al. (2013) p. 34)

The bias-variance decomposition



Source: Gareth et al. (2013) p. 36

Outline

1. The Bias-Variance Decomposition
2. How to Avoid Overfitting

Penalty for the complexity

- In the linear model, if $X_1, \ldots, X_n$ are fixed constants and $n = m$, the expected optimism is

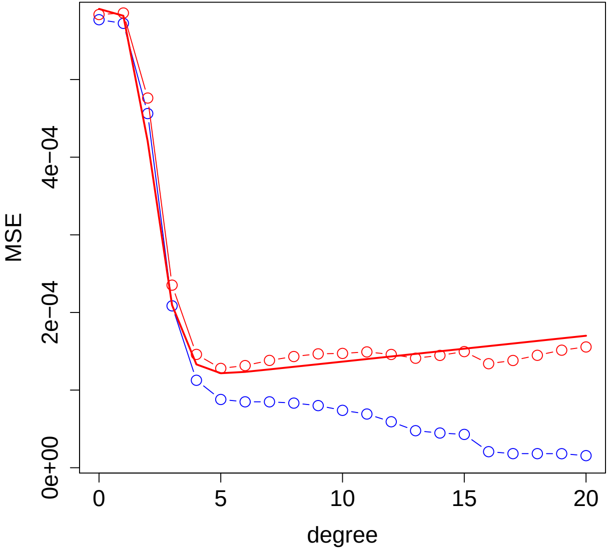
$$\mathbb{E}(\mathrm{Optimism}) = \mathbb{E}(\mathrm{MSE}_{\mathrm{Te}}) - \mathbb{E}(\mathrm{MSE}_{\mathrm{Tr}}) = \frac{2\sigma^2 p}{n}$$

- We can estimate $\mathbb{E}(\mathrm{MSE}_{\mathrm{Te}})$ as

$$\mathbb{E}(\widehat{\mathrm{MSE}_{\mathrm{Te}}}) = \mathrm{MSE}_{\mathrm{Tr}} + \frac{2\sigma^2 p}{n}$$

where the last term represents the penalty for complexity

σ^2 known



σ^2 unknown: AIC and BIC

- σ^2 is usually estimated by $\hat{\sigma}^2 = \dfrac{n\mathrm{MSE}_{\mathrm{Tr}}}{n-p}$
- Finding the minimum of $\mathrm{MSE}_{\mathrm{Tr}} + \frac{2\hat{\sigma}^2 p}{n}$ is equivalent to finding the minimum value of the Akaike Information Criterion

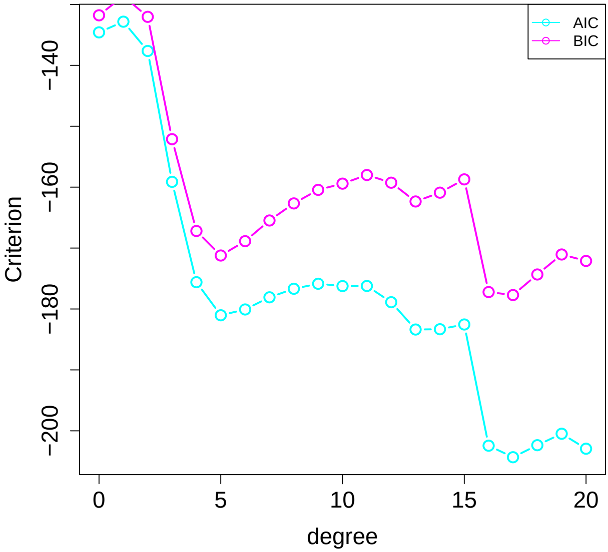
$$\mathrm{AIC} = -2\ell(\hat{\beta}, \hat{\sigma}^2) + 2p$$

  where for the linear model $-2\ell(\hat{\beta}, \hat{\sigma}^2) = n\log(\mathrm{MSE}_{\mathrm{Tr}})$
- A different penalty for complexity:

$$\mathrm{BIC} = -2\ell(\hat{\beta}, \hat{\sigma}^2) + \log(n)p$$

```
AICs <- unlist( lapply(fits, AIC) )
BICs <- unlist( lapply(fits, BIC) )

plot(ds, AICs, type="b", col=5)
lines(ds, BICs, type="b", col=6)
```

AIC and BIC



Cross-validation

- Cross-validation is a method for estimating $\mathbb{E}(\mathrm{MSE}_{\mathrm{Te}})$
- The idea is to hold out a subset of the training observations from the fitting process, and then applying the model to those held out observations
- Cross-validation is a non-parametric method, i.e. it applies to any model

A single hold-out test point

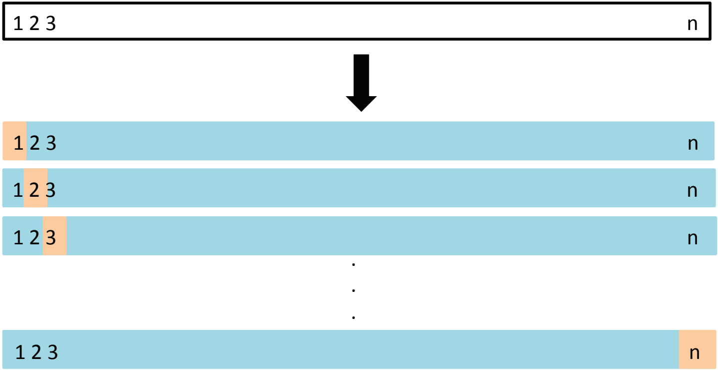
- Hold out the ith training observation (x_i, y_i)
- Use $n-1$ training observations

$$(x_1, y_1), \ldots, (x_{i-1}, y_{i-1}), (x_{i+1}, y_{i+1}), \ldots, (x_n, y_n)$$

  to estimate $\hat{f}^{-i}(x)$
- Use (x_i, y_i) as a test observation

$$\mathbb{E}(\widehat{\mathrm{MSE}_{\mathrm{Te}}}) = \left(y_i - \hat{f}^{-i}(x_i)\right)^2$$

Leave-One-Out Cross-Validation



Source: Gareth et al. (2013), p. 179

Shortcut for LOOCV

For the linear model

$$\frac{1}{n}\sum_{i=1}^{n}\left(y_i - \hat{f}^{-i}(x_i)\right)^2 = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{y_i - \hat{f}(x_i)}{1-h_i}\right)^2$$

where h_i is the ith diagonal element of the projection (hat) matrix

$$\underset{n\times n}{\mathbf{H}} = \mathbf{X}(\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top$$

where $\underset{n\times p}{\mathbf{X}}$ is the design matrix

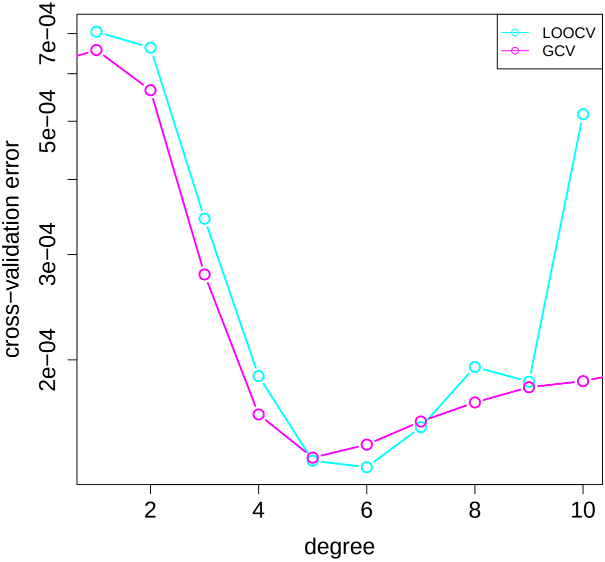
Generalized Cross-Validation

$$\frac{1}{n}\sum_{i=1}^{n}\left(\frac{y_i - \hat{f}(x_i)}{1 - h_i}\right)^2 \approx \frac{\mathrm{MSE}_{\mathrm{Tr}}}{\left(1 - \frac{p}{n}\right)^2}$$

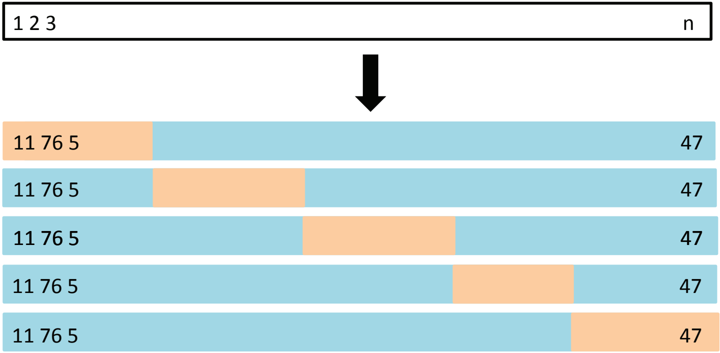
where we approximate each h_i with their average $\frac{1}{n}\sum_{i=1}^{n} h_i = \frac{p}{n}$

```
library(boot)

LOOCV = sapply(1:10, function(d)
     cv.glm(train, glm(y~poly(x,degree=d),
     train, family = gaussian) )$delta[1] )

GCV = MSEs.tr/(1-(ds+1)/n )^2

plot(1:10, LOOCV, type="b", log="y")
lines(ds, GCV)
```

LOOCV and GCV



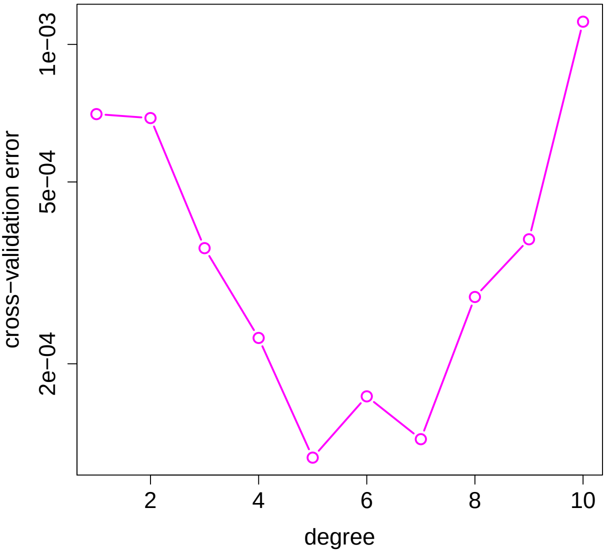
K-fold cross-validation



Source: Gareth et al. (2013), p. 179

```
set.seed(123)
KCV = sapply(1:10, function(d)
     cv.glm(train, glm(y~poly(x,degree=d),
     train, family = gaussian), K=5 )$delta[1] )

plot(1:10, KCV, type="b", log="y")
```

5-fold cross-validation



Cross-validation bias-variance trade-off

Bias

- K-fold CV with $K = 5$ or 10 gives a biased (upward) estimate of $\mathbb{E}(\mathrm{MSE}_{\mathrm{Te}})$ because it uses less information ($4/5$ or $9/10$ of the observations)
- LOOCV has very low bias (it uses $n - 1$ observations)

Variance

- LOOCV has high variance because it is an average of n extremely correlated quantities ($\hat{f}^{-i}$ and $\hat{f}^{-l}$ are fitted on $n - 2$ common observations)[1]
- K-fold CV with $K = 5$ or 10 has less variance because it is an average of quantities that are less correlated

[1] remember that $\mathbb{V}\mathrm{ar}(X + Y) = \mathbb{V}\mathrm{ar}(X) + \mathbb{V}\mathrm{ar}(Y) + 2\mathbb{C}\mathrm{ov}(X, Y)$