Chapter 7: Model Assessment and Selection

DD3364

April 20, 2012

Introduction

Regression: Review of our problem

- Have target variable Y to estimate from a vector of inputs X.
- A prediction model $\hat{f}(X)$ has been estimated from training data $\mathcal{T} = \{(x_1, y_1), \ldots, (x_n, y_n)\}$.
- The loss function $L(Y, \hat{f}(X))$ measures the errors between Y and $\hat{f}(X)$.
- Common loss functions are

$$L(Y, \hat{f}(X)) = \begin{cases} (Y - \hat{f}(X))^2 & \textbf{squared error}, \\ |Y - \hat{f}(X)| & \textbf{absolute error} \end{cases}$$

Regression: Review of our problem

- Have target variable Y to estimate from a vector of inputs X.
- A prediction model $\hat{f}(X)$ has been estimated from training data $\mathcal{T} = \{(x_1, y_1), \ldots, (x_n, y_n)\}$.
- The loss function $L(Y, \hat{f}(X))$ measures the errors between Y and $\hat{f}(X)$.
- Common loss functions are

$$L(Y, \hat{f}(X)) = \begin{cases} (Y - \hat{f}(X))^2 & \textbf{squared error}, \\ |Y - \hat{f}(X)| & \textbf{absolute error} \end{cases}$$

Regression: Review of our problem

- Have target variable Y to estimate from a vector of inputs X.
- A prediction model $\hat{f}(X)$ has been estimated from training data $\mathcal{T} = \{(x_1, y_1), \ldots, (x_n, y_n)\}$.
- The loss function $L(Y, \hat{f}(X))$ measures the errors between Y and $\hat{f}(X)$.
- Common loss functions are

$$L(Y, \hat{f}(X)) = \begin{cases} (Y - \hat{f}(X))^2 & \textbf{squared error}, \\ |Y - \hat{f}(X)| & \textbf{absolute error} \end{cases}$$

Definition of **Test Error**

Test Error a.k.a. **generalization error**

$$\boxed{\mathsf{Err}_{\mathcal{T}} = E[\, L(Y, \hat{f}(X)) \,|\, \mathcal{T} \,]}$$

- Prediction error over an independent test sample.
- X and Y are drawn randomly from $p(X, Y)$.
- The training set $\mathcal{T}$ if fixed.

Definition of **Expected Prediction Error**

Expected Prediction Error (expected test error)

$$\boxed{\text{Err} = E[\,L(Y, \hat{f}(X))\,] = E[\,\text{Err}_{\mathcal{T}}\,]}$$

- In this case take expectation over all the random quantities including the training set.

Which quantities interest us

- Would like to estimate $\text{Err}_{\mathcal{T}}$.
- But in most cases it is easier to estimate Err. Why??

Definition of **Expected Prediction Error**

Expected Prediction Error (expected test error)

$$\boxed{\text{Err} = E[\,L(Y, \hat{f}(X))\,] = E[\,\text{Err}_{\mathcal{T}}\,]}$$

- In this case take expectation over all the random quantities including the training set.

Which quantities interest us

- Would like to estimate $\text{Err}_{\mathcal{T}}$.
- But in most cases it is easier to estimate Err. Why??

Definition of **Training Error**

Training error

$$\boxed{\overline{\text{err}} = \frac{1}{n}\sum_{i=1}^{n} L(y_i, \hat{f}(x_i))}$$

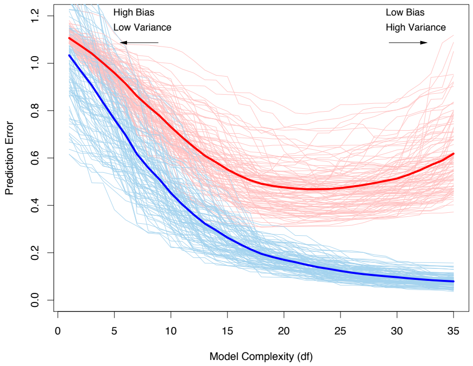
Already know as complexity of $\hat{f}$ increases

- then $\overline{\text{err}} \to 0$,
- but there is a tendency to overfit and $\text{Err}_{\mathcal{T}}$ increases
- $\overline{\text{err}}$ is not a good estimate of $\text{Err}_{\mathcal{T}}$ or Err.

We will be revisiting the **Bias-Variance** trade-off.

Issues in assessing generalization ability

Test and training error as model complexity increases.



1. Light blue curve: training error $\overline{\text{err}}$.
2. Solid blue curve: expected training error $E[\overline{\text{err}}]$.
3. Light red curve: conditional test error $\text{Err}_{\mathcal{T}}$.
4. Solid red curve: expected test error Err.

Same story for classification

- Have target categorical variable $G \in \{1, \dots, K\}$ to estimate from a vector of inputs X.
- Typically model $p_k(X) = P(G = k|X)$ and define

$$\hat{G}(X) = \arg\max_k p_k(X)$$

- Common loss functions are
  1. **0-1 loss**

$$L(G, \hat{G}(X)) = \text{Ind}(G \neq \hat{G}(X))$$

  2. **log-likelihood** a.ka. deviance

$$L(G, \hat{p}(X)) = -2 \sum_{k=1}^{K} \text{Ind}(G = k)\, \hat{p}_k(X) = -2\, \log \hat{p}_G(X)$$

Same story for classification

- Have target categorical variable $G \in \{1, \ldots, K\}$ to estimate from a vector of inputs X.
- Typically model $p_k(X) = P(G = k|X)$ and define

$$\hat{G}(X) = \arg\max_k p_k(X)$$

- Common loss functions are
    1. **0-1 loss**

$$L(G, \hat{G}(X)) = \text{Ind}(G \neq \hat{G}(X))$$

    2. **log-likelihood** a.ka. deviance

$$L(G, \hat{p}(X)) = -2 \sum_{k=1}^{K} \text{Ind}(G = k)\, \hat{p}_k(X) = -2\, \log \hat{p}_G(X)$$

Same story for classification

- Have target categorical variable $G \in \{1, \ldots, K\}$ to estimate from a vector of inputs X.

- Typically model $p_k(X) = P(G = k|X)$ and define

$$\hat{G}(X) = \arg\max_k p_k(X)$$

- Common loss functions are

1. **0-1 loss**

$$L(G, \hat{G}(X)) = \text{Ind}(G \neq \hat{G}(X))$$

2. **log-likelihood** a.ka. deviance

$$L(G, \hat{p}(X)) = -2 \sum_{k=1}^{K} \text{Ind}(G = k)\, \hat{p}_k(X) = -2\, \log \hat{p}_G(X)$$

Performance scores for classification

- **Test Error**

$$\boxed{\mathsf{Err}_{\mathcal{T}} = E[\, L(G, \hat{G}(X)) \,|\, \mathcal{T}\,]}$$

- **Training Error** one common definition

$$\boxed{\overline{\mathsf{err}} = -\frac{2}{n} \sum_{i=1}^{n} \log \hat{p}_{g_i}(x_i)}$$

Goal of this chapter

- $\hat{f}_\alpha(x)$ typically has a tunable parameter α controlling its complexity.

- Want to find the value of α s.t.

$$\boxed{\hat{\alpha} = \arg\min_{\alpha} E[\, L(Y, \hat{f}_\alpha(X)) \,]}$$

- Estimate $E[\, L(Y, \hat{f}_\alpha(X)) \,]$ for different values of α.

- This chapter presents methods how to do this.

- Choose the α with minimum estimate.

Our two separate goals

Model selection

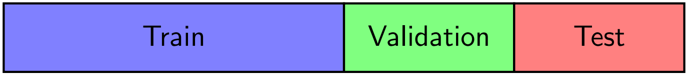
Estimate the performance of different models in order to choose the best one.

Model Assessment

Having chosen a final model, estimate its prediction error on new data.

For a data-rich situation

Randomly divide the dataset into 3 parts



Common split ratio 50%, 25%, 25%.

Model Selection

- Use **training set** to fit each model.
- Use **validation set** to estimate $\text{Err}_{\mathcal{T}}$ for each model.
- Choose model with lowest $\text{Err}_{\mathcal{T}}$ estimate.

Model Assessment of the chosen model

- Use the **test set** - unseen until this stage - to estimate $\text{Err}_{\mathcal{T}}$.

What if labelled data-sets are small ?

Approximate the **validation step** either

- analytically with approaches such as
  1. Akaike Information Criterion
  2. Baysian Information Criterion
  3. Minimum Description Length
  4. Structural Risk Minimization

  or

- with efficient sample re-use
  1. cross-validation
  2. the bootstrap

Each method also provides estimates of Err or $\text{Err}_{\mathcal{T}}$ of the final chosen model.

What if labelled data-sets are small ?

Approximate the **validation step** either

- analytically with approaches such as
  1. Akaike Information Criterion
  2. Baysian Information Criterion
  3. Minimum Description Length
  4. Structural Risk Minimization

  or

- with efficient sample re-use
  1. cross-validation
  2. the bootstrap

Each method also provides estimates of Err or $\text{Err}_{\mathcal{T}}$ of the final chosen model.

The Bias-Variance Decomposition

The bias-variance decomposition

- Will assume an additive model

$$Y = f(X) + \epsilon$$

where $\mathrm{E}[\epsilon] = 0$ and $\mathrm{Var}[\epsilon] = \sigma_\epsilon^2$.

- Then the **expected prediction error** of $\hat{f}(X)$ at $X = x_0$

$$\mathrm{Err}(x_0) = \mathrm{E}[(Y - \hat{f}(x_0))^2 | X = x_0]$$

can be expressed as

$$\boxed{\mathrm{Err}(x_0) = \text{Irreducible Error} + \text{Bias}^2 + \text{Variance}}$$

Irreducible error:	σ_ϵ^2,
Bias:	$\mathrm{E}[\hat{f}(x_0) - f(x_0)]$,
Variance:	$\mathrm{Var}[\hat{f}(x_0)]$

The bias-variance decomposition

- Will assume an additive model

$$Y = f(X) + \epsilon$$

  where $\mathrm{E}[\epsilon] = 0$ and $\mathrm{Var}[\epsilon] = \sigma_\epsilon^2$.

- Then the **expected prediction error** of $\hat{f}(X)$ at $X = x_0$

$$\mathrm{Err}(x_0) = \mathrm{E}[(Y - \hat{f}(x_0))^2 | X = x_0]$$

  can be expressed as

$$\boxed{\mathrm{Err}(x_0) = \text{Irreducible Error} + \text{Bias}^2 + \text{Variance}}$$

Irreducible error:	σ_ϵ^2,
Bias:	$\mathrm{E}[\hat{f}(x_0) - f(x_0)]$,
Variance:	$\mathrm{Var}[\hat{f}(x_0)]$

k-nearest neighbour regression fit

$$\boxed{\text{Err}(x_0) = \sigma_\epsilon^2 + \underbrace{\left[f(x_0) - \frac{1}{k}\sum_{l=1}^{k} f(x_{(l)}) \right]^2}_{\text{squared bias}} + \underbrace{\frac{\sigma_\epsilon^2}{k}}_{\text{variance}}}$$

- Complexity of model is inversely related k.
- As k increases the variance decreases.
- As k increases the squared bias increases.

The above expression was computed by assuming the x_i's are fixed.

Linear model - least square fit

Have a linear model

$$\hat{f}_p(x) = x^t \hat{\beta}$$

where $\hat{\beta}$ is p-dimensional and fit by least squares, then

$$\boxed{\text{Err}(x_0) = \sigma_\epsilon^2 + \left[f(x_0) - E[\,\hat{f}_p(x_0)\,] \right]^2 + \|h(x_0)\|^2 \, \sigma_\epsilon^2}$$

with $h(x_0) = \mathbf{X}(\mathbf{X}^t\mathbf{X})^{-1} x_0$ and $\hat{f}_p(x_0) = x_0^t(\mathbf{X}^t\mathbf{X})^{-1}\mathbf{X}^t y$.

Linear model - ridge regression fit

Have a linear model

$$\hat{f}_{p,\alpha}(x) = x^t \hat{\beta}_\alpha$$

where $\hat{\beta}_\alpha$ is p-dimensional and fit via ridge regression, then

$$\boxed{\text{Err}(x_0) = \sigma_\epsilon^2 + \left[f(x_0) - E[\,\hat{f}_{p,\alpha}(x_0)\,]\right]^2 + \|h_\alpha(x_0)\|^2\,\sigma_\epsilon^2}$$

with

$$h_\alpha(x_0) = \mathbf{X}(\mathbf{X}^t\mathbf{X} + \alpha I)^{-1} x_0$$

$$\hat{f}_{p,\alpha}(x_0) = x_0^t(\mathbf{X}^t\mathbf{X} + \alpha I)^{-1}\mathbf{X}^t\, y$$

Therefore this regression fit model has a different bias and variance to the least square fit.

Linear model - ridge regression fit

Have a linear model

$$\hat{f}_{p,\alpha}(x) = x^t \hat{\beta}_\alpha$$

where $\hat{\beta}_\alpha$ is p-dimensional and fit via ridge regression, then

$$\boxed{\text{Err}(x_0) = \sigma_\epsilon^2 + \left[f(x_0) - E[\,\hat{f}_{p,\alpha}(x_0)\,] \right]^2 + \|h_\alpha(x_0)\|^2 \, \sigma_\epsilon^2}$$

with

$$h_\alpha(x_0) = \mathbf{X}(\mathbf{X}^t\mathbf{X} + \alpha I)^{-1} x_0$$

$$\hat{f}_{p,\alpha}(x_0) = x_0^t (\mathbf{X}^t\mathbf{X} + \alpha I)^{-1} \mathbf{X}^t \, y$$

Therefore this regression fit model has a different bias and variance to the least square fit.

Linear model - Finer decomposition of the bias

Let β_* denote the parameters of the best-fitting linear approx to f:

$$\beta_* = \arg\min_{\beta} E[\,(f(X) - X^t\beta)^2\,]$$

Can write the **averaged squared bias**

$$E_{x_0}\left[(f(x_0) - E[\,\hat{f}_\alpha(x_0)\,])^2\right]$$

as

$$\boxed{\underbrace{E_{x_0}[\,(f(x_0) - x_0^t\beta_*)^2\,]}_{\text{Ave[Model Bias]}^2} + \underbrace{E_{x_0}[\,(x_0^t\beta_* - E[x_0^t\hat{\beta}_\alpha])^2\,]}_{\text{Ave[Estimation Bias]}^2}}$$

- Estimation bias is zero for ordinary least sq. estimate.
- Estimation bias is positive for ridge regression estimate.

Linear model - Finer decomposition of the bias

Let β_* denote the parameters of the best-fitting linear approx to f:

$$\beta_* = \arg\min_{\beta} E[\,(f(X) - X^t\beta)^2\,]$$

Can write the **averaged squared bias**

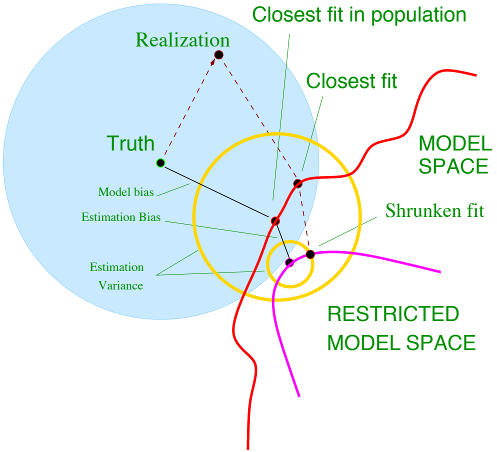
$$E_{x_0}\left[(f(x_0) - E[\,\hat{f}_\alpha(x_0)\,])^2\right]$$

as

$$\boxed{\underbrace{E_{x_0}[\,(f(x_0) - x_0^t\beta_*)^2\,]}_{\text{Ave[Model Bias]}^2} + \underbrace{E_{x_0}[\,(x_0^t\beta_* - E[x_0^t\hat{\beta}_\alpha])^2\,]}_{\text{Ave[Estimation Bias]}^2}}$$

- Estimation bias is zero for ordinary least sq. estimate.
- Estimation bias is positive for ridge regression estimate.

Behaviour of bias and variance



Bias-variance trade-off: Example 1

The Set-up

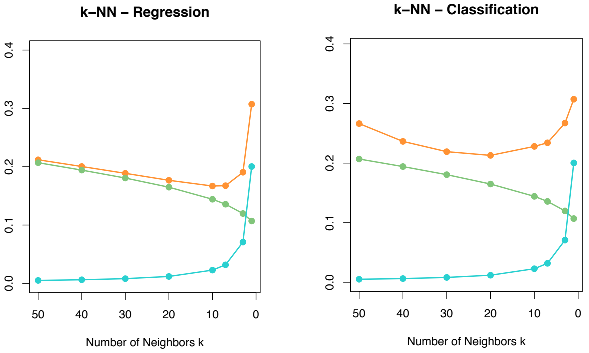
- Have $n = 80$ observations and $p = 20$ predictors.
- X is uniformly distributed in $[0, 1]^{20}$ and

$$Y = \begin{cases} 0 & \text{if } X_1 \le .5 \\ 1 & \text{if } X_1 > .5 \end{cases}$$

- Apply k-nn to perform both the classification and regression tasks.
- Use **squared error loss** to measure **Err** for the **regression task**.
- Use **0-1 loss** to measure **Err** for the **classification task**.

Bias-variance trade-off: Example 1

Expected prediction error as k varies

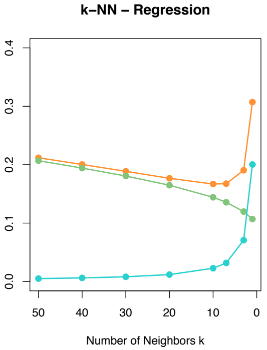


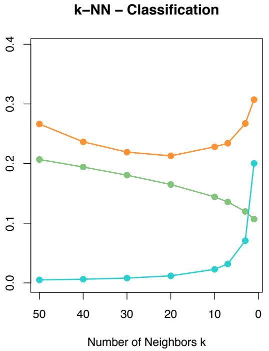
- Orange curve: **expected prediction error**
- Green curve: **squared bias**
- Blue curve: **variance**

Note prediction error curves are not the same as the loss functions differ.

Bias-variance trade-off: Example 1

Expected prediction error as k varies





- Orange curve: **expected prediction error**
- Green curve: **squared bias**
- Blue curve: **variance**

Note prediction error curves are not the same as the loss functions differ.

Bias-variance trade-off: Example 2

The Set-up

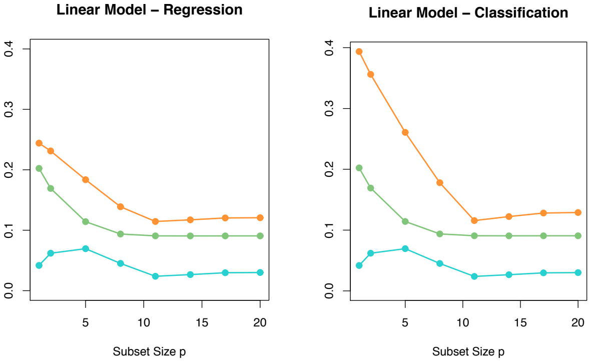
- Have $n = 80$ observations and $p = 20$ predictors.
- X is uniformly distributed in $[0,1]^{20}$ and

$$Y = \begin{cases} 1 & \text{if } \sum_{j=1}^{10} X_j > 5 \\ 0 & \text{otherwise} \end{cases}$$

- Use best subset linear regression of size p for classification and regression tasks.
- Use **squared error loss** to measure **Err** for the **regression task**.
- Use **0-1 loss** to measure **Err** for the **classification task**.

Bias-variance trade-off: Example 2

Expected prediction error as p varies



- Orange curve: **expected prediction error**
- Green curve: **squared bias**
- Blue curve: **variance**

Note prediction error curves are not the same as the loss functions differ.

Optimism of the Training Error Rate

Estimating the optimism of $\overline{\text{err}}$

- Training error $\overline{\text{err}} \ll \text{Err}_{\mathcal{T}}$ as it uses $\mathcal{T}$ for both fitting and assessment.

- **One factor is**

  The training and test input vectors

  - for $\overline{\text{err}}$ are the same.
  - while for $\text{Err}_{\mathcal{T}}$ they differ.

- Can begin to understand the optimism of $\overline{\text{err}}$ if we focus on **in-sample error**

$$\text{Err}_{\text{in}} = \frac{1}{n} \sum_{i=1}^{n} E_{Y'}[\, L(y_i', \hat{f}(x_i)) | \mathcal{T}\,]$$

  where expectation is over new responses y_i' at each training point x_i.

Estimating the optimism of $\overline{\text{err}}$

- Training error $\overline{\text{err}} \ll \text{Err}_{\mathcal{T}}$ as it uses $\mathcal{T}$ for both fitting and assessment.

- **One factor is**

  The training and test input vectors

  - for $\overline{\text{err}}$ are the same.
  - while for $\text{Err}_{\mathcal{T}}$ they differ.

- Can begin to understand the optimism of $\overline{\text{err}}$ if we focus on **in-sample error**

$$\boxed{\text{Err}_{\text{in}} = \frac{1}{n}\sum_{i=1}^{n} E_{Y'}[\,L(y_i', \hat{f}(x_i))|\mathcal{T}\,]}$$

  where expectation is over new responses y_i' at each training point x_i.

Estimating the optimism of $\overline{\text{err}}$

- Training error $\overline{\text{err}} \ll \text{Err}_{\mathcal{T}}$ as it uses $\mathcal{T}$ for both fitting and assessment.

- **One factor is**

  The training and test input vectors

  - for $\overline{\text{err}}$ are the same.
  - while for $\text{Err}_{\mathcal{T}}$ they differ.

- Can begin to understand the optimism of $\overline{\text{err}}$ if we focus on **in-sample error**

$$\boxed{\text{Err}_{\text{in}} = \frac{1}{n}\sum_{i=1}^{n} E_{Y'}[\,L(y_i', \hat{f}(x_i))|\mathcal{T}\,]}$$

  where expectation is over new responses y_i' at each training point x_i.

The optimism of $\overline{\text{err}}$

- Define the **optimism** as

$$\boxed{\text{op} = \text{Err}_{\text{in}} - \overline{\text{err}}}$$

- The **average optimism** is

$$\boxed{\omega = \text{E}_{\mathbf{y}}[\text{op}]}$$

where

  - the training input vectors are held fixed,
  - the expectation is over the training output values.

- For many loss functions

$$\boxed{\omega = \frac{1}{n}\sum_{i=1}^{n} \text{Cov}(\hat{y}_i, y_i)}$$

The optimism of $\overline{\text{err}}$

$$\boxed{\omega = \frac{1}{n}\sum_{i=1}^{n} \text{Cov}(\hat{y}_i, y_i)}$$

- The more strongly y_i affects its prediction $\hat{y}_i$ the larger ω.
- The larger ω the greater the optimism of $\overline{\text{err}}$.
- In summary get the important relation

$$\boxed{\text{E}_{\mathbf{y}}[\text{Err}_{\text{in}}] = \text{E}_{\mathbf{y}}[\overline{\text{err}}] + \frac{1}{n}\sum_{i=1}^{n} \text{Cov}(\hat{y}_i, y_i)}$$

How to estimate prediction error?

Option 1

- Estimate the optimism and add it to $\overline{\text{err}}$
- The methods $C_p, \text{AIC}, \text{BIC}$ work in this way for a special class estimates.
- Can use **in-sample** error for model selection but not a good estimate of Err.

How to estimate prediction error?

Option 1

- Estimate the optimism and add it to $\overline{\text{err}}$
- The methods $C_p, \text{AIC}, \text{BIC}$ work in this way for a special class estimates.
- Can use **in-sample** error for model selection but not a good estimate of Err.

Option 2

- Use cross-validation and bootstrap as direct estimates of the **extra-sample** Err.

Estimates of In-Sample Prediction Error

Err_{in} estimate: C_p statistic

- If $\hat{y}_i$ is obtained by a linear fit with d inputs then

$$\sum_{i=1}^{n} \text{Cov}(\hat{y}_i, y_i) = d\,\sigma_\epsilon^2$$

for the additive error model $Y = f(X) + \epsilon$.

- And so

$$\boxed{\text{E}_{\mathbf{y}}[\text{Err}_{\text{in}}] = \text{E}_{\mathbf{y}}[\overline{\text{err}}] + 2\,\frac{d}{n}\sigma_\epsilon^2}$$

- Adapting this expression leads to the C_p statistic

$$\boxed{C_p = \overline{\text{err}} + 2\,\frac{d}{n}\hat{\sigma}_\epsilon^2}$$

where $\hat{\sigma}_\epsilon^2$ is an estimate of the noise variance.

Err_{in} estimate: C_p statistic

- If $\hat{y}_i$ is obtained by a linear fit with d inputs then

$$\sum_{i=1}^{n} \text{Cov}(\hat{y}_i, y_i) = d\,\sigma_\epsilon^2$$

  for the additive error model $Y = f(X) + \epsilon$.

- And so

$$\boxed{\text{E}_{\mathbf{y}}[\text{Err}_{\text{in}}] = \text{E}_{\mathbf{y}}[\overline{\text{err}}] + 2\,\frac{d}{n}\sigma_\epsilon^2}$$

- Adapting this expression leads to the C_p statistic

$$\boxed{C_p = \overline{\text{err}} + 2\,\frac{d}{n}\hat{\sigma}_\epsilon^2}$$

  where $\hat{\sigma}_\epsilon^2$ is an estimate of the noise variance.

Err_{in} estimate: C_p statistic

- If $\hat{y}_i$ is obtained by a linear fit with d inputs then

$$\sum_{i=1}^{n} \text{Cov}(\hat{y}_i, y_i) = d\,\sigma_\epsilon^2$$

  for the additive error model $Y = f(X) + \epsilon$.

- And so

$$\boxed{\text{E}_{\mathbf{y}}[\text{Err}_{\text{in}}] = \text{E}_{\mathbf{y}}[\overline{\text{err}}] + 2\,\frac{d}{n}\sigma_\epsilon^2}$$

- Adapting this expression leads to the C_p statistic

$$\boxed{C_p = \overline{\text{err}} + 2\,\frac{d}{n}\hat{\sigma}_\epsilon^2}$$

  where $\hat{\sigma}_\epsilon^2$ is an estimate of the noise variance.

Akaike Information Criterion

$$\boxed{\text{AIC} = \overset{\text{rewards the fit between the model and the data}}{-\frac{2}{n}\text{loglik}} + \underset{\text{penalty for including extra predictors in the model}}{2\frac{d}{n}}}$$

where

$$\text{loglik} = \sum_{i=1}^{n} \log P_{\hat{\theta}}(y_i)$$

and $\hat{\theta}$ is the MLE of θ.

Note: AIC can be seen as an estimate of $\mathrm{Err}_{\mathsf{in}}$ in this case with a log-likelihood loss.

Akaike Information Criterion: using training error

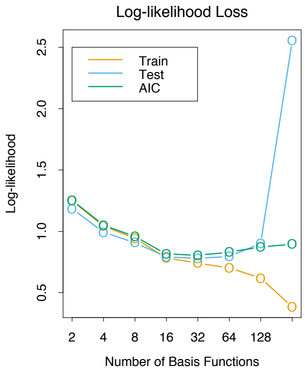
- Have a set of models $f_\alpha(x)$ indexed by α,
- $\overline{\text{err}}$ is the training error,
- $d(\alpha)$ the # of parameters for each model.

then

$$\boxed{\text{AIC}(\alpha) = \overline{\text{err}}(\alpha) + 2\frac{d(\alpha)}{n}\hat{\sigma}_\epsilon^2}$$

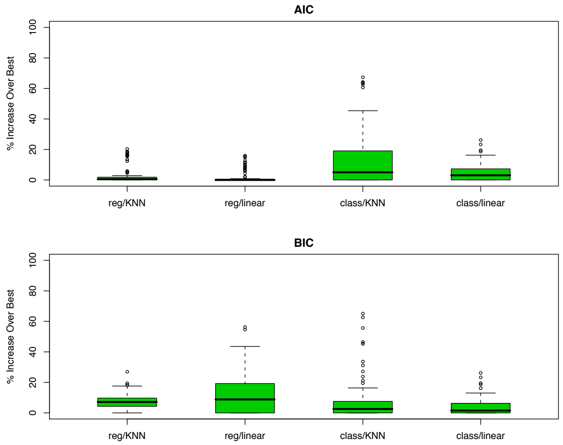
Note: AIC can be seen as an estimate of Err_{in} in this case with a squared-error loss.

AIC used for model selection



- Classifier is a logistic regression function with an expansion of M spline basis functions.
- AIC is used to estimate Err_{in} with a log-likelihood loss,
- AIC does well except when $M = 256$ is large and $n = 1000$.

How well AIC and BIC perform wrt model selection



- Boxplots show $100 \frac{\text{Err}(\hat{\alpha}) - \min_\alpha \text{Err}(\alpha)}{\max_\alpha \text{Err}(\alpha) - \min_\alpha \text{Err}(\alpha)}$ where $\hat{\alpha}$ is the best parameter found via the selection method under investigation.
- 100 training sets were used.

The Effective Number of Parameters

Generalization of the *number of parameters*

- For **regularized fitting** need to generalize the concept of *number of parameters*.
- Consider regularized linear fitting - ridge regression, cubic smoothing splines

$$\boxed{\hat{y} = \mathbf{S}\, y}$$

where

1. $y = (y_1, y_2, \ldots, y_n)^t$ is the vector of training outputs,
2. $\hat{y} = (\hat{y}_1, \ldots, \hat{y}_n)$ is the vector of predictions,
3. $\mathbf{S}$ is an $n \times n$ matrix - depends on $x_1, \ldots, x_n$ but not $y_1, \ldots, y_n$.

- Then the **effective number of parameters** is defined as

$$\boxed{\text{df}(\mathbf{S}) = \text{trace}(\mathbf{S})}$$

Generalization of the *number of parameters*

- For **regularized fitting** need to generalize the concept of *number of parameters*.

- Consider regularized linear fitting - ridge regression, cubic smoothing splines

$$\boxed{\hat{y} = \mathbf{S}\, y}$$

where

1. $y = (y_1, y_2, \ldots, y_n)^t$ is the vector of training outputs,
2. $\hat{y} = (\hat{y}_1, \ldots, \hat{y}_n)$ is the vector of predictions,
3. $\mathbf{S}$ is an $n \times n$ matrix - depends on $x_1, \ldots, x_n$ but not $y_1, \ldots, y_n$.

- Then the **effective number of parameters** is defined as

$$\boxed{\text{df}(\mathbf{S}) = \text{trace}(\mathbf{S})}$$

General definition: Effective degrees-of-freedom

- If y arises from an additive-error model

$$Y = f(X) + \epsilon$$

  with $\mathsf{Var}(\epsilon) = \sigma_\epsilon^2$ then

$$\sum_{i=1}^{n} \mathsf{Cov}(\hat{y}_i, y_i) = \mathsf{trace}(\mathbf{S})\sigma_\epsilon^2$$

- The more general definition of **effective dof** is then

$$\boxed{\mathsf{df}(\hat{y}) = \frac{\sum_{i=1}^{n} \mathsf{Cov}(\hat{y}_i, y_i)}{\sigma_\epsilon^2}}$$

General definition: Effective degrees-of-freedom

- If y arises from an additive-error model

$$Y = f(X) + \epsilon$$

with $\text{Var}(\epsilon) = \sigma_\epsilon^2$ then

$$\sum_{i=1}^{n} \text{Cov}(\hat{y}_i, y_i) = \text{trace}(\mathbf{S})\sigma_\epsilon^2$$

- The more general definition of **effective dof** is then

$$\boxed{\text{df}(\hat{y}) = \frac{\sum_{i=1}^{n} \text{Cov}(\hat{y}_i, y_i)}{\sigma_\epsilon^2}}$$

The Bayesian Approach and BIC

Generic form of BIC

Bayesian Information Criterion

$$\boxed{\text{BIC} = -2\,\text{loglik} + \log(n)\,d}$$

- Assuming **Gaussian model** and known variance σ_ϵ^2 then

$$-2\,\text{loglik} = \frac{1}{\sigma_\epsilon^2}\sum_{i=1}^{n}(y_i - \hat{h}(x_i))^2 = \frac{n\,\overline{\text{err}}}{\sigma_\epsilon^2}$$

  and

$$\boxed{\text{BIC} = \frac{n}{\sigma_\epsilon^2}\left(\overline{\text{err}} + \log(n)\,\frac{d}{n}\sigma_\epsilon^2\right)}$$

- Note BIC $\propto$ AIC, but BIC penalizes complex model more heavily than AIC.

Generic form of BIC

Bayesian Information Criterion

$$\boxed{\mathsf{BIC} = -2\,\mathsf{loglik} + \log(n)\,d}$$

- Assuming **Gaussian model** and known variance σ_ϵ^2 then

$$-2\,\mathsf{loglik} = \frac{1}{\sigma_\epsilon^2}\sum_{i=1}^{n}(y_i - \hat{h}(x_i))^2 = \frac{n\,\overline{\mathsf{err}}}{\sigma_\epsilon^2}$$

  and

$$\boxed{\mathsf{BIC} = \frac{n}{\sigma_\epsilon^2}\left(\overline{\mathsf{err}} + \log(n)\,\frac{d}{n}\sigma_\epsilon^2\right)}$$

- Note BIC $\propto$ AIC, but BIC penalizes complex model more heavily than AIC.

Derivation of BIC

- **Starting point:**

  Have $\{\mathcal{M}_1, \dots, \mathcal{M}_M\}$ a set of candidate models and their corresponding parameters $\theta_1, \dots, \theta_m$.

- **Goal:**

  Choose the best model $\mathcal{M}_i$.

- **How:**

  - Have training data $\mathbf{Z} = \{(x_1, y_1), \dots, (x_n, y_n)\}$
  - Have priors $p(\theta_m|\mathcal{M}_m)$.
  - The posterior of model $\mathcal{M}_m$ is

$$
\begin{aligned}
P(\mathcal{M}_m|\mathbf{Z}) &\propto P(\mathcal{M}_m)\, p(\mathbf{Z}|\mathcal{M}_m) \\
&\propto P(\mathcal{M}_m) \int p(\mathbf{Z}|\theta_m, \mathcal{M}_m)\, p(\theta_m|\mathcal{M}_m)\, d\theta_m
\end{aligned}
$$

Derivation of BIC

- **Starting point:**

  Have $\{\mathcal{M}_1, \ldots, \mathcal{M}_M\}$ a set of candidate models and their corresponding parameters $\theta_1, \ldots, \theta_m$.

- **Goal:**

  Choose the best model $\mathcal{M}_i$.

- **How:**

  - Have training data $\mathbf{Z} = \{(x_1, y_1), \ldots, (x_n, y_n)\}$
  - Have priors $p(\theta_m | \mathcal{M}_m)$.
  - The posterior of model $\mathcal{M}_m$ is

$$\begin{aligned} P(\mathcal{M}_m | \mathbf{Z}) &\propto P(\mathcal{M}_m)\, p(\mathbf{Z} | \mathcal{M}_m) \\ &\propto P(\mathcal{M}_m) \int p(\mathbf{Z} | \theta_m, \mathcal{M}_m)\, p(\theta_m | \mathcal{M}_m)\, d\theta_m \end{aligned}$$

Derivation of BIC

- **Starting point:**

  Have $\{\mathcal{M}_1, \ldots, \mathcal{M}_M\}$ a set of candidate models and their corresponding parameters $\theta_1, \ldots, \theta_m$.

- **Goal:**

  Choose the best model $\mathcal{M}_i$.

- **How:**

  - Have training data $\mathbf{Z} = \{(x_1, y_1), \ldots, (x_n, y_n)\}$
  - Have priors $p(\theta_m | \mathcal{M}_m)$.
  - The posterior of model $\mathcal{M}_m$ is

$$
\begin{aligned}
P(\mathcal{M}_m | \mathbf{Z}) &\propto P(\mathcal{M}_m)\, p(\mathbf{Z} | \mathcal{M}_m) \\
&\propto P(\mathcal{M}_m) \int p(\mathbf{Z} | \theta_m, \mathcal{M}_m)\, p(\theta_m | \mathcal{M}_m)\, d\theta_m
\end{aligned}
$$

Derivation of BIC

- The posterior of model $\mathcal{M}_m$ is

$$P(\mathcal{M}_m|\theta_m) \propto P(\mathcal{M}_m) \int p(\mathbf{Z}|\theta_m, \mathcal{M}_m)\, p(\theta_m|\mathcal{M}_m)\, d\theta_m$$

- Usually assume uniform prior: $P(\mathcal{M}_m) = 1/M$.
- Approximate the above integral by simplification and Laplace approximation to get

$$\log P(\mathbf{Z}|\mathcal{M}_m) = \log P(\mathbf{Z}|\hat{\theta}_m, \mathcal{M}_m) - \frac{d_m}{2} \log n + O(1)$$

  where $\hat{\theta}_m$ is a MLE and d_m is # free parameters in $\mathcal{M}_m$.
- Then BIC $\propto -2 \log P(\mathcal{M}_m|\mathbf{Z})$

Derivation of BIC

- The posterior of model $\mathcal{M}_m$ is

$$P(\mathcal{M}_m|\theta_m) \propto P(\mathcal{M}_m) \int p(\mathbf{Z}|\theta_m, \mathcal{M}_m)\, p(\theta_m|\mathcal{M}_m)\, d\theta_m$$

- Usually assume uniform prior: $P(\mathcal{M}_m) = 1/M$.

- Approximate the above integral by simplification and Laplace approximation to get

$$\log P(\mathbf{Z}|\mathcal{M}_m) = \log P(\mathbf{Z}|\hat{\theta}_m, \mathcal{M}_m) - \frac{d_m}{2} \log n + O(1)$$

  where $\hat{\theta}_m$ is a MLE and d_m is # free parameters in $\mathcal{M}_m$.

- Then BIC $\propto -2 \log P(\mathcal{M}_m|\mathbf{Z})$

Derivation of BIC

- The posterior of model $\mathcal{M}_m$ is

$$P(\mathcal{M}_m|\theta_m) \propto P(\mathcal{M}_m) \int p(\mathbf{Z}|\theta_m, \mathcal{M}_m)\, p(\theta_m|\mathcal{M}_m)\, d\theta_m$$

- Usually assume uniform prior: $P(\mathcal{M}_m) = 1/M$.

- Approximate the above integral by simplification and Laplace approximation to get

$$\log P(\mathbf{Z}|\mathcal{M}_m) = \log P(\mathbf{Z}|\hat{\theta}_m, \mathcal{M}_m) - \frac{d_m}{2} \log n + O(1)$$

  where $\hat{\theta}_m$ is a MLE and d_m is # free parameters in $\mathcal{M}_m$.

- Then BIC $\propto -2 \log P(\mathcal{M}_m|\mathbf{Z})$

Derivation of BIC

- The posterior of model $\mathcal{M}_m$ is

$$P(\mathcal{M}_m|\theta_m) \propto P(\mathcal{M}_m) \int p(\mathbf{Z}|\theta_m, \mathcal{M}_m)\, p(\theta_m|\mathcal{M}_m)\, d\theta_m$$

- Usually assume uniform prior: $P(\mathcal{M}_m) = 1/M$.

- Approximate the above integral by simplification and Laplace approximation to get

$$\log P(\mathbf{Z}|\mathcal{M}_m) = \log P(\mathbf{Z}|\hat{\theta}_m, \mathcal{M}_m) - \frac{d_m}{2}\, \log n + O(1)$$

  where $\hat{\theta}_m$ is a MLE and d_m is $\#$ free parameters in $\mathcal{M}_m$.

- Then BIC $\propto -2 \log P(\mathcal{M}_m|\mathbf{Z})$

AIC Vs BIC

- If $\mathcal{M}_{\text{true}} \in \{\mathcal{M}_1, \ldots, \mathcal{M}_M\}$ then as $n \to \infty$
  - BIC will select $\mathcal{M}_{\text{true}}$. ✓
  - AIC will not. It tends to choose too complex models as $n \to \infty$. ✗
- However, when n is small
  - BIC often chooses models which are too simple. ✗

AIC Vs BIC

- If $\mathcal{M}_{\text{true}} \in \{\mathcal{M}_1, \ldots, \mathcal{M}_M\}$ then as $n \to \infty$
  - BIC will select $\mathcal{M}_{\text{true}}$. ✓
  - AIC will not. It tends to choose too complex models as $n \to \infty$. ✗
- However, when n is small
  - BIC often chooses models which are too simple. ✗

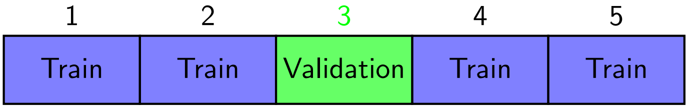
Cross-Validation

K-Fold Cross-Validation

K-Fold Cross-Validation

General Approach

- Split the data into K roughly equal-size parts.



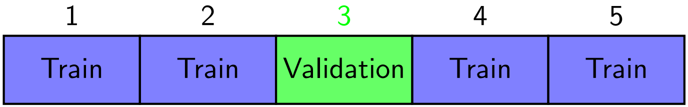
- For the kth part calculate the prediction error of the model fit using the other $K-1$ parts.

- Do this for $k = 1, 2, \ldots, K$ and combine the K estimates of the prediction error.

K-Fold Cross-Validation

General Approach

- Split the data into K roughly equal-size parts.



- For the kth part calculate the prediction error of the model fit using the other $K-1$ parts.

- Do this for $k = 1, 2, \ldots, K$ and combine the K estimates of the prediction error.

When and why

- It is applied when labelled training data is relatively sparse.

- This method directly estimates $\text{Err} = \mathsf{E}[L(Y, \hat{f}(X))]$.

K-Fold Cross-validation: Detailed description

- The mapping $\kappa : \{1, \ldots, n\} \to \{1, \ldots, K\}$ indicates observation i belongs to partition $\kappa(i)$.

- $\hat{f}^{-k}(x)$ is the function fitted with the kth part of the data removed.

- **Cross-validation** estimate of the prediction error is

$$\boxed{\mathrm{CV}(\hat{f}) = \frac{1}{n} \sum_{i=1}^{n} L(y_i, \hat{f}^{-\kappa(i)}(x_i))}$$

- Typical choices for K are 5 or 10.

- The case $K = n$ is known as leave-one-out cross-validation.

K-Fold Cross-validation: Detailed description

- The mapping $\kappa : \{1, \ldots, n\} \to \{1, \ldots, K\}$ indicates observation i belongs to partition $\kappa(i)$.
- $\hat{f}^{-k}(x)$ is the function fitted with the kth part of the data removed.
- **Cross-validation** estimate of the prediction error is

$$\boxed{\mathrm{CV}(\hat{f}) = \frac{1}{n} \sum_{i=1}^{n} L(y_i, \hat{f}^{-\kappa(i)}(x_i))}$$

- Typical choices for K are 5 or 10.
- The case $K = n$ is known as leave-one-out cross-validation.

K-Fold Cross-validation: Detailed description

- The mapping $\kappa : \{1, \ldots, n\} \to \{1, \ldots, K\}$ indicates observation i belongs to partition $\kappa(i)$.
- $\hat{f}^{-k}(x)$ is the function fitted with the kth part of the data removed.
- **Cross-validation** estimate of the prediction error is

$$\boxed{\mathrm{CV}(\hat{f}) = \frac{1}{n} \sum_{i=1}^{n} L(y_i, \hat{f}^{-\kappa(i)}(x_i))}$$

- Typical choices for K are 5 or 10.
- The case $K = n$ is known as leave-one-out cross-validation.

K-Fold Cross-validation: Model selection

- Have models $f(x, \alpha)$ indexed by a parameter α.
- $\hat{f}^{-k}(x, \alpha)$ is αth model fit with kth part of the data removed.
- Then define

$$\boxed{\mathrm{CV}(\hat{f}, \alpha) = \frac{1}{n} \sum_{i=1}^{n} L(y_i, \hat{f}^{-\kappa(i)}(x_i, \alpha))}$$

- Choose the model

$$\hat{\alpha} = \arg\min_{\alpha} \mathrm{CV}(\hat{f}, \alpha)$$

K-Fold Cross-validation: Model selection

- Have models $f(x, \alpha)$ indexed by a parameter α.
- $\hat{f}^{-k}(x, \alpha)$ is αth model fit with kth part of the data removed.
- Then define

$$\boxed{\mathrm{CV}(\hat{f}, \alpha) = \frac{1}{n} \sum_{i=1}^{n} L(y_i, \hat{f}^{-\kappa(i)}(x_i, \alpha))}$$

- Choose the model

$$\hat{\alpha} = \arg\min_{\alpha} \mathrm{CV}(\hat{f}, \alpha)$$

What quantity does K-fold validation estimate?

Intuition says

- When $K = 5$ or 10 then $CV(\hat{f}) \approx \mathrm{Err}$ as training sets for each fold are fairly different.

- When $K = n$ then $CV(\hat{f}) \approx \mathrm{Err}_{\mathcal{T}}$ as training sets for each fold are almost identical.

What quantity does K-fold validation estimate?

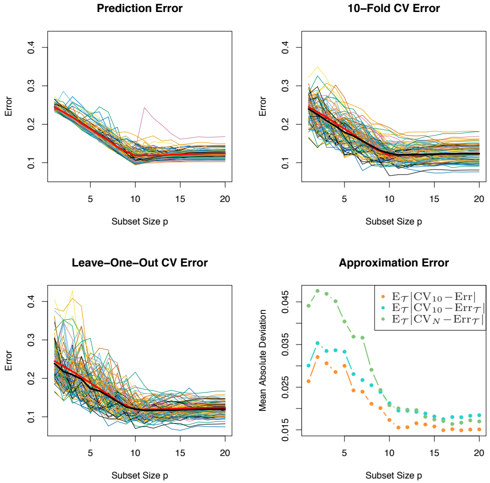
Intuition says

- When $K = 5$ or 10 then $CV(\hat{f}) \approx \mathrm{Err}$ as training sets for each fold are fairly different.
- When $K = n$ then $CV(\hat{f}) \approx \mathrm{Err}_{\mathcal{T}}$ as training sets for each fold are almost identical.

Book's simulation experiments say

- Cross-validation, really only effectively estimates Err.

What quantity does K-fold validation estimate?



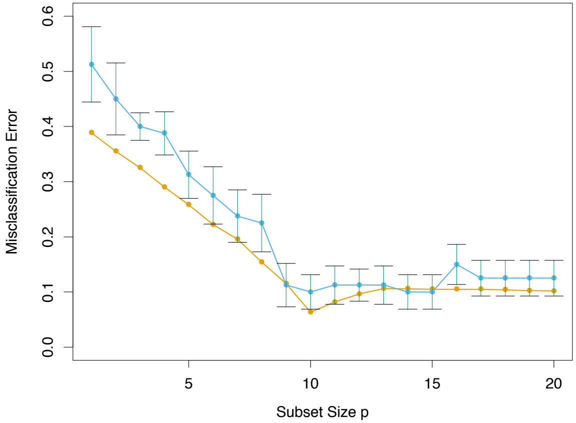
- **Thick red curve: $\mathbf{Err}$**
- **Thick black curve: $\mathrm{E}_{\mathcal{T}}[\mathrm{CV}_K]$**

What value of K?

- When $K = n$
  - $CV(\hat{f})$ is approx an unbiased estimate of Err. ✓
  - $CV(\hat{f})$ has high variance as the n *training sets* are similar. ✗
  - Computational burden is high. ✗ (except for a few exceptions)
- When $K = 5$ (is lowish)
  - $CV(\hat{f})$ has low variance. ✓
  - $CV(\hat{f})$ is potentially an upward biased estimate of Err. ✗

    Only occurs if at each fold there is not enough training data to fit a good model.

Example of a K-fold cross validation curve



- Orange curve: $\mathrm{Err}_{\mathcal{T}}$
- Blue curve: $\mathrm{CV}_{10}(\hat{f})$

Right & Wrong way to do Cross-validation

Classification problem with a large # of predictors

What's wrong with this strategy?

1. Screen the predictors Find a subset of *good* predictors that are correlated with the class labels.
2. Build a classifier based on the subset of good predictors.
3. Perform cross-validation to estimate the unknown tuning parameters and to estimate Err of the final model.

Classification problem with a large # of predictors

What's wrong with this strategy?

1. Screen the predictors Find a subset of *good* predictors that are correlated with the class labels.
2. Build a classifier based on the subset of good predictors.
3. Perform cross-validation to estimate the unknown tuning parameters and to estimate Err of the final model.

Classification problem with a large # of predictors

What's wrong with this strategy?

1. Screen the predictors Find a subset of *good* predictors that are correlated with the class labels.
2. Build a classifier based on the subset of good predictors.
3. Perform cross-validation to estimate the unknown tuning parameters and to estimate Err of the final model.

Classification problem with a large # of predictors

What's wrong with this strategy?

1. Screen the predictors Find a subset of *good* predictors that are correlated with the class labels.
2. Build a classifier based on the subset of good predictors.
3. Perform cross-validation to estimate the unknown tuning parameters and to estimate Err of the final model.

The good predictors were chosen after seeing all the data.

Should have done this

1. Divide the samples into K groups randomly.

2. For each fold $k = 1, \ldots, K$

    - Find a subset of *good* predictors using all the samples minus the kth fold.

    - Build a classifier using all the samples minus the kth fold.

    - Use the classifier to predict the labels for the samples in the kth fold.

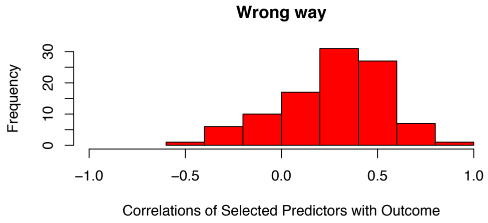
Example

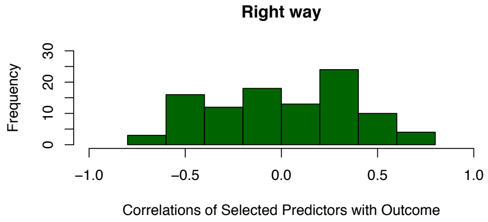
Set-up

- Have a binary classification problem.
- $n = 50$ with an equal number of points from each class.
- Have $p = 5000$ quantitative predictors that are independent of the class labels.
- The true error rate of any classifier is 50%.

✗ If one performs pre-selection of 100 predictors and then builds a 1-nn classifier the average CV error rate was 3% over 50 simulations ! ✗

Example: correlation of class labels with predictors





To perform Multistep Modelling

- Cross-validation must be applied to the entire sequence of modelling steps.
- Samples must be **left out** before any selection or filtering is applied which uses the labels.
- One exception: An unsupervised screening step can use all the samples.

To perform Multistep Modelling

- Cross-validation must be applied to the entire sequence of modelling steps.
- Samples must be **left out** before any selection or filtering is applied which uses the labels.
- One exception: An unsupervised screening step can use all the samples.

Bootstrap Method

- Have a training set $\mathbf{Z} = (z_1, z_2, \ldots, z_n)$ with each $z_i = (x_i, y_i)$.
- The **bootstrap** idea is

```
for $b = 1, 2, \ldots, B$
    1. Randomly draw $n$ samples with replacement from $\mathbf{Z}$ to get $\mathbf{Z}^{*b}$ that is
           $\mathbf{Z}^{*b} = (z_{b_1}, z_{b_2}, \ldots, z_{b_n})$   with $b_i \in \{1, \ldots, n\}$
    2. Refit the model using $\mathbf{Z}^{*b}$ to get $S(\mathbf{Z}^{*b})$
Examine the behaviour of the $B$ fits
    $S(\mathbf{Z}^{*1}), S(\mathbf{Z}^{*2}), \ldots, S(\mathbf{Z}^{*B})$.
```

The Bootstrap

- Have a training set $\mathbf{Z} = (z_1, z_2, \dots, z_n)$ with each $z_i = (x_i, y_i)$.

- The **bootstrap** idea is

```
for $b = 1, 2, \dots, B$
    1. Randomly draw $n$ samples with replacement from $\mathbf{Z}$ to get $\mathbf{Z}^{*b}$ that is
           $\mathbf{Z}^{*b} = (z_{b_1}, z_{b_2}, \dots, z_{b_n})$   with $b_i \in \{1, \dots, n\}$
    2. Refit the model using $\mathbf{Z}^{*b}$ to get $S(\mathbf{Z}^{*b})$
Examine the behaviour of the $B$ fits
    $S(\mathbf{Z}^{*1}), S(\mathbf{Z}^{*2}), \dots, S(\mathbf{Z}^{*B})$.
```

Can estimate any aspect of the distribution of $S(\mathbf{Z})$

- For example its variance

$$\boxed{\widehat{\mathrm{Var}}[S(\mathbf{Z})] = \frac{1}{B-1}\sum_{b=1}^{B}(S(Z^{*b}) - \bar{S}^{*})^2}$$

where

$$\bar{S}^{*} = \frac{1}{B}\sum_{b=1}^{B} S(Z^{*b})$$

Use Bootstrap to estimate Prediction Error

Attempt 1

$$\boxed{\widehat{\text{Err}}_{\text{boot}} = \frac{1}{B}\frac{1}{n}\sum_{b=1}^{B}\sum_{i=1}^{n} L(y_i, \hat{f}^{*b}(x_i))}$$

where $\hat{f}^{*b}(x_i)$ is the predicted value at x_i using the model computed from $\mathbf{Z}^{*b}$.

- Why is this not a good estimate??
  - Overlap between training and test sets
- How could we do better?
  - Mimic cross-validation

Use Bootstrap to estimate Prediction Error

Attempt 1

$$\boxed{\widehat{\text{Err}}_{\text{boot}} = \frac{1}{B}\frac{1}{n}\sum_{b=1}^{B}\sum_{i=1}^{n} L(y_i, \hat{f}^{*b}(x_i))}$$

where $\hat{f}^{*b}(x_i)$ is the predicted value at x_i using the model computed from $\mathbf{Z}^{*b}$.

- Why is this not a good estimate??
  - Overlap between training and test sets
- How could we do better?
  - Mimic cross-validation

Use Bootstrap to estimate Prediction Error

Attempt 1

$$\boxed{\widehat{\text{Err}}_{\text{boot}} = \frac{1}{B}\frac{1}{n}\sum_{b=1}^{B}\sum_{i=1}^{n} L(y_i, \hat{f}^{*b}(x_i))}$$

where $\hat{f}^{*b}(x_i)$ is the predicted value at x_i using the model computed from $\mathbf{Z}^{*b}$.

- Why is this not a good estimate??
  - Overlap between training and test sets
- How could we do better?
  - Mimic cross-validation

Use Bootstrap to estimate Prediction Error

Attempt 1

$$\boxed{\widehat{\text{Err}}_{\text{boot}} = \frac{1}{B}\frac{1}{n}\sum_{b=1}^{B}\sum_{i=1}^{n} L(y_i, \hat{f}^{*b}(x_i))}$$

where $\hat{f}^{*b}(x_i)$ is the predicted value at x_i using the model computed from $\mathbf{Z}^{*b}$.

- Why is this not a good estimate??
  - Overlap between training and test sets
- How could we do better?
  - Mimic cross-validation

Use Bootstrap to estimate Prediction Error

Attempt 2: Leave-one-out bootstrap

$$\boxed{\widehat{\text{Err}}^{(1)} = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{|C^{-i}|}\sum_{b\in C^{-i}} L(y_i, \hat{f}^{*b}(x_i))}$$

where C^{-i} is the set of bootstrap samples b not containing observation i.

- Either make
- Make B large enough so $|C^{-i}| > 0$ for all i **or**
- Omit observation i from testing if $|C^{-i}| = 0$.

Use Bootstrap to estimate Prediction Error

Attempt 2: Leave-one-out bootstrap

$$\boxed{\widehat{\mathrm{Err}}^{(1)} = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{|C^{-i}|}\sum_{b\in C^{-i}} L(y_i, \hat{f}^{*b}(x_i))}$$

where C^{-i} is the set of bootstrap samples b not containing observation i.

- Either make
- Make B large enough so $|C^{-i}| > 0$ for all i **or**
- Omit observation i from testing if $|C^{-i}| = 0$.

Is the leave-one-out prediction any good?

- **Pros**:
  1. avoids the overfitting problem of $\widehat{\text{Err}}_{\text{boot}}$
- **Cons**:
  1. Has the training-set-size bias of cross-validation
  2. The $P(\text{observation } i \in \mathbf{Z}^{*b})$ is
     $$1 - \left(1 - \frac{1}{n}\right)^n \approx 1 - e^{-1} = .632$$
     Therefore the average number of distinct observations in $\mathbf{Z}^{*b}$ is $.632\, n$.
  3. $\widehat{\text{Err}}^{(1)}$'s bias is thus similar to twofold cross-validation.

To alleviate this bias

Attempt 3: The .632 estimator

$$\boxed{\widehat{\text{Err}}^{(.632)} = .368\,\overline{\mathsf{err}} + .632\,\widehat{\text{Err}}^{(1)}}$$

- Compromise between the **training error** $\overline{\mathsf{err}}$ and the **leave-one-out bootstrap** estimate.
- Its derivation is not easy.
- Obviously the constant .632 relates to $P(\text{observation } i \in \mathbf{Z}^{*b})$.

The .632 estimator **does not do well** if predictor **overfits**.

To alleviate this bias

Attempt 3: The .632 estimator

$$\boxed{\widehat{\mathrm{Err}}^{(.632)} = .368\,\overline{\mathsf{err}} + .632\,\widehat{\mathrm{Err}}^{(1)}}$$

- Compromise between the **training error** $\overline{\mathsf{err}}$ and the **leave-one-out bootstrap** estimate.
- Its derivation is not easy.
- Obviously the constant .632 relates to $P(\text{observation } i \in \mathbf{Z}^{*b})$.

The .632 estimator **does not do well** if predictor **overfits**.

Estimate the degree of overfitting

No-information error rate:

$$\boxed{\hat{\gamma} = \frac{1}{n^2} \sum_{i=1}^{n} \sum_{j=1}^{n} L(y_i, \hat{f}(x_j))}$$

- Estimate of the error rate of $\hat{f}$ if inputs and outputs were independent.
- Note the prediction rule, $\hat{f}$, is evaluated on all possible combinations of targets y_i and predictors x_j.

Estimate the degree of overfitting

No-information error rate:

$$\boxed{\hat{\gamma} = \frac{1}{n^2} \sum_{i=1}^{n} \sum_{j=1}^{n} L(y_i, \hat{f}(x_j))}$$

- Estimate of the error rate of $\hat{f}$ if inputs and outputs were independent.
- Note the prediction rule, $\hat{f}$, is evaluated on all possible combinations of targets y_i and predictors x_j.

Estimate the degree of overfitting

Relative overfitting rate:

$$\boxed{\hat{R} = \frac{\widehat{\mathrm{Err}}^{(1)} - \overline{\mathsf{err}}}{\hat{\gamma} - \overline{\mathsf{err}}}}$$

- $0 \leq \hat{R} \leq 1$
- $\hat{R} = 0 \implies$ no overfitting.
- $\hat{R} = 1 \implies$ overfitting equals no-information value $\hat{\gamma} - \overline{\mathsf{err}}$.

Use Bootstrap to estimate Prediction Error

Attempt 4: The .632+ estimator

$$\boxed{\widehat{\text{Err}}^{(.632+)} = (1 - \hat{w})\,\overline{\text{err}} + \hat{w}\,\widehat{\text{Err}}^{(1)}}$$

with

$$\hat{w} = \frac{.632}{1 - .368\hat{R}}$$

- $.632 \le \hat{w} \le 1$ as $\hat{R}$ ranges from 0 to 1.
- $\widehat{\text{Err}}^{(.632+)}$ ranges from $\widehat{\text{Err}}^{(.632)}$ to $\widehat{\text{Err}}^{(1)}$.
- $\widehat{\text{Err}}^{(.632+)}$ is a compromise between $\widehat{\text{Err}}^{(.632)}$ and $\overline{\text{err}}$ that depends on the amount of overfitting.
- Derivation of the above eqn is non-trivial.

Use Bootstrap to estimate Prediction Error

Attempt 4: The .632+ estimator

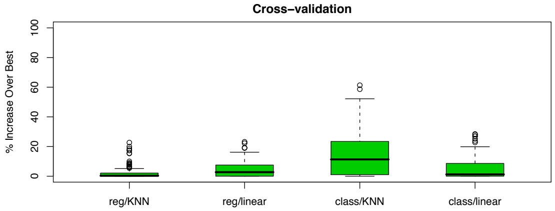
$$\boxed{\widehat{\text{Err}}^{(.632+)} = (1-\hat{w})\,\overline{\text{err}} + \hat{w}\,\widehat{\text{Err}}^{(1)}}$$

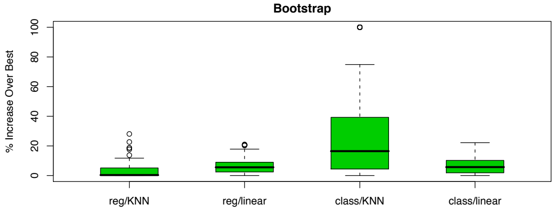
with

$$\hat{w} = \frac{.632}{1 - .368\hat{R}}$$

- $.632 \le \hat{w} \le 1$ as $\hat{R}$ ranges from 0 to 1.
- $\widehat{\text{Err}}^{(.632+)}$ ranges from $\widehat{\text{Err}}^{(.632)}$ to $\widehat{\text{Err}}^{(1)}$.
- $\widehat{\text{Err}}^{(.632+)}$ is a compromise between $\widehat{\text{Err}}^{(.632)}$ and $\overline{\text{err}}$ that depends on the amount of overfitting.
- Derivation of the above eqn is non-trivial.

How bootstrap and cv perform wrt model selection





- Boxplots show $100\frac{\mathrm{Err}(\hat{\alpha}) - \min_{\alpha}\ \mathrm{Err}(\alpha)}{\max_{\alpha}\ \mathrm{Err}(\alpha) - \min_{\alpha}\ \mathrm{Err}(\alpha)}$ where $\hat{\alpha}$ is the best parameter found via the selection method under investigation.
- 100 training sets were used.