The Bootstrap and Cross-Validation

Hadas Barkay

Faculty of Industrial Engineering and Management
Technion - Israel Institute of Technology
Quenouille’s (1949) a-parametric estimator of bias:

Suppose we want to estimate the bias of an estimator
\( \hat{\theta} = \theta(\hat{F}) = \hat{\theta}(X_1, \ldots, X_n) \),

\[
\text{Bias}(\hat{\theta}) \equiv E_F \theta(\hat{F}) - \theta(F)
\]

We can create n sub-samples, deleting one point at a time, and calculate the estimator for each sub-sample:

\[
\hat{\theta}_{-i} = \theta(X_1, \ldots, X_{i-1}, X_{i+1}, \ldots, X_n)
\]

Quenouille’s jackknife estimator of the bias is:

\[
\text{Bias}(\hat{\theta}) = (n-1) \left( \hat{\theta} - \hat{\theta} \right)
\]

where \( \hat{\theta} = \frac{1}{n} \sum_{i=1}^{n} \hat{\theta}_{-i} \).
Example 1: expectation

\[ \theta = E_F X \]
\[ \hat{\theta} = \bar{X} \]
\[ \hat{\theta}_i = \sum_{j \neq i} X_i = \frac{1}{n-1}(n\bar{X} - X_i) \]
\[ \hat{\theta}. = \frac{1}{n} \sum_{i=1}^{n} \hat{\theta}_i = \frac{1}{n-1}(n\bar{X} - \bar{X}) = \bar{X} \]
\[ \text{Bias} = (n-1)(\hat{\theta}. - \hat{\theta}) = 0 \]
Example 2: variance

\[ \theta = E_F \left( X - E_F X \right)^2 \]

\[ \hat{\theta} = \frac{1}{n} \sum_{i=1}^{n} (X_i - \bar{X})^2 \]

\[ \hat{\theta}_i = \frac{1}{n-1} \sum_{j=1}^{n} (X_j - (n\bar{X} - X_i))^2 \]

\[ \hat{\theta}_* = \frac{1}{n(n-1)} \sum_{i=1}^{n} \sum_{j=1}^{n} (X_j - (n\bar{X} - X_i))^2 \]

\[ \hat{\text{Bias}} = \frac{1}{n(n-1)} \sum_{i=1}^{n} (X_i - \bar{X})^2 \]

Yielding the unbiased jackknife estimator for variance:

\[ \tilde{\theta} = \hat{\theta} - \hat{\text{Bias}} = \frac{1}{n-1} \sum_{i=1}^{n} (X_i - \bar{X})^2 \]
Theorem:

If $\hat{\theta}$ is a quadratic functional, i.e.,

\begin{align*}
(1) \quad & \hat{\theta} = \theta(\hat{F}) = \hat{\theta}(X_1, \ldots, X_n) \\
(2) \quad & \hat{\theta} = \mu^{(n)} + \frac{1}{n} \sum_{i=1}^{n} \alpha^{(n)}(X_i) + \frac{1}{n^2} \sum_{1 \leq i < j \leq n} \beta^{(n)}(X_i, X_j)
\end{align*}

then

\[
\text{Bias}(\hat{\theta}) = (n-1)(\hat{\theta} - \bar{\theta})
\]

is unbiased for estimating the true bias.
Tukey’s a-parametric estimator of variance (1958):

Let \( \text{Var}(\hat{\theta}) = E_F \left[ \hat{\theta}(X_1, \ldots, X_n) - E_F \hat{\theta} \right]^2 \)

The jackknife estimator of the variance is:

\[
\widehat{\text{Var}}(\hat{\theta}) = \frac{(n-1)}{n} \sum_{i=1}^{n} (\hat{\theta}_i - \hat{\theta}.)^2
\]

Example: expectation. \( \theta = E_F X \); \( \hat{\theta} = \bar{X} \)

\[
\hat{\theta}_i = \frac{n\hat{\theta} - X_i}{n-1}, \quad \hat{\theta}. = \hat{\theta}, \quad \hat{\theta}_i - \hat{\theta}. = \frac{\bar{X} - X_i}{n-1}
\]

\[
\widehat{\text{Var}}(\hat{\theta}) = \frac{1}{n(n-1)} \sum_{i=1}^{n} (X_i - \bar{X})^2
\]
(Efron, 1979)

Let $R(X, F)$ be a random variable of interest, where $X = (X_1, \ldots, X_n)$ is an i.i.d. sample from a distribution $F$. We wish to estimate some aspect $f$ of $R$’s distribution.

Examples:

• The bias of an estimator $\hat{\theta}$ to $\theta$.
• The variance of the random variable $R$.
• The probability that $R > \tau$ ($\tau$ fixed).
The bootstrap Monte-Carlo algorithm:

1. Fit the nonparametric MLE of $F$,
   
   $\hat{F} : \text{mass } \frac{1}{n} \text{ at } x_i, \quad i = 1, \ldots, n$

2. Draw a “bootstrap sample” of size $n$ with replacement from $\hat{F}$,
   
   $X_1^*, \ldots, X_n^* \sim \text{i.i.d.} \hat{F}$

   and calculate $R^* \left( X^*, \hat{F} \right)$.

3. Independently repeat step 2 a large number $B$ of times, obtaining “bootstrap replications” $R_1^*, \ldots, R_B^*$ and estimate the aspect of choice from the bootstrap sample.
$F$

Original distribution

$(X_1, \ldots, X_n)$

Original sample

$(X^*_1, \ldots, X^*_n)$

Bootstrap samples

$(X^*_B_1, \ldots, X^*_B_n)$
Examples:

Let $\hat{\theta}$ be an estimator to $\theta$.

Take $R(X, F) = \theta(\hat{F}) - \theta(F)$

and $R^* = R(X^*, \hat{F}) = \theta(\hat{F}^*) - \theta(\hat{F}) = \hat{\theta}^* - \hat{\theta}$

where $\hat{F}^*$ is the empirical probability distribution of the bootstrap sample, which puts mass $M_i^*/n$ on $x_i$,

$M_i^*$ being the number of times $x_i$ appears in the bootstrap sample.

The bootstrap estimate of the bias is $\widehat{\text{BIAS}}_{\text{boot}} = E_* R^*$ approximated by

$$\widehat{\text{BIAS}} = \frac{1}{B} \sum_{b=1}^{B} (\hat{\theta}^{*b} - \hat{\theta}) = \hat{\theta}^* - \hat{\theta}$$
Examples:

2. **Bootstrap estimation of variance.**
   
   Let $\hat{\theta}$ be any functional statistic.
   
   Take $R(X, F) = \theta(F)$
   
   and $R^* = \hat{\theta}^*$

   The bootstrap estimate of the variance of $\hat{\theta}$ is
   
   $$\widehat{\text{VAR}}_{\text{boot}} = E_*(R^* - E_*(R^*))$$

   approximated by
   
   $$\widehat{\text{VAR}} = \frac{1}{B-1} \sum_{b=1}^{B} (\hat{\theta}^*_b - \hat{\theta}^*)^2$$
Note that there are two sources of error in bootstrap estimation:

- The error caused by the finite sample size, \( n \).
  
  \( \hat{F}_n \) with respect to \( F \).

- The error caused by the finite number of bootstrap samples, \( B. \left( \hat{f} \text{ with respect to } \hat{f}_{\text{boot}} \right) \).

We wish to minimize the second term (compared to the first term), such that the bootstrap error will be dominated by the inherited error of the finite sample size. Rule of thumb: take \( B = 40 \ n \).
General Bootstrap Scheme:

1. Original Data
2. Data i.i.d?
3. Calculate Residuals
4. Standard Bootstrap Resampling
5. Model OK?
6. Data Dependent Bootstrap Resampling
The bootstrap for regression models

Example: Fitting a circle to 2D data.
An operator is asked to select a small number of points 
\((x_i, y_i), i = 1, \ldots, n\) from the iris outline of a photo of an eye. 
We want to fit a circle to the iris.
The model:

\[
x^2 + y^2 + 2xx_0 + 2yy_0 + x_0^2 + y_0^2 - R = 0
\]

Formulated as a linear regression model:

\[
\begin{bmatrix}
x_1^2 + y_1^2 \\
\vdots \\
x_n^2 + y_n^2
\end{bmatrix}
= \begin{bmatrix}
2x_1 & 2y_1 & 1 \\
\vdots & \vdots & \vdots \\
2x_n & 2y_n & 1
\end{bmatrix}
\begin{bmatrix}
p_1 \\
p_2 \\
p_3
\end{bmatrix}
+ \begin{bmatrix}
\epsilon_1 \\
\vdots \\
\epsilon_n
\end{bmatrix}
\]

Where,

\[
p_1 = x_0, \quad p_2 = y_0, \quad p_3 = x_0^2 + y_0^2 - R
\]
The estimates of $p_1, p_2, p_3$ can be obtained by linear regression:

$$ Y = X \cdot P + E $$

$$ \hat{P} = \left( X^T X \right)^{-1} X^T Y $$

Suppose we want to study the distribution of $\hat{P}$.

We can ask the operator to resample points 100 times, but ‘the results would most definitely be affected by the subsequently decreasing commitment of the operator’.

Alternatively, we could use the bootstrap.

Problem: $(x_i, y_i), i = 1, \ldots, n$ are not i.i.d.!
Solution: resampling the residuals of the fitted model.

1) Estimate the parameters $\hat{P}$ and construct an estimate of the circle $\hat{Y} = X \cdot \hat{P}$.

2) Calculate the residuals $\hat{E} = Y - \hat{Y}$.

*The linear regression model assumes that the residuals are i.i.d.*

3) Create a set of bootstrap residuals $E^*$ by resampling with replacement from $\hat{E}$. (Centering before resampling?)

4) Create a new estimate of the circle by adding the bootstrapped residuals to the original fitted model

$$Y^* = X \cdot \hat{P} + E^*$$

5) Fit a new set of parameters from the bootstrap sample,

$$\hat{P}^* = (X^T X)^{-1} X^T Y^*$$

6) Repeat steps 3-5 B times to obtain a set of estimators $\hat{P}^1, \ldots, \hat{P}^B$ from which the empirical distributions of the studied parameters can be obtained.
The bootstrap for general regression models:

The model: \[ Y_i = g(x_i, \beta), \quad i = 1, \ldots, n \] where \( x_i, \beta \in \mathbb{R}^p \) and the \( \varepsilon_i \) are i.i.d. for some distribution \( F \) on \( \mathbb{R}^1 \).

\( F \) is assumed to be centered at zero in some sense, e.g., \( E_F \varepsilon = 0 \) or \( \Pr_F (\varepsilon < 0) = 0.5 \).

Having observed the data \((y, x)\), we estimate \( \beta \) by minimizing some measure of distance \( D(y, \eta) \) between \( y \) and the vector of predictors \( \eta(\beta) = \left( g(x_1, \beta), \ldots, g(x_n, \beta) \right) \),

\[ \hat{\beta} : \min_{\beta} D(y, \eta(\beta)). \]

The most common choice of \( D \) is \( D(y, \eta) = \sum_{i=1}^{n} (y_i - \eta_i)^2 \), but suppose our model is:

\[ g(x_i, \beta) = e^{x_i \beta} \quad ; \quad F \text{ unknown} \quad ; \quad D(y, \eta) = \sum_{i=1}^{n} |y_i - \eta_i| \]
The bootstrap algorithm:

1) Construct $\hat{F}$ putting mass $\frac{1}{n}$ at each observed residual,

$$\hat{F} : \text{mass } \frac{1}{n} \text{ at } \hat{\varepsilon}_i = y_i - g(x_i, \hat{\beta})$$

2) Draw a bootstrap data set

$$Y^*_i = g(x_i, \hat{\beta}) + \varepsilon^*_i, \quad i = 1, \ldots, n$$

where $\varepsilon^*_i$ are i.i.d from $\hat{F}$, and calculate

$$\hat{\beta}^* : \min_{\beta} D(Y^*, \eta(\beta)).$$

3) Independently repeat step 2 $B$ times, obtaining bootstrap replications $\hat{\beta}^*_1, \ldots, \hat{\beta}^*_B$. 
Suppose we want to fit a model based on data

\[(X, Y) = (X_{i1}, X_{i2}, \ldots, X_{ik}, Y_i)_{i=1, \ldots, n}\]

If the number of observations, n, is large, we usually divide the data into 3 sets: training, validation, test.
Why do we need validation and test sets?

Figure: Typical training set error

If we increase the complexity of the model, we fit a better model to the data set – but we might be *over-fitting*. 

Example: Suppose $k = 1$, $X,Y \in \mathbb{R}$, our loss function is squared error, and the model is linear regression.

• Simple linear regression model (2 parameters):
  \[ Y = \beta_0 + \beta_1 X + \varepsilon \]

• A model that allows for non-linear effects (3 parameters):
  \[ Y = \beta_0 + \beta_1 X + \beta_2 X^2 + \varepsilon \]

• General linear model (d+1 parameters),
  \[ Y = \beta_0 + \sum_{j=1}^{d} \beta_j X^j + \varepsilon \]

If we take the degree of the polynomial to be high enough, we can fit a perfect model (SSE=0)!
• We need the training set to fit the model ($\beta$).

• We need the validation set to select parameters (d).

• We need the test set to estimate the error of the model.

Figure: typical training and test errors

![Diagram showing typical training and test errors](image)
More examples:

• KNN (parameter: k)
• NNet (weight decay parameter)
• MARS (number of basis functions)
• CART (complexity parameter for pruning)

Suppose we don’t have a large sample, and the learning curve of the model requires allocating as many observations as possible to the training set.

**Cross-Validation (CV)** uses the training set both for fitting the model and for selecting parameters.
**Cross Validation:**

- Divide the training set into $r$ disjoint groups, $\{T\} = \{T_1, T_2, \ldots, T_r\}$
- for $i=1$ to $r$, fit the model using $\{T\} \cap \{T_i\}$, and calculate the prediction (or classification) error $Err_i$ using $\{T_i\}$ as the test set.
- The CV error of the model is the mean of the errors.
- Select the parameter (or model) which minimizes the CV error.

$$\hat{Err} = \sum_{i=1}^{r} Err_i$$
How to choose \( r \)?

A tradeoff between the computational effort required for large \( r \) and the bias caused by the learning curve for small \( r \). Usually, either 10-fold CV or 5-fold CV are chosen.

Leave-one-out cross-validation: as above, omitting one observation at a time. Used in KNN.
Cross-Validation: Formalism

Let \((X, Y)\) be the observed data.
Let \(Y\) be the target variable, and \(X\) the corresponding vector of explanatory variables.
Let \(\hat{f}(X)\) be the model that has been fitted by the training sample.
Let \(L(Y, \hat{f}(X))\) be the loss function for prediction or classification.
The *test error*, denoted by $\text{Err}$, is the expected prediction error over an independent test sample:

$$\text{Err} = E\left[ L(Y, \hat{f}(X)) \right]$$

The *training error*, denoted by $\overline{\text{err}}$, is the average loss over the training sample:

$$\overline{\text{err}} = \frac{1}{N_{\text{tr}}} \sum_{i=1}^{N_{\text{tr}}} L(y_i, \hat{f}(x_i))$$
Let $f(X, \alpha) = f_{\alpha}(X)$ be a model with tuning parameter $\alpha$.

**Model selection**: estimating the performance of different models in order to choose the (approximate) best one, i.e.,

$$f(X, \alpha^*) = \arg \min_{\alpha} L(Y, f(X, \alpha))$$

**Model assessment**: having chosen a final model, estimating its prediction error on new data.

Cross-validation aims at model selection, based on the training set.
Cross-validation:

Let \( \kappa : \{1, \ldots, N_{tr} \} \mapsto \{1, \ldots, r \} \) be an indexing function that indicates the partition to which observation \( i \) is randomly allocated.

Denote by \( \hat{f}^{-k}(x, \alpha) \) the fitted model, computed with the \( k \)-th part of the data removed.

The cross-validation estimate of prediction error is

\[
\hat{Err}(\alpha) = CV(\alpha) = \frac{1}{N_{tr}} \sum_{i=1}^{N_{tr}} L(y_i, \hat{f}^{-\kappa(i)}(x_i, \alpha))
\]
We find the tuning parameter $\alpha^*$ that minimizes the CV error**, and choose $f(X, \alpha^*)$ as the best model.

We then fit the chosen model by the entire training set. This model, $\hat{f}(X, \alpha^*)$, is our final model.

The prediction (classification) error of the model is estimated by the mean test set error:

$$\hat{Err} = \frac{1}{N_{te}} \sum_{j=1}^{N_{te}} L(y_j, \hat{f}(x_j, \alpha^*))$$

** 1-SE rule: choose the simplest model whose error is no more than one standard error above the error of the best model.
Generalized Cross-Validation (GCV)

GCV is an approximation to leave-one-out cross-validation, for linear fitting under squared-error loss.

Linear fitting: \( \hat{y} = Sy \)

For many linear fitting methods,

\[
\frac{1}{N} \sum_{i=1}^{N} \left[ y_i - \hat{f}^{-i}(x_i, \alpha) \right]^2 = \frac{1}{N} \sum_{i=1}^{N} \left[ \frac{y_i - \hat{f}(x_i, \alpha)}{1 - S_{ii}} \right]^2
\]

where \( S_{ii} \) is the i-th diagonal element of \( S \).

The GCV approximation is

\[
GCV = \frac{1}{N} \sum_{i=1}^{N} \left[ \frac{y_i - \hat{f}(x_i, \alpha)}{1 - \text{trace}(S_{ii}) / N} \right]^2
\]
Estimating prediction (classification) errors using the bootstrap

Naive bootstrap estimator:

Let $\hat{f}^{*b}(x_i)$ be the predicted value at $x_i$, from the model fitted by the $b$-th bootstrap dataset.

The estimated prediction (classification) error is:

$$\hat{Err}_{boot} = \frac{1}{B} \frac{1}{N_{tr}} \sum_{b=1}^{B} \sum_{i=1}^{N_{tr}} L(y_i, \hat{f}^{*b}(x_i))$$
Problem: the bootstrap datasets are acting as the training set, while the original training set is acting as the test set. Why is this bad? These sets have common observations, and are therefore biased in favor of overfit models.

Example: 1-NN.
Suppose we have two equal-size classes, with the targets (class label = Y) statistically independent of the explanatory variables (X), the loss function is 0-1, and the classification method is 1-NN.
• The true classification error is 0.5.
• The contribution of an observation i to the bootstrap estimate $\hat{Err}_{boot}$ will be:
  
  $0, \quad$ if i appears in the bootstrap sample $b$
  $0.5, \quad$ otherwise
The probability that observation $i$ appears in bootstrap sample $b$ is:

$$\Pr(i \in \text{sample } b) = 1 - \left(1 - \frac{1}{N_{tr}}\right)^{N_{tr}}$$

$$\approx 1 - e^{-1}$$

$$= 0.632$$

Hence the expectation of $\widehat{Err}_{boot}$ is approximately

$$E\left(\widehat{Err}_{boot}\right) \approx 0.5 \times 0.632 = 0.184$$

which is far below the true error rate of 0.5.
Solutions:

1. The *leave-one-out bootstrap* estimate of prediction (classification) error (Efron, 1983):

\[ \hat{Err}^{(1)} = \frac{1}{N_{tr}} \sum_{i=1}^{N_{tr}} \frac{1}{|C^{-i}|} \sum_{b \in C^{-i}} L(y_i, \hat{f}^*_b(x_i)) \]

The estimator mimics the leave-one out cross-validation. For each observation \( i \), we keep only the predictions from bootstrap samples that do *not* contain observation \( i \) (denoted by \( C^{-i} \)).

Pros: solves the overfitting problem.
Cons: might have training-set-size bias, since the number of distinct observations in each bootstrap sample is \( \approx 0.632 \times N_{tr} \).
2. The “.632 estimator” (Efron, 1983) is defined by:

\[
\hat{\text{Err}}^{(.632)} = 0.368 \times \text{err} + 0.632 \times \hat{\text{Err}}^{(1)}
\]

where \( \text{err} \) is the training set error. This estimator is designed to pull the leave-one-out bootstrap estimate down towards the training error rate.

In the two equal random class 1-NN example, the expectation of the .632 estimator is

\[
\begin{align*}
\bar{\text{err}} &= 0 \\
E\left(\hat{\text{Err}}^{(1)}\right) &= 0.5 \\
\Rightarrow E\left(\hat{\text{Err}}^{(.632)}\right) &= 0.328 \times 0 + 0.632 \times 0.5 = 0.316
\end{align*}
\]
3. *The “.632+” estimator* (Efron and Tibshirani, 1997)

First define the *no-information error rate*, $\gamma$, to be the error rate of our prediction (classification) rule if the predicting variables and the response were statistically independent. An estimate of $\gamma$ is obtained by averaging the loss over all the permutations:

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

The *relative overfitting rate* is defined to be

$$\hat{R} = \frac{\hat{\text{Err}}^{(1)} - \text{err}}{\hat{\gamma} - \text{err}}$$
The “.632+ estimator” is defined by

\[
\hat{Err}^{(.632+)} = (1 - \hat{w}) \times err + \hat{w} \times Err^{(1)},
\]

where \( \hat{w} = \frac{0.632}{1 - 0.328 \hat{R}} \).

Extreme cases:
(a) If there is no overfitting, \( \hat{Err}^{(1)} = err \)

\[
\hat{R} = 0
\]

\[
\hat{w} = 0.632
\]

\[
\Rightarrow \hat{Err}^{(.632+)} = \hat{Err}^{(.632)} = \hat{Err}
\]
(b) If the overfitting equals the no-information value,

\[ \hat{Err}^{(1)} = (1 - \hat{w}) \times \text{err} + \hat{w} \times \hat{Err}^{(1)} \]

where \( \hat{w} = \frac{0.632}{1 - 0.328\hat{R}} \).

\begin{align*}
\hat{R} &= 1 \\
\hat{w} &= 1 \\
\Rightarrow \hat{Err}^{(.632+)} &= \hat{Err}^{(1)}
\end{align*}
Comparison of the three bootstrap estimators:

References: