Model Assessment and Selection

Thomas Lonon

Financial Engineering/Financial Analytics
Stevens Institute of Technology

September 27, 2018

Confusion Matrix

A binary classifier can make two types of errors:

- Incorrectly predict the alternative when the null is true
- Incorrectly predict the null when the alternative is true

The confusion matrix displays this information, for example using the LDA to determine whether a person will default or not:

	Did Default	Did Not Default	Total
Classified to Default	79	22	101
Classified Not to Default	254	9645	9899
Total	333	9667	10000

The error rate among those who defaulted is very high

Sensitivity: (True Positive Rate) the percentage of observations that actually satisfy the null hypothesis that are classified as belonging to the null

Specificity: (True Negative Rate) the percentage of observations that actually do not satisfy the null hypothesis that are classified as not belonging to the null.

In the previous table, of the 333 who did default, 79 were predicted to do so $\frac{79}{333} = 23.7\%$ is the sensitivity. Where as of the 9667 who did not default, 9645 were correctly classified so $\frac{9645}{9677} = 99.67\%$ is the specificity.

Type I and II Errors

	Actual Positive	Actual Negative
Classified Positive	True Positive	False Positive
Classified Negative	False Negative	True Negative

- Type I error: incorrect rejection of a true null hypothesis (false negative)
- Type II error: incorrect failure to reject a false null hypothesis (false positive)

	Actual Positive	Actual Negative
Classified Positive	TP	FP
Classified Negative	FN	TN
Total	P	N

- Sensitivity: $TP/P = TPR$
- Specificity: $TN/N = TNR$
- Accuracy: $(TP + TN)/(P + N)$
- Total Error Rate: $(FP + FN)/(P + N)$

ROC

The **Receiver Operating Characteristic:** (ROC) displays the true positive rate against the false positive rate.

The overall performance is given by the area under the curve (AUC)[1]

Let Y be a target variable, X a vector of inputs, $\hat{f}(X)$ a prediction model that has been estimated from a set of training data $\mathcal{T}$.

We need a function to measure the "loss", the errors between Y and $\hat{f}(X)$. Typically we have:

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

The *test error*, also referred to as the *generalization error*, is given by:

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

The expected prediction error (or expected test error) is given by:

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

The *training error* is the average loss over the training sample:

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

[2]

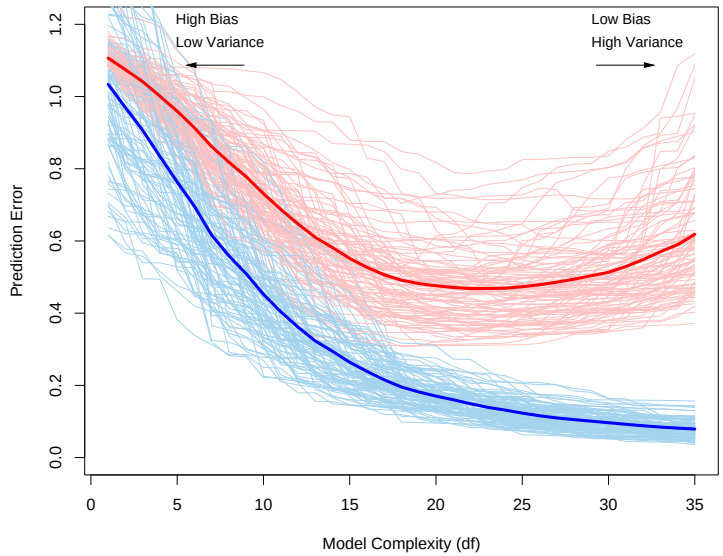

FIGURE 7.1. *Behavior of test sample and training sample error as the model complexity is varied. The light blue curves show the training error* $\overline{\text{err}}$*, while the light red curves show the conditional test error* $\text{Err}_{\mathcal{T}}$ *for* 100 *training sets of size* 50 *each, as the model complexity is increased. The solid curves show the expected test error* Err *and the expected training error* $\text{E}[\overline{\text{err}}]$*.*

Considerations

Typically we have tuning parameter(s) α and so our predictions are given by $\hat{f}_\alpha(X)$. We will often suppress this notation.

There are two separate goals we have in mind:

- **Model Selection:** estimating the performance of different models in order to choose the best one
- **Model Assessment:** having chosen a final model, estimating its prediction error (generalization error) on new data.

Bias-Variance Decomposition

Assume $Y = f(X) + \varepsilon$ where $\mathbb{E}[\varepsilon] = 0$ and $\mathbb{V}(\varepsilon) = \sigma_\varepsilon^2$. We have:

$$\begin{aligned}
\text{Err}(x_0) &= \mathbb{E}[(Y - \hat{f}(x_0))^2 | X = x_0] \\
&= \sigma_\varepsilon^2 + (\mathbb{E}[\hat{f}(x_0)] - f(x_0))^2 + \mathbb{E}[\hat{f}(x_0) - \mathbb{E}[\hat{f}(x_0)]]^2 \\
&= \sigma_\varepsilon^2 + \text{Bias}^2(\hat{f}(x_0)) + \text{Var}(\hat{f}(x_0)) \\
&= \text{Irreducible Error} + \text{Bias}^2 + \text{Variance}
\end{aligned}$$

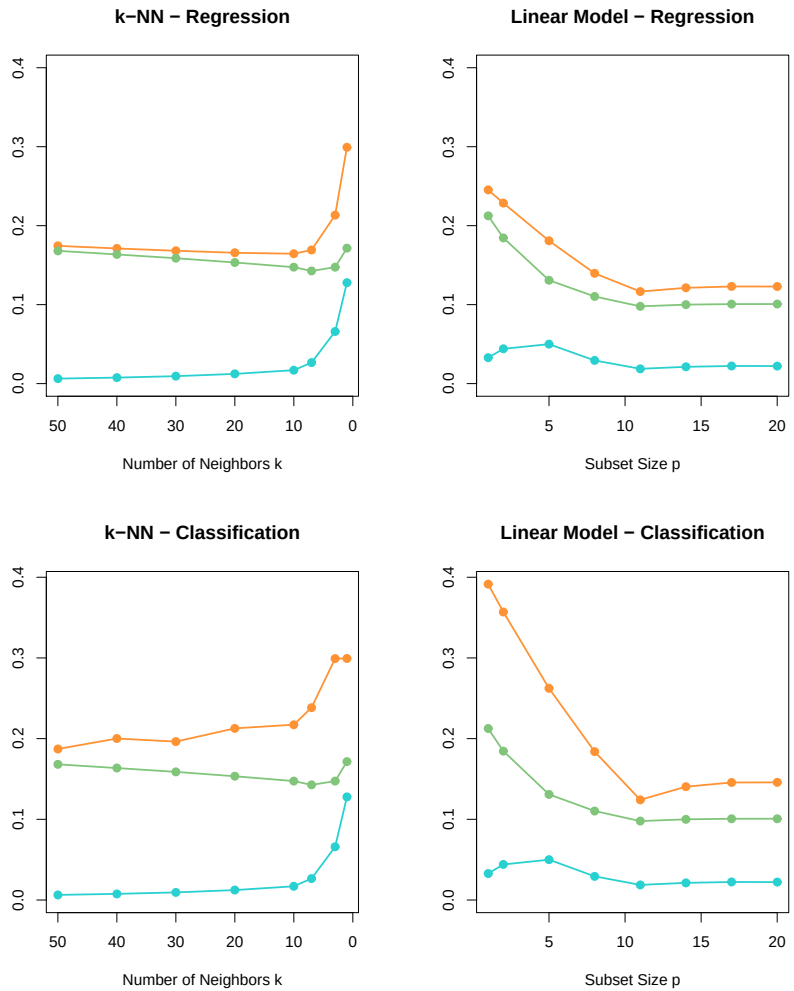

FIGURE 7.3. *Expected prediction error (orange), squared bias (green) and variance (blue) for a simulated example. The top row is regression with squared error loss; the bottom row is classification with* 0–1 *loss. The models are k-nearest neighbors (left) and best subset regression of size p (right). The variance and bias*

Validation Set Approach

Validation Set Approach is the division of data into two subsets:

1. **training set:** set to estimate the parameters in some model of interest
2. **validation set:** set to assess whether the model with these parameters fits adequately

Directly estimates the expected extra-sample error

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

Validation and Cross-Validation

We compute the validation set error (or the cross-validation set error) for each model $\mathcal{M}_k$ and select the k for which the test error is smallest.

Another advantage is this doesn't require an estimate for the error variance.

K-Fold Cross-Validation

Divide the set of observations randomly into k groups (or folds) of approximately equal size, use each group as the validation set to get an estimate for the test error.

For $\kappa : \{1, \ldots, N\} \mapsto \{1, \ldots, K\}$, an indexing function and $\hat{f}^{-k}(x)$ the fitting of the function with the k^{th} data set removed we have the cross validation error given by:

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

In practice, $k = 5, 10$.

LOOCV

Leave One Out Cross Validation: (LOOCV) is where $K = N$. In this case we have $\kappa(i) = i$ and for the i^{th} observation, we fit using all the data points except that observation.

- Has far less bias
- Every time you perform the LOOCV, you get the same result (non-random)

For a set of models $f(x, \alpha)$ indexed by a tuning parameter α, we can define $\hat{f}^{-k}(x, \alpha)$ as the α^{th} model fit with the k^{th} part of the data removed.

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

This function $CV(\hat{f}, \alpha)$ gives us an estimate of the test error curve, and we seek the value of α that minimizes it. For that value of α we then fit to the entire data.

Bias-Variance Trade-Off

Bias: The K-fold CV bias lies between that of the VS approach and the LOOCV. So if we only wanted to reduce the bias, we would choose the LOOCV.

Variance: LOOCV has higher variance than does the K-fold CV with $K < n$

One performs K-fold cross-validation using $K = 5$ or $K = 10$ because these values have been shown to give test error rate estimates that suffer from neither excessively high bias nor from high variance.

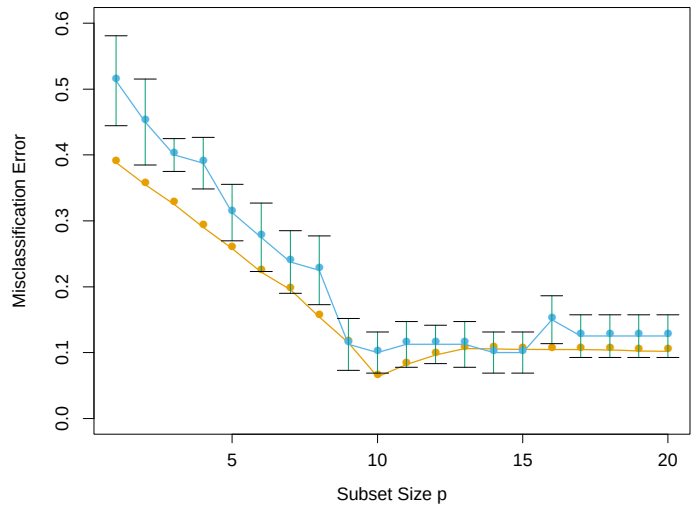

FIGURE 7.9. *Prediction error (orange) and tenfold cross-validation curve (blue) estimated from a single training set, from the scenario in the bottom right panel of Figure 7.3.*

Generalized Cross-Validation

For a linear fitting method, we can write: $\hat{\mathbf{y}} = \mathbf{S}\mathbf{y}$
For many linear fitting methods we have:

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

where S_{ii} is the i^{th} diagonal element of $\mathbf{S}$.

The GCV approximation is:

$$\text{GCV}(\hat{f}) = \frac{1}{N}\sum_{i=1}^{N}\left[\frac{y_i - \hat{f}(x_i)}{1 - \text{trace}(\mathbf{S})/N}\right]^2$$

A correct implementation of K-fold CV is as follows:

1. Divide the samples into K cross-validation folds at random
2. For each fold $k = 1, \ldots, K$
   - 2.1 Find a subset of "good" predictors that show fairly strong correlation with the class labels, using all of the samples except those in fold k
   - 2.2 Using just this subset of predictors, build a multivariate classifier, using all of the samples except those in fold k
   - 2.3 Use the classifier to predict the class labels for the samples in fold k

[2]

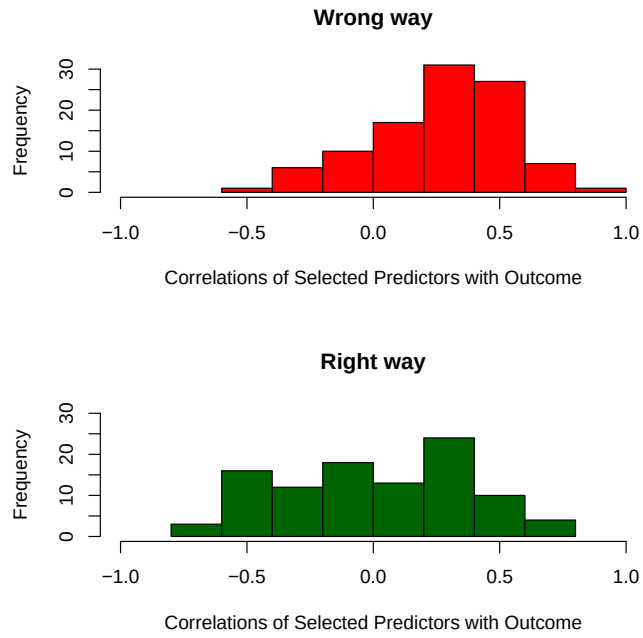

FIGURE 7.10. *Cross-validation the wrong and right way: histograms shows the correlation of class labels, in* 10 *randomly chosen samples, with the* 100 *predictors chosen using the incorrect (upper red) and correct (lower green) versions of cross-validation.*

Bootstrap

Suppose we have a model fit to a set of training data, denoted $\mathbf{Z} = (z_1, z_2, \ldots, z_N)$ where $z_i = (x_i, y_i)$.

1. Perform B samplings from this data set (with replacement) of sample size N, producing B bootstrap data sets.
2. Refit the model to each of these bootstrapped data sets.
3. Examine the behavior of the fits

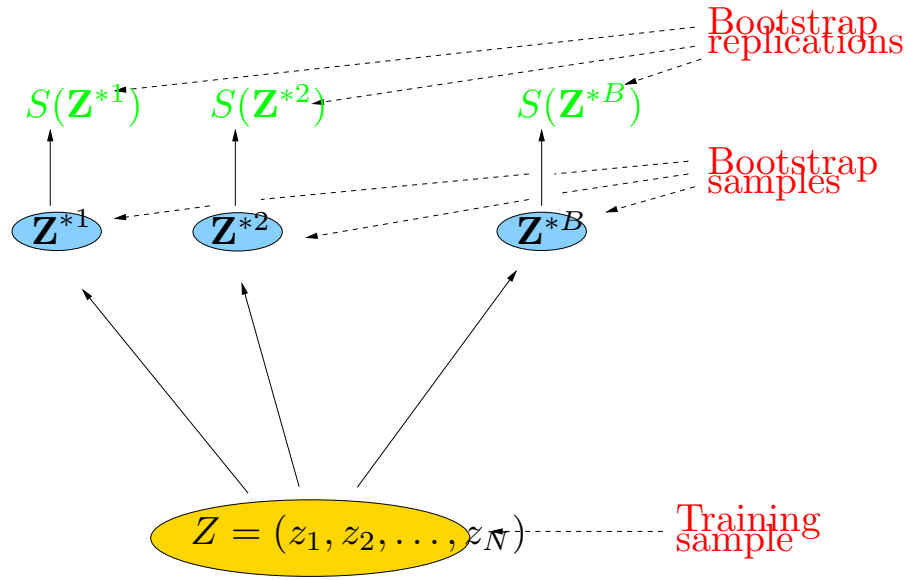

FIGURE 7.12. *Schematic of the bootstrap process. We wish to assess the statistical accuracy of a quantity $S(\mathbf{Z})$ computed from our dataset. B training sets $\mathbf{Z}^{*b}$, $b = 1, \ldots, B$ each of size N are drawn with replacement from the original dataset. The quantity of interest $S(\mathbf{Z})$ is computed from each bootstrap training set, and the values $S(\mathbf{Z}^{*1}), \ldots, S(\mathbf{Z}^{*B})$ are used to assess the statistical accuracy of $S(\mathbf{Z})$.*

In this figure, we let $S(\mathbf{Z})$ be any quantity computed from the data. We can then use the bootstrap sampling to estimate the distribution of $S(\mathbf{Z})$ and get its variance given by:

$$\mathbb{V}(\widehat{S(\mathbf{Z})}) = \frac{1}{B-1}\sum_{b=1}^{B}(S(\mathbf{Z}^{*b}) - \bar{S}^{*})^2$$

where

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

The bootstrap typically estimates well only the expected prediction error Err.

One method to estimate the prediction error would be to keep track of how well it predicts the training set. If $\hat{f}^{*b}(x_i)$ is the predicted value at x_i from the b^{th} bootstrap, then our estimate is given by:

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

This approach uses overlapping data sets and so is not the most reliable.

A better bootstrap approach can be found by mimicking cross-validation. An example of Leave-One-Out bootstrap estimation would be:

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

where C^{-i} is the set of indices of the bootstrap samples b that do *not* contain observation i.

We either need B to be large enough that each of these are nonempty, or to ignore those that are empty.

[1] Trevor Hastie Gareth James, Daniela Witten and Robert Tibshirani. *An Introduction to Statistical Learning with Applications in R*. Number v. 6. Springer, 2013.

[2] Robert Tibshirani Trevor Hastie and Jerome Friedman. *The Elements of Stastical Learning: Data Mining, Inference, and Prediction*. Number v.2 in Springer Series in Statistics. Springer, 2009.