Machine Learning
Cross Validation

Irving Gómez Méndez

August-December, 2021
Introduction

Remember that the ridge estimator is given by

$$\hat{\beta}_R = (X^T X + \lambda I)^{-1} X^T Y,$$

and that there is an optimum value of $\lambda$, say $\lambda^*$ that minimizes the MSE of the estimator. This parameter can be tuned by cross-validation (CV).

In a more general framework, it is common to have two separate goals 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 on new data.
Since a different value of the parameter $\lambda$ corresponds to a different ridge parameter and in turn into a different model, finding the optimum value of $\lambda$ can be understood as a model selection problem.

If we are in a data-rich situation, the best approach for both problems is to randomly divide the dataset into three parts: a training set, a validation set, and a testing set.
► The **training set** is used to fit the models.
► The **validation test** is used to estimate the prediction error for model selection.
► the **testing set** is used to estimate the generalization error of the final chosen model.

A typical split might be 50% for training, 25% for validation and testing.

When there is insufficient data to split it into three parts, we can use cross-validation to approximate the validation step.
Cross-Validation
**K-Fold Cross-Validation**

*K*-fold cross-validation uses part of the available data to fit the model, and a different part to test it. We split the data into *K* roughly equal-sized parts; for example, when *K* = 5:

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Train</td>
<td>Train</td>
<td>Validation</td>
<td>Train</td>
<td>Train</td>
</tr>
</tbody>
</table>

For the *k*-th part, we fit the model to the other *K* − 1 parts of the data, and calculate the prediction error of the fitted model when predicting the *k*-th part of the data. We do this for *k* = 1, 2, ..., *K* and combine the *K* estimates of the error.
Let \( \kappa : \{1, \ldots, N\} \to \{1, \ldots, K\} \) be an indexing function that indicates the partition to which observation \( i \) is allocated by the randomization. Denote by \( \hat{f}^{-k} \) the fitted function, computed with the \( k \)-th part of the data removed. Then the cross-validation estimate of the prediction error is

\[
CV(\hat{f}) = \frac{1}{N} \sum_{i=1}^{N} L\left(y_i, \hat{f}^{-\kappa(i)}(x_i)\right).
\]

Typical choices of \( K \) are 5 or 10. The case \( K = N \) is known as leave-one-out cross-validation (LOO-CV). In this case \( \kappa(i) = i \), and for the \( i \)-th observation the fit is computed using all the data except the \( i \)-th
Tuning Parameters using Cross-Validation

Given a set of models $f(x, \alpha)$ indexed by a tuning parameter $\alpha$, denote by $\hat{f}^{-k}(x, \alpha)$ the $\alpha$-th model fit with the $k$-th part of the data removed. Then for this set of models we define

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

The function $CV(\hat{f}, \alpha)$ provides an estimator of the test error curve (as a function of $\alpha$) and we find the tuning parameter that minimizes it. Our final chosen model is $f(x, \hat{\alpha})$ which we then fit to all the data.

As an estimator of the prediction error, cross-validation would be biased upward.
Tuning the Ridge Parameter

Let be

\[ \hat{Y}(\lambda) = X\hat{\beta}_R(\lambda) \]
\[ = X(X^TX + \lambda I)^{-1}X^TY \]

and

\[ L(Y, \hat{Y}(\lambda)) = (Y - \hat{Y}(\lambda))^T(Y - \hat{Y}(\lambda)) \]
\[ = \sum_{i=1}^{n} (y_i - \hat{y}_i(\lambda))^2 \]
\[ \equiv L(\lambda) \]