Project Notes

Tanner Fiez

1 Ordinary Cross Validation (Leave one out cross-validation)

In leave-one-out cross validation we can compute the error very efficiently for linear models. For these models we can write them as $f = X\beta$, where X is a $n \times p$ design matrix and β is a $p \times 1$ parameter vector. Consider a minimization problem of a sum of squares plus a quadratic penalty term with a known matrix D

$$\sum_{i=1}^{n}(y_i - x_i\beta)^2 + \lambda\beta^T D\beta. \tag{1}$$

Fitting the model to $n-1$ points contained in y_{-i} gives the following minimization problem

$$\min \sum_{j=1, j\neq 1}^{n} [y_j - f_{-i}(x_j)]^2 + \lambda\beta^T D\beta \tag{2}$$

which can equivalently be written as

$$\min \sum_{j=1}^{n} [y_j^\star - f_{-i}(x_j)]^2 + \lambda\beta^T D\beta \tag{3}$$

where

$$y_j^\star = \begin{cases} y_j & \text{if } j \neq i \\ y_i - y_i + f_{-i}(x_i) & \text{if } j = i \end{cases} \tag{4}$$

Now solving the minimization problem by taking the derivative with respect to β and setting it equal to zero.

$$\begin{aligned}
\frac{\partial}{\partial\beta}\sum_{j=1}^{n}[y_j^\star - x_j\beta]^2 + \lambda\beta^T D\beta &= -2\sum_{j=1}^{n}(y_j^\star - x_j) + 2\lambda D\beta \\
\sum_{j=1}^{n}(-2x_j^T(y_j^\star - x_j\beta)) + 2\lambda D\beta &= 0 \\
\sum_{j=1}^{n}(-x_j^T y_j^\star + x_j^T x_j\beta) + \lambda D\beta &= 0 \\
\sum_{j=1}^{n} -x_j^T y_j^\star + \sum_{j=1}^{n} x_j^T x_j\beta + \lambda D\beta &= 0 \\
\sum_{j=1}^{n} x_j^T x_j\beta + \lambda D\beta &= \sum_{j=1}^{n} x_j^T y_j^\star \\
(\sum_{j=1}^{n} x_j^T x_j + \lambda D)\beta &= \sum_{j=1}^{n} x_j^T y_j^\star \\
(X^T X + \lambda D)\beta &= X^T Y^\star \\
\beta &= (X^T X + \lambda D)^{-1} X^T Y^\star
\end{aligned} \tag{5}$$

Hence, the estimator is given by $\hat{f} = X(X^TX + \lambda D)^{-1}X^TY^\star$ which can be written as a linear smoother $\hat{f} = S^{(\lambda)}Y^\star$, where $S^{(\lambda)} = X(X^TX + \lambda D)^{-1}X^T$. Then we have the leave one out cross validation estimate $\hat{f}^{(\lambda)}_{-1}(x_i) = S^{(\lambda)}_i Y^\star$. Now we can move terms around

$$\begin{aligned}
\hat{f}^{(\lambda)}_{-1}(x_i) &= S^{(\lambda)}_i Y^\star \\
\hat{f}^{(\lambda)}_{-1}(x_i) &= S^{(\lambda)}_i Y - S^{(\lambda)}_{ii} y_i + S^{(\lambda)}_{ii} \hat{f}^{(\lambda)}_{-i}(x_i) \\
\hat{f}^{(\lambda)}_{-1}(x_i) &= \hat{f}^{(\lambda)}(x_i) - S^{(\lambda)}_{ii} y_i + S^{(\lambda)}_{ii} \hat{f}^{(\lambda)}_{-1}(x_i) \\
\hat{f}^{(\lambda)}_{-1}(x_i) - S^{(\lambda)}_{ii} \hat{f}^{(\lambda)}_{-1}(x_i) &= \hat{f}^{(\lambda)}(x_i) - S^{(\lambda)}_{ii} y_i \\
\hat{f}^{(\lambda)}_{-1}(x_i) &= \frac{\hat{f}^{(\lambda)}(x_i) - S^{(\lambda)}_{ii} y_i}{1 - S^{(\lambda)}_{ii}}
\end{aligned} \tag{6}$$

Then the error in the estimate is

$$y_i - \hat{f}^{(\lambda)}_{-i}(x_i) = \frac{y_i(1 - S^{(\lambda)}_{ii}) - \hat{f}^{(\lambda)}(x_i) + S^{(\lambda)}_{ii} y_i}{1 - S^{(\lambda)}_{ii}} = \frac{y_i - \hat{f}^{(\lambda)}(x_i)}{1 - S^{(\lambda)}_{ii}}. \tag{7}$$

In LOOCV the complete error is

$$\frac{1}{n} \sum_{i=1}^{n} [y_i - \hat{f}^{(\lambda)}_{-1}(x_i)]^2 \tag{8}$$

and we can plug in the result that was derived for $y_i - \hat{f}^{(\lambda)}_{-i}(x_i)$ to get

$$\frac{1}{n} \sum_{i=1}^{n} \left(\frac{y_i - \hat{f}^{(\lambda)}(x_i)}{1 - S^{(\lambda)}_{ii}}\right)^2. \tag{9}$$

This result allows us to efficiently compute the LOOCV error since we can use the smoothing matrix to avoid the need to fit n independent models. Instead we simply need to fit one model to get the error.

2 Forward-stepwise selection

The forward-stepwise selection algorithm is a greedy algorithm for subset selection of features. The advantages of it are the computational efficiency of it, the result will have lower variance. The algorithm works by first starting with the intercept term, and then sequentially adds to the model the feature that improves the fit.