Regression

Context

- Supervised learning
- Output variable $Y$ is quantitative
- Collection of $p$ predictor variables $X = (X_1, \ldots, X_p)$
- Assume function $Y = f(X) + \epsilon$, $f$ is unknown, but sought for
  - prediction: May be used to estimate $Y$ at new points $X = x$
  - inference: Process may illuminate importance of some predictors, lack thereof of others
  - inference: If model not too complex, may reveal contribution of various individual predictors to the response

Estimating $f$

Take the true (ideal) $f$ to be the following: at each $x$,

$$f(x) = E(Y|X = x).$$

Equation (1) gives theoretical formula for $f$, but cannot calculate it unless (miraculously) one has actual conditional probability model $f_{Y|X}(y|X = x)$. Thus, we must estimate $f$. For any estimate $g(x)$,

$$E((Y - g(x))^2|X = x) = \frac{[f(x) - g(x)]^2}{\text{reducible error}} + \frac{\text{Var}(\epsilon)}{\text{irreducible error}}$$

This quantity is minimized when $g = f$, with $\text{Var}(\epsilon)$ there as a threshold value we cannot hope to get below.

Some estimation “approaches”:

- For every new input $x$, estimate $E(Y|X = x)$ as the mean of data values at $X = x$
  - may not be any points for which $X = x$ in dataset
  - If any points exist for which $X = x$, still likely too few to make accurate estimate of $E(Y|X = x)$
- Nearest-Neighbor Local Averaging
  Idea: Let sliding neighborhood dictate points $(x, y)$ to employ in average
  - Works well for large-ish $N$ (many subjects) and $p \leq 4$ (small number of predictors)
  - For $p > 4$ increasingly subject to the **curse of dimensionality** as nearest neighbors tend to be far away
• Other methods (later in book): linear regression, thin-plate splines, subset selection, least squares, bagging, boosting, SVMs

Assessing Model Accuracy

For a collection of labeled data
\[(x_1, y_1), \ldots, (x_M, y_M),\]
define the average squared prediction error
\[
\text{MSE} = \frac{1}{M} \sum_{i=1}^{M} (y_i - \hat{f}(x_i))^2.
\]

Having picked a model type, one uses labeled data, a collection called the training set, to specify parameters of that model and obtain \(\hat{f}\). As flexibility in the model type increases, generally speaking MSE\(_{\text{Tr}}\) decreases; this reflects the fact that one can generally reduce the average prediction error on the training data by increasing the flexibility of the model. However, if we hold apart labeled data in a test set (labeled data which is not used to train parameters of the model), it holds generally that MSE\(_{\text{Te}}\) decreases to a point, then begins to increase. This minimum MSE\(_{\text{Te}}\) can be thought of as the break between underfitting and overfitting.

Some Trade-offs in Model Selection

• prediction accuracy vs. interpretability
  linear models are easy to interpret but low in flexibility; many other models are more difficult to interpret
• want to get fit just right, not overfit nor underfit
• may prefer simpler model (fewer vars.) over black-box predictor based on full set of predictors
• bias vs. variance. One can imagine many different sets of training data with each, for a given estimation method, leading to a different estimate \(\hat{f}\) to \(f\). If we randomly select a test data point \((x_0, y_0)\), we might look (again) at the expected value of squared prediction error, with an eye toward the fact that there are now two sources of variability:
  1. the random selection of a test point \((x_0, y_0)\), and
  2. the random selection of the training set (felt as variability in the predicted response \(\hat{f}(x_0)\)).
Earlier, we broke \( E((Y - \hat{f}(X))^2) \) into two components, one labeled reducible and the other irreducible error. It can be shown that the reducible part breaks down further, as

\[
E((y_0 - \hat{f}(x_0))^2) = \text{Var}(\hat{f}(x_0)) + [E(\hat{f}(x_0)) - f(x_0)]^2 + \text{Var}(\epsilon)
\]

An increase in flexibility generally corresponds to increases in \( \text{Var}(\hat{f}(x_0)) \), a positive component within the (positive) reducible error. Correspondingly, the squared bias term, which measures how close we expect our predicted value \( \hat{f}(x_0) \) is to the ideal one \( f(x_0) \), decreases.

The Classification Problem

We suppose there are \( J \) different values, or classes, to which our observations belong. For convenience, when an observation belongs to class \( j, j = 1, 2, \ldots, J, \) we will write \( Y = j \). In an ideal world, we might know, at each instantiation \( X = x \) of the predictor variables, the conditional probabilities

\[
P(Y = 1|X = x), \ P(Y = 2|X = x), \ldots, \ P(Y = J|X = x).
\] (2)

The Bayes classifier, which is the gold standard, producing the lowest possible error rate (the fewest proportion of observations assigned to the wrong class), uses these conditional probabilities, assigning each unlabeled instance \((x, ?)\) to the class \( y = j \) for which

\[
P(Y = j|X = x)
\]

is the largest.

The problem with the Bayes classifier is that the conditional probabilities (2) are unknown, except in the case of simulated data, concocted from such probabilities. So, a natural thing to do is to estimate them from training data, and use the estimates to classify unlabeled data in precisely the same way we would use the true conditional probabilities. That is what the \( K \)-nearest neighbors (KNN) classifier does.

KNN classification

Suppose \( X = x \), an observation which is unlabeled/unclassified. For a fixed integer value \( K \geq 1 \), we let \( N_0 \) be a ball centered at \( x \) just large enough to contain \( K \) observations \((x_1, y_1), \ldots, (x_K, y_K)\) from the training set. We take as our approximate conditional probability

\[
P(Y = j|X = x) = \text{the proportion of training data points in } N_0 \text{ from class } j,
\]

then assign \((x, ?)\) to the class \( j \) for which this proportion is largest. This method is most responsive to changes in training data when \( K = 1 \), with flexibility dropping as \( K \) grows.