Notes on Statistical and Machine Learning

Projection Pursuit Regression

Chapter: *19* **Prepared by:** *Chenxi Zhou*

This note is prepared based on *Chapter 11, Neural Networks* in Hastie, Tibshirani, and Friedman (2009).

I. Projection Pursuit Regression

1. **Setup:** We assume that the input vector $\mathbf{x}$ has p components and the target variable is $y \in \mathbb{R}$. Let $\boldsymbol{\omega}_1, \boldsymbol{\omega}_2, \cdots, \boldsymbol{\omega}_M$ be p-dimensional unit vectors of unknown parameters.

2. **Model Specification:** The projection pursuit regression (PPR) model has the form

$$f(\mathbf{x}) = \sum_{m=1}^{M} g_m(\boldsymbol{\omega}_m^\top \mathbf{x}). \tag{1}$$

   (a) This is an additive model in the derived features $v_m = \boldsymbol{\omega}_m^\top \mathbf{x}$. Here, the functions g_m are *unspecified* and are estimated along with the unknown directions $\boldsymbol{\omega}_m$;

   (b) The function $\mathbf{x} \mapsto g_m(\boldsymbol{\omega}^\top \mathbf{x})$ is called a *ridge function* in $\mathbb{R}^p$ and only varies in the direction defined by the vector ω_m.

   (c) The scalars $v_m = \boldsymbol{\omega}_m^\top \mathbf{x}$ is the projection of $\mathbf{x}$ onto the unit vector $\boldsymbol{\omega}_m$.

3. **Comments on PPR Model:**

   (a) If M is taken arbitrarily large, for appropriate choices of g_m, the PPR model (1) can approximate any continuous function in $\mathbb{R}^p$ arbitrarily well. Such a class of models is called a *universal approximator*.

   (b) *Interpretation* of such a universal approximator is usually difficult. The PPR model is most useful for *prediction*, and *not* very useful for producing an understandable model for the data.

   An exception is when $M = 1$, which is called *single index model* in econometrics.

4. **Fitting the PPR Model:** To fit PPR model, given the training data $\{(\mathbf{x}_i, y_i)\}_{i=1}^n$, we minimize the following criterion

$$\sum_{i=1}^{n} \left[y_i - \sum_{m=1}^{M} g_m(\boldsymbol{\omega}_m^\top \mathbf{x}_i) \right]^2 \tag{2}$$

   over functions g_m and direction vectors $\boldsymbol{\omega}_m$ for all $m = 1, \cdots, M$.

   We consider the case when $M = 1$.

- Given the direction vector $\boldsymbol{\omega}$, we have the derived variables $v_i = \boldsymbol{\omega}^\top \mathbf{x}_i$. Then, the resulting problem is a one-dimensional smoothing problem and we can apply any smoother (e.g., smoothing spline) to obtain an estimate of g;
- Given the function g, we wish to minimize with respect to the direction vector $\boldsymbol{\omega}$. We can use the *Gauss-Newton algorithm*, which is a quasi-Newton method where the Hessian matrix part involving the second-order derivative of g is discarded. Letting $\boldsymbol{\omega}^{(\text{old})}$ be the current estimate of $\boldsymbol{\omega}$, by Taylor's expansion, we have

$$g(\boldsymbol{\omega}^\top \mathbf{x}_i) \approx g(\boldsymbol{\omega}^{(\text{old})^\top} \mathbf{x}_i) + g'(\boldsymbol{\omega}^{(\text{old})^\top} \mathbf{x}_i) \cdot (\boldsymbol{\omega} - \boldsymbol{\omega}^{(\text{old})^\top} \mathbf{x}_i).$$

Plugging into the objective function (2), we have

$$\begin{aligned}
&\sum_{i=1}^n \big(y_i - g(\boldsymbol{\omega}^\top \mathbf{x}_i)\big)^2 \\
\approx &\sum_{i=1}^n \Big(y_i - g(\boldsymbol{\omega}^{(\text{old})^\top} \mathbf{x}_i) - g'(\boldsymbol{\omega}^{(\text{old})^\top} \mathbf{x}_i) \cdot (\boldsymbol{\omega} - \boldsymbol{\omega}^{(\text{old})})^\top \mathbf{x}_i\Big)^2 \\
= &\sum_{i=1}^n \big(g'(\boldsymbol{\omega}^{(\text{old})^\top} \mathbf{x}_i)\big)^2 \left[\frac{y_i - g(\boldsymbol{\omega}^{(\text{old})^\top} \mathbf{x}_i)}{g'(\boldsymbol{\omega}^{(\text{old})^\top} \mathbf{x}_i)} - (\boldsymbol{\omega} - \boldsymbol{\omega}^{(\text{old})})^\top \mathbf{x}_i\right]^2 \\
= &\sum_{i=1}^n \big(g'(\boldsymbol{\omega}^{(\text{old})^\top} \mathbf{x}_i)\big)^2 \left[\left(\boldsymbol{\omega}^{(\text{old})^\top} \mathbf{x}_i + \frac{y_i - g(\boldsymbol{\omega}^{(\text{old})^\top} \mathbf{x}_i)}{g'(\boldsymbol{\omega}^{(\text{old})^\top} \mathbf{x}_i)}\right) - \boldsymbol{\omega}^\top \mathbf{x}_i\right]^2.
\end{aligned}$$

To minimize the right-hand side, one can use the weighted least squares regression with the target

$$\boldsymbol{\omega}^{(\text{old})^\top} \mathbf{x}_i + \frac{y_i - g(\boldsymbol{\omega}^{(\text{old})^\top} \mathbf{x}_i)}{g'(\boldsymbol{\omega}^{(\text{old})^\top} \mathbf{x}_i)}$$

on the input $\mathbf{x}_i$, weights $(g'(\boldsymbol{\omega}^{(\text{old})^\top} \mathbf{x}_i))^2$ and no intercept. Then, we can obtain the the updated coefficient vector $\boldsymbol{\omega}^{(\text{new})}$.
- In this view, the estimation of the unknown function g and parameter $\boldsymbol{\omega}$ has two steps. These two steps are iterated until convergence.

Remark. If $M > 1$, the model can be built in a *forward stage-wise* manner, adding a pair $(\boldsymbol{\omega}_m, g_m)$ at each stage.

5. Implementation Details:

(a) It is more convenient to use local regression and smoothing splines to estimate g;

(b) The number of terms M is usually estimated as part of the forward stage-wise strategy. The model building stops when the next term does *not* appreciably improve the fit of the model. Cross-validation can also be used to determine M.

References

Hastie, Trevor, Robert Tibshirani, and Jerome Friedman (2009). *The Elements of Statistical Learning*. Vol. 1. Springer Series in Statistics. New York, NY, USA: Springer New York Inc.