Notes on Statistical and Machine Learning

An Overview of Supervised Learning

Chapter: *4* **Prepared by:** *Chenxi Zhou*

This note is produced based on *Chapter 2, Overview of Supervised Learning* in Hastie, Tibshirani, and Friedman (2009).

I. Introduction

1. **Regression and Classification Problems:**
   (a) *Similarities:* Both regression and classification problems
      i. attempt to use the inputs to predict the outputs, and
      ii. can be viewed as function approximation tasks.
   (b) *Differences:*
      i. In *regression* problems, the output variables are quantitative, meaning that some measurements are bigger than others and measurements close in value are close in nature;
      ii. In the *classification* problems, the output variable are qualitative or categorical variables, meaning that the values belong to a finite set. There is no explicit ordering in these values.

II. Linear Regression and Least Squares

1. **Least Squares for Prediction:** Given a vector of inputs $(X_1, X_2, \cdots, X_{p-1})^\top \in \mathbb{R}^{p-1}$, we predict the output $Y \in \mathbb{R}$ via the model

$$\widehat{Y} = \hat{\beta}_0 + \sum_{i=1}^{p-1} X_j \hat{\beta}_j, \tag{1}$$

   where the term $\hat{\beta}_0$ is the *intercept* in statistics or the *bias* in machine learning.
   Usually, we let $X := (1, X_1, X_2, \cdots, X_{p-1})^\top \in \mathbb{R}^p$ and $\widehat{\boldsymbol{\beta}} := (\hat{\beta}_0, \hat{\beta}_1, \cdots, \hat{\beta}_{p-1})^\top$, then (1) can be written succinctly as

$$\widehat{Y} = X^\top \widehat{\boldsymbol{\beta}}. \tag{2}$$

   Remark. Here, we assume the output Y is a scalar; in general, Y can be a multi-dimensional vector.

2. **Geometric Interpretation of Linear Model:** Notice that in the $(p+1)$-dimensional input-output space, $(X^\top, \widehat{Y})^\top \in \mathbb{R}^{p+1}$ represents a *hyperplane*:

(a) if the constant term is included in X, the hyperplane includes the origin and is a subspace;

(b) if the constant term is *not* included in X, it is an affine space cutting the Y-axis at the point $(0, \hat{\beta}_0)$.

3. **Parameter Estimation of $\boldsymbol{\beta}$:** To estimate the value of $\boldsymbol{\beta}$ based on independent and identically distributed (i.i.d) samples, denoted by $\{(\mathbf{x}_i, y_i)\}_{i=1}^n$, we use the *least squares* method and minimize the *residual sum of squares*

$$\mathrm{RSS}(\boldsymbol{\beta}) := \sum_{i=1}^{n} (y_i - \mathbf{x}_i^\top \boldsymbol{\beta})^2 = (\mathbf{Y} - \mathbf{X}\boldsymbol{\beta})^\top (\mathbf{Y} - \mathbf{X}\boldsymbol{\beta}), \tag{3}$$

where $\mathbf{X} \in \mathbb{R}^{n \times p}$ is a matrix with each row being an input vector, $\mathbf{x}_i \in \mathbb{R}^p$ is the i-row of $\mathbf{X}$, and $\mathbf{Y} := (y_1, \cdots, y_n)^\top \in \mathbb{R}^n$ is a vector of the response variables.

Differentiating (3) with respect to $\boldsymbol{\beta}$ and setting the resulting equation to $\mathbf{0}_p$, the p-dimensional vector with all entries being 0, yield the *normal equation*

$$\mathbf{X}^\top (\mathbf{Y} - \mathbf{X}\boldsymbol{\beta}) = \mathbf{0}_p. \tag{4}$$

If $\mathbf{X}^\top \mathbf{X}$ is nonsingular, then the *unique* solution to the normal equation is given by

$$\widehat{\boldsymbol{\beta}} = (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{Y}, \tag{5}$$

and the *fitted value* at the i-th input $\mathbf{x}_i$ is $\widehat{Y}_i = \mathbf{x}_i^\top \widehat{\boldsymbol{\beta}}$.

At an arbitrary input $\mathbf{x}_0$, the prediction is $\widehat{Y}_0 = \mathbf{x}_0^\top \widehat{\boldsymbol{\beta}}$.

Remark. $\mathrm{RSS}(\boldsymbol{\beta})$ is a quadratic function of $\boldsymbol{\beta}$ and, hence, the minimum always exists but may *not* be unique.

III. k-Nearest Neighbor Method for Prediction

1. **Idea of k-Nearest Neighbor Method:** Use observations in the training set $\mathcal{T}$ closest in input space to $\mathbf{x}$ to form $\widehat{Y}$. Mathematically, the k-nearest neighbor fit for $\widehat{Y}$ is

$$\widehat{Y}(\mathbf{x}) = \frac{1}{k} \sum_{x_i \in \mathcal{N}_k(\mathbf{x})} y_i, \tag{6}$$

where $\mathcal{N}_k(\mathbf{x})$ is the neighborhood of $\mathbf{x}$ defined by the k closest points $\mathbf{x}_i$ in $\mathcal{T}$ to $\mathbf{x}$. That is, we find the k observations $\mathbf{x}_i$ closest to $\mathbf{x}$ in the input space and average their responses.

Remark. One needs a measure of *closeness* when applying this k-nearest neighbor method. One choice is the Euclidean distance.

2. **Relationship between Error and k:** For k-nearest neighbor method, the error on the *training data* is approximately an *increasing* function of k.

Remark. This relationship does *not* necessarily hold for the test data.

3. **Effective Number of Parameters:** The *effective* number of parameters of the k-nearest neighbor method is n/k, and decreases with increasing k. This is because, if the neighborhoods are non-overlapping, there would be n/k neighborhoods and we fit one parameter in each neighborhood.

4. **A Comparison between Least Squares and k-Nearest Neighbor Methods:**

   - *Least Squares Method:*
     (a) The decision boundary is linear, smooth and stable to fit;
     (b) It relies on the stringent assumption that a linear decision boundary is appropriate;
     (c) It has low variance and potentially high bias.
   - *k-Nearest Neighbor Method:*
     (a) It does *not* rely on any stringent assumption and can adapt to any situation;
     (b) The decision boundary can be wiggle and unstable;
     (c) It has low bias but may have high variance.

IV. Statistical Decision Theory

1. **General Setup:** Let
   (a) $X \in \mathcal{X} \subseteq \mathbb{R}^p$ be a real-valued random *input* vector,
   (b) $Y \in \mathcal{Y} \subseteq \mathbb{R}$ be real-valued random *output* variable,
   (c) $\mathbb{P} : \mathcal{X} \times \mathcal{Y} \to [0, 1]$ be the joint distribution function of X and Y,
   (d) $f : \mathcal{X} \to \mathcal{Y}$ be a function, and
   (e) $L(Y, f(X))$ be a loss function for penalizing errors between Y and $f(X)$.

2. **Squared Error Loss Function:** The *squared error loss function* is

$$L(Y, f(X)) := (Y - f(X))^2. \tag{7}$$

   We search for a function f^* that minimizes the *expected squared prediction error* (EPE)

$$\mathrm{EPE}(f) := \mathbb{E}\big[(Y - f(X))^2\big] = \int_{\mathcal{X}\times\mathcal{Y}} (y - f(\mathbf{x}))^2 \, \mathbb{P}(\mathrm{d}\mathbf{x}, \mathrm{d}y); \tag{8}$$

   that is,

$$f^* := \arg\min_f \mathrm{EPE}(f).$$

   Since we can write the EPE as

$$\mathrm{EPE}(f) = \mathbb{E}\big[(Y - f(X))^2\big] = \mathbb{E}_X\big[\mathbb{E}_{Y \,|\, X=\mathbf{x}}[(Y - f(\mathbf{x}))^2 \,|\, X = \mathbf{x}]\big],$$

to obtain f^*, it is sufficient to minimize the inner expectation $\mathbb{E}_{Y\,|\,X=\mathbf{x}}[(Y-f(\mathbf{x}))^2\,|\,X=\mathbf{x}]$, and the pointwise minimizer is

$$f^*(\mathbf{x}) = \arg\min_{c\in\mathbb{R}} \mathbb{E}_{Y|X}\big[(Y-c)^2\,|\,X=\mathbf{x}\big] = \mathbb{E}[Y\,|\,X=\mathbf{x}]; \tag{9}$$

that is, the best prediction of Y at any point $X=\mathbf{x}$ under the squared error loss function is the *conditional expectation*, also known as *regression function*.

3. **Prediction Error at an Arbitrary Test Point:** Suppose that the relationship between Y and X is linear up to a random term,

$$Y = X^\top\boldsymbol{\beta} + \varepsilon, \qquad \text{where } \varepsilon \sim \text{Normal}(0,\sigma^2). \tag{10}$$

We estimate $\boldsymbol{\beta}$ using the least squares method with data $\{(\mathbf{x}_i, y_i)\}_{i=1}^n \overset{\text{i.i.d}}{\sim} \mathbb{P}$. We calculate the prediction error at an arbitrary but *fixed* point $\mathbf{x}_0$.

Note that the fitted value at $\mathbf{x}_0$ under the linear model (10) is

$$\widehat{Y}_0 = \mathbf{x}_0^\top\widehat{\boldsymbol{\beta}} = \mathbf{x}_0^\top(\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top\mathbf{Y}.$$

We first show that $\widehat{Y}_0 = \mathbf{x}_0^\top\boldsymbol{\beta} + \sum_{i=1}^n \ell_i(\mathbf{x}_0)\varepsilon_i$, where $\ell_i(\mathbf{x}_0)$ is the i-th element of $\mathbf{X}(\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{x}_0$. Since

$$\widehat{\boldsymbol{\beta}} = (\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top\mathbf{Y} = (\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top(\mathbf{X}\boldsymbol{\beta}+\boldsymbol{\varepsilon}) = \boldsymbol{\beta} + (\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top\boldsymbol{\varepsilon},$$

where $\boldsymbol{\varepsilon} := (\varepsilon_1, \cdots, \varepsilon_n)^\top \in \mathbb{R}^n$, we have

$$\widehat{Y}_0 = \mathbf{x}_0^\top\widehat{\boldsymbol{\beta}} = \mathbf{x}_0^\top\boldsymbol{\beta} + \mathbf{x}_0^\top(\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top\boldsymbol{\varepsilon} = \mathbf{x}_0^\top\boldsymbol{\beta} + \sum_{i=1}^n \ell_i(\mathbf{x}_0)\varepsilon_i.$$

Then, we show the *prediction error* at $\mathbf{x}_0$ is

$$\text{EPE}(\mathbf{x}_0) := \mathbb{E}_{Y_0|X=\mathbf{x}_0}\Big[\mathbb{E}_{\mathcal{T}}\big[(Y_0-\widehat{Y}_0)^2\big]\Big] = \sigma^2 + \sigma^2\,\mathbb{E}_{\mathcal{T}}[\mathbf{x}_0^\top(\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{x}_0],$$

where $\mathcal{T}$ denotes the set of training data. We first note that

$$\begin{aligned}
(Y_0-\widehat{Y}_0)^2 &= \big(Y_0 - \mathbb{E}_{Y_0|X=\mathbf{x}_0}[Y_0] + \mathbb{E}_{Y_0|X=\mathbf{x}_0}[Y_0] - \mathbb{E}_{\mathcal{T}}[\widehat{Y}_0] + \mathbb{E}_{\mathcal{T}}[\widehat{Y}_0] - \widehat{Y}_0\big)^2 \\
&= \big(Y_0 - \mathbb{E}_{Y_0|X=\mathbf{x}_0}[Y_0]\big)^2 + \big(\mathbb{E}_{Y_0|X=\mathbf{x}_0}[Y_0] - \mathbb{E}_{\mathcal{T}}[\widehat{Y}_0]\big)^2 + \big(\mathbb{E}_{\mathcal{T}}[\widehat{Y}_0] - \widehat{Y}_0\big)^2 \\
&\quad + 2\big(Y_0 - \mathbb{E}_{Y_0|X=\mathbf{x}_0}[Y_0]\big)\big(\mathbb{E}_{Y_0|X=\mathbf{x}_0}[Y_0] - \mathbb{E}_{\mathcal{T}}[\widehat{Y}_0]\big) \\
&\quad + 2\big(Y_0 - \mathbb{E}_{Y_0|X=\mathbf{x}_0}[Y_0]\big)\big(\mathbb{E}_{\mathcal{T}}[\widehat{Y}_0] - \widehat{Y}_0\big) \\
&\quad + 2\big(\mathbb{E}_{Y_0|X=\mathbf{x}_0}[Y_0] - \mathbb{E}_{\mathcal{T}}[\widehat{Y}_0]\big)\big(\mathbb{E}_{\mathcal{T}}[\widehat{Y}_0] - \widehat{Y}_0\big).
\end{aligned}$$

Then, notice

$$Y_0 = \mathbf{x}_0^\top\boldsymbol{\beta} + \varepsilon, \qquad \mathbb{E}_{Y_0|X=\mathbf{x}_0}[Y_0] = \mathbf{x}_0^\top\boldsymbol{\beta}, \qquad \mathbb{E}_{\mathcal{T}}[\widehat{Y}_0] = \mathbf{x}_0^\top\boldsymbol{\beta}. \tag{11}$$

It follows that all cross terms have zero expectations, and we obtain

$$\begin{aligned}\mathrm{EPE}(\mathbf{x}_0) &= \mathbb{E}_{Y_0|X=\mathbf{x}_0}\Big[\mathbb{E}_{\mathcal{T}}\big[(Y_0 - \mathbb{E}_{Y_0|X=\mathbf{x}_0}[Y_0])^2\big]\Big] + \mathbb{E}_{Y_0|X=\mathbf{x}_0}\Big[\mathbb{E}_{\mathcal{T}}\big[(\mathbb{E}_{Y_0|X=\mathbf{x}_0}[Y_0] - \mathbb{E}_{\mathcal{T}}[\widehat{Y}_0])^2\big]\Big] \\ &\quad + \mathbb{E}_{Y_0|X=\mathbf{x}_0}\Big[\mathbb{E}_{\mathcal{T}}\big[(\mathbb{E}_{\mathcal{T}}[\widehat{Y}_0] - \widehat{Y}_0)^2\big]\Big] \\ &= \mathbb{E}_{Y_0|X=\mathbf{x}_0}[\varepsilon^2] + \big(\mathbb{E}_{Y_0|X=\mathbf{x}_0}[Y_0] - \mathbb{E}_{\mathcal{T}}[\widehat{Y}_0]\big)^2 + \mathbb{E}_{\mathcal{T}}\big[(\mathbb{E}_{\mathcal{T}}[\widehat{Y}_0] - \widehat{Y}_0)^2\big] \\ &= \sigma^2 + \mathrm{Bias}^2(\widehat{Y}_0) + \mathrm{Var}_{\mathcal{T}}[\widehat{Y}_0].\end{aligned}$$

By (11), we have

$$\mathrm{Bias}^2(\widehat{Y}_0) = \big(\mathbb{E}_{Y_0|X=\mathbf{x}_0}[Y_0] - \mathbb{E}_{\mathcal{T}}[\widehat{Y}_0]\big)^2 = 0.$$

In addition,

$$\begin{aligned}\mathrm{Var}_{\mathcal{T}}[\widehat{Y}_0] &= \mathbb{E}_{\mathcal{T}}\big[(\mathbb{E}_{\mathcal{T}}[\widehat{Y}_0] - \widehat{Y}_0)^2\big] \\ &= \mathbb{E}_{\mathcal{T}}\big[(\mathbf{x}_0^\top(\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top\boldsymbol{\varepsilon})^2\big] \\ &= \sigma^2\,\mathbb{E}_{\mathcal{T}}[\mathbf{x}_0^\top(\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{x}_0].\end{aligned}$$

Combining all pieces above, we have

$$\mathrm{EPE}(\mathbf{x}_0) = \sigma^2 + 0 + \sigma^2\,\mathbb{E}_{\mathcal{T}}[\mathbf{x}_0^\top(\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{x}_0].$$

4. **Prediction Function in k-Nearest Neighbor Method:** By (6), the k-nearest neighbor method predicts at an arbitrary point $\mathbf{x}$ by taking the average of k y_i's closest to $\mathbf{x}$ in the training set, that is,

$$\hat{f}_{k\mathrm{NN}}(\mathbf{x}) = \mathrm{Ave}\big(y_i \,|\, \mathbf{x}_i \in \mathcal{N}_k(\mathbf{x})\big), \tag{12}$$

where "Ave" denotes the *average operator*, and $\mathcal{N}_k(\mathbf{x})$ is the neighborhood containing the k points in the training set closest to $\mathbf{x}$.

Note that there are two approximations going on:

- expectation is approximated by averaging over sample data;
- conditioning at a point is relaxed to conditioning on some region "close" to the target point.

For large training sample size n, as k gets large, the average will get more stable.

Fact: Under mild regularity conditions on $\mathbb{P}$, as $n, k \to \infty$ such that $k/n \to 0$, $\hat{f}_{k\mathrm{NN}}(\mathbf{x}) \to \mathbb{E}[Y \,|\, X = \mathbf{x}]$.

5. **Comparing the Least Squares Method and k-Nearest Neighbor:**

   (a) *Similarities:* Both least squares and k-nearest neighbor approximate conditional expectations by averages.

   (b) *Differences:*

- *Least squares*
  - assumes the regression function f is approximately linear in its argument and is well approximated by a globally linear function, i.e., $f(\mathbf{x}) \approx \mathbf{x}^\top \boldsymbol{\beta}$, and
  - is a model-based approach;
- *k-nearest neighbor method* assumes f is well approximated by a locally constant function.

6. **Using L_1 Loss Function:** If we use the L_1 loss function

$$L_1(Y, f(X)) := |Y - f(X)|,$$

the solution in this case is the conditional median

$$\hat{f}(\mathbf{x}) := \arg\min_f \mathbb{E}\big[L_1(Y, f(X))\big] = \text{Median}(Y \mid X = \mathbf{x}).$$

The estimate from the L_1 loss function is more *robust* than those for the conditional mean.

7. **Loss Function for Categorical Variable:** Let G be a categorical variable taking values in $\mathcal{W} := \{1, \cdots, W\}$, the set of possible classes. The loss function can be represented by a $W \times W$ matrix $\mathbf{L}$. The matrix $\mathbf{L}$ is zero on the diagonal and is nonnegative everywhere else. The (w, ℓ)-entry of $\mathbf{L}$, denoted by $L(w, \ell)$, is the price paid for classifying an observation belonging to Class w to Class ℓ.

The expected prediction error in this setting is

$$\text{EPE}(\widehat{G}) := \mathbb{E}\big[L(G, \widehat{G}(X))\big],$$

where $\widehat{G} : \mathcal{X} \to \mathcal{W}$ and $\widehat{G}(X)$ is an estimate of the class that X belongs to. We can write

$$\text{EPE}(\widehat{G}) = \mathbb{E}_X \left[\sum_{w=1}^{W} L(w, \widehat{G}(\mathbf{x}))\, \mathbb{P}(G = w \mid X = \mathbf{x}) \right].$$

It is sufficient to minimize EPE pointwise, i.e.,

$$\widehat{G}(\mathbf{x}) := \arg\min_{g \in \mathcal{W}} \left\{ \sum_{w=1}^{W} L(w, g)\, \mathbb{P}(G = w \,|\, X = \mathbf{x}) \right\}.$$

8. **The Case of 0-1 Loss Function:** If all non-diagonal entries of the matrix $\mathbf{L}$ is 1, we obtain the *0-1 loss function*, where all misclassifications are charged a single unit. With this 0-1 loss function, we have

$$\begin{aligned} \widehat{G}(\mathbf{x}) &= \arg\min_{g \in \mathcal{W}} \left\{ \sum_{w=1}^{W} \mathbb{1}(w \neq g)\, \mathbb{P}(G = w \,|\, X = \mathbf{x}) \right\} \\ &= \arg\min_{g \in \mathcal{W}} \Big\{ 1 - \mathbb{P}(G = g \,|\, X = \mathbf{x}) \Big\} \\ &= \arg\max_{g \in \mathcal{W}} \Big\{ \mathbb{P}(G = g \,|\, X = \mathbf{x}) \Big\}. \end{aligned} \tag{13}$$

The classifier (13) is known as the *Bayes classifier*. It says we classify an observation $\mathbf{x}$ to the most probable class using the conditional distribution $\mathbb{P}(G \mid X = \mathbf{x})$. The error rate of the Bayes classifier is called *Bayes rate*.

V. Statistical Models, Supervised Learning, and Function Approximation

1. **Example of Statistical Models:** Assume the data arose from the statistical model

$$Y = f(X) + \varepsilon, \tag{14}$$

where the random error ε satisfies $\mathbb{E}[\varepsilon] = 0$ and is independent of X. If we use the least squares loss, then

$$f^* := \arg\min_f \Big\{ \mathbb{E}\big[(Y - f(X))^2\big] \Big\}$$

is

$$f^*(\mathbf{x}) = \mathbb{E}[Y \mid X = \mathbf{x}], \qquad \text{for all } \mathbf{x} \in \mathcal{X}.$$

2. **Supervised Learning:** In supervised learning, given the training samples $\{(\mathbf{x}_i, y_i)\}_{i=1}^n$, the input values $\mathbf{x}_i$'s are fed into an artificial system, called a *learning algorithm*, to produce outputs $\hat{f}(\mathbf{x}_i)$. The learning algorithm can modify the relationship $\hat{f}$ in response to differences $y_i - \hat{f}(\mathbf{x}_i)$ between the original and generated outputs. This process is known as *learning by examples*.

   Upon the completion of the learning process, we hope that the artificial outputs $\hat{f}(\mathbf{x}_i)$ will be close enough to the real ones y_i.

3. **Perspective From Function Approximation:** Data pairs $\mathcal{T} := \{(\mathbf{x}_i, y_i)\}_{i=1}^n$ can be viewed as points in a $(p+1)$-dimensional Euclidean space, where we assume $\mathbf{x}_i \in \mathcal{X} \subseteq \mathbb{R}^p$ for all $i = 1, 2, \cdots, n$. The function f has the domain $\mathcal{X}$, and is related to the data via a model such as $y_i = f(\mathbf{x}_i) + \varepsilon_i$. The *goal* is to obtain a useful approximation to f for all $\mathbf{x} \in \mathcal{X}$, given the representations in $\mathcal{T}$.

4. **From Function Approximation to Parameter Estimation:** Many approximations assume a certain functional form of f and have associated a set of parameters θ that are *unknown*. With the presence of training data, we only need to estimate θ.

   Examples include:

   - *Linear Models:* Assume $f_\theta(\mathbf{x}) = \mathbf{x}^\top \theta$;
   - *Linear Basis Expansions:* Assume

$$f_\theta(\mathbf{x}) = \sum_{k=1}^K \theta_k h_k(\mathbf{x}),$$

   where h_k's are a suitable set of functions or transformations of the input vector $\mathbf{x}$, and $\theta := (\theta_1, \theta_2, \cdots, \theta_K)^\top \in \mathbb{R}^K$.

5. **Parameter Estimation (I) — Least Squares Method:** We estimate θ by minimizing the residual sum-of-squares

$$\mathrm{RSS}(\theta) := \sum_{i=1}^{n} \big(y_i - f_\theta(\mathbf{x}_i)\big)^2. \tag{15}$$

6. **Parameter Estimation (II) — Principle of Maximum Likelihood:** Suppose we have a random sample y_i, $i = 1, \cdots, n$, from a density $\mathbb{P}_\theta(y)$ indexed by θ. The log-probability of the observed sample is

$$L(\theta) := \sum_{i=1}^{n} \log \mathbb{P}_\theta(y_i) \tag{16}$$

   The principle of maximum likelihood assumes that the most reasonable values for θ are those for which the probability of the observed sample is the largest.

7. **Equivalence of Least Squares Method and the Principle of Maximum Likelihood:** Suppose the output random variable Y and the input (fixed) variable X are related by the additive error model $Y = f_\theta(X) + \varepsilon$, with $\varepsilon \sim \mathrm{Normal}(0, \sigma^2)$. Here, we assume σ^2 is known. Under this setting, Y is a random variable following $\mathrm{Normal}(f_\theta(X), \sigma^2)$ distribution.

   We estimate θ using the principle of maximum likelihood. The log-likelihood function is

$$\begin{aligned} L(\theta) &= -\frac{n}{2}\log(2\pi) - n\log\sigma - \frac{1}{2\sigma^2}\sum_{i=1}^{n}\big(y_i - f_\theta(\mathbf{x}_i)\big)^2 \\ &= -\frac{n}{2}\log(2\pi) - n\log\sigma - \frac{1}{2\sigma^2}\mathrm{RSS}(\theta). \end{aligned}$$

   Therefore, in this case, maximizing the log-likelihood function L is equivalent to the least squares method.

8. **Principle of Maximum Likelihood for Categorical Output Variable:** Assume that the output variable $G \in \mathcal{W} := \{1, \cdots, W\}$ is categorical, and that the conditional probability for the w-th category is

$$\mathbb{P}(G = w \,|\, X = \mathbf{x}) = p_{w,\theta}(\mathbf{x}), \qquad \text{for all } w = 1, \cdots, W,$$

   where $p_{w,\theta}$ is indexed by some parameter θ. Then, with data $\{(\mathbf{x}_i, y_i)\}_{i=1}^n$, the log-likelihood function is

$$L(\theta) = \sum_{i=1}^{n} \log p_{y_i,\theta}(\mathbf{x}_i).$$

   We can estimate θ by maximizing L.

VI. Structured Regression Models

1. **Problem:** Consider the problem of minimizing the RSS criterion

$$\mathrm{RSS}(f) = \sum_{i=1}^{n} \big(y_i - f(\mathbf{x}_i)\big)^2 \tag{17}$$

over the class of all functions. Any function $\hat{f}$ passing through all training points $\{(\mathbf{x}_i, y_i)\}_{i=1}^n$ is a solution and there are infinitely many solutions. In order to obtain useful results for a finite sample size n, we must restrict the eligible solutions to (17) to a smaller set of functions.

2. **Imposing Constraints:** The constraints imposed by most learning methods can be described as *complexity restrictions*, meaning some kind of *regular behavior* in small neighborhoods of the input space.

   The *strength* of the constraints is dictated by the *neighborhood size*. The larger the size of the neighborhood, the stronger the constraint, and the more sensitive the solution is to the particular choice of constraint.

VII. Classes of Restricted Estimators

1. **Roughness Penalty:** Rather than minimizing the RSS alone, we minimize the RSS plus a roughness penalty, called the penalized RSS,

$$\mathrm{PRSS}_\lambda(f) = \mathrm{RSS}(f) + \lambda J(f), \tag{18}$$

   where J is the penalty functional and is large for functions f that vary too rapidly over small regions of input space.

   Minimizing PRSS_λ is a trade-off of two conflicting goals:

   (a) requiring the function f to have a goodness of fit to data, measured by $\mathrm{RSS}(f)$, and

   (b) requiring it is *not* too complex, measured by $J(f)$.

   The value of λ controls the strength of this trade-off.

   Remark. Penalty functionals J can be constructed for functions in any dimension, and special versions can be created to impose special structure.

2. **Example of Roughness Penalty — Cubic Smoothing Spline:** The cubic smoothing spline for one-dimensional input is the solution to

$$\mathrm{PRSS}_\lambda(f) = \sum_{i=1}^{n} \big(y_i - f(x_i)\big)^2 + \lambda \int [f''(x)]^2 \mathrm{d}x. \tag{19}$$

   The roughness penalty functional J here controls large values of the second derivative of f, i.e., the curvature of f. The *amount* of penalty is dictated by $\lambda \geq 0$; more specifically,

- for $\lambda = 0$, no penalty is imposed, and any interpolating function is a solution; and
- for $\lambda = \infty$, only functions linear in x are permitted.

3. **Bayesian Interpretation of Roughness Penalty:** Penalty functionals J express our *prior* belief that the type of functions we seek exhibit a certain type of *smooth* behavior.

   The penalty functional J corresponds to a log-prior, and $\text{PRSS}_\lambda(f)$ corresponds to the log-posterior distribution, and minimizing PRSS_λ amounts to finding the *posterior mode*.

4. **Kernel Methods and Local Regression:** Local regression explicitly specifies the nature of the local neighborhood and the class of regular functions fitted locally.

   The nature of the local neighborhood is specified through a *kernel function* $K_\lambda : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$, and $K_\lambda(\mathbf{x}_0, \mathbf{x})$ can be regarded as the weight to the point $\mathbf{x}$ in a region around $\mathbf{x}_0$. The parameter λ controls the width of the neighborhood. One example of the kernel function is the *Gaussian kernel*

$$K_\lambda(\mathbf{x}_0, \mathbf{x}) = \exp\left(\frac{\|\mathbf{x}_0 - \mathbf{x}\|_2^2}{2\lambda}\right).$$

   - *Example of Kernel Estimator — Nadaraya-Watson Weighted Average:*

$$\hat{f}(x_0) = \frac{\sum_{i=1}^n K_\lambda(\mathbf{x}_0, \mathbf{x}_i) y_i}{\sum_{i=1}^n K_\lambda(\mathbf{x}_0, \mathbf{x}_i)}. \tag{20}$$

   - *Local Regression Estimate:* Define the local regression estimate of $f(\mathbf{x}_0)$ as $f_{\hat{\theta}}(\mathbf{x}_0)$, where

$$\hat{\theta} := \arg\min_\theta \sum_{i=1}^n K_\lambda(\mathbf{x}_0, \mathbf{x}_i)\big(y_i - f_\theta(\mathbf{x}_i)\big)^2, \tag{21}$$

     and f_θ is some parametrized function.

   - *Nearest-Neighbor Method:* Nearest-neighbor methods can be thought of as kernel methods having a more data-dependent metric. The kernel function for the k-nearest neighbor method is

$$K_k(\mathbf{x}, \mathbf{x}_0) = \mathbb{1}\big(\{\mathbf{x} \,|\, \|\mathbf{x} - \mathbf{x}_0\| \le \|\mathbf{x}_{(k)} - \mathbf{x}_0\|\}\big), \tag{22}$$

     where $\mathbf{x}_{(k)}$ is the training observation ranked the k-th in distance from $\mathbf{x}_0$, $\mathbb{1}(S)$ is the indicator of the set S, and $\|\cdot\|$ is a metric we choose.

5. **Basis Functions and Dictionary Methods:** Suppose f is a linear expansion of basis functions

$$f_\theta(x) = \sum_{m=1}^M \theta_m h_m(\mathbf{x}). \tag{23}$$

   Examples include

- linear regression and polynomial regression;
- smoothing spline regression; and
- single-layer feed-forward neural network model with linear output weights.

VIII. Model Selection and the Bias-Variance Tradeoff

1. **Example – k-Nearest Neighbor Method:** Suppose the data arise from a model $Y = f(X) + \varepsilon$, with $\mathbb{E}[\varepsilon] = 0$ and $\text{Var}[\varepsilon] = \sigma^2$. Assume the values of $\mathbf{x}_i$'s are fixed. The expected prediction error at $\mathbf{x}_0$, which is not present in the data and is fixed, using the k-nearest neighbor method is

$$\begin{aligned}\text{EPE}_k(\mathbf{x}_0) &= \mathbb{E}\big[(Y - \hat{f}_k(\mathbf{x}_0))^2 \,|\, X = \mathbf{x}_0\big] \\ &= \sigma^2 + \text{Bias}^2[\hat{f}_k(\mathbf{x}_0)] + \text{Var}_{\mathcal{T}}[\hat{f}_k(\mathbf{x}_0)] \\ &= \sigma^2 + \left[f(\mathbf{x}_0) - \frac{1}{k}\sum_{\ell=1}^{k} f(\mathbf{x}_{(\ell)})\right]^2 + \frac{\sigma^2}{k}, \end{aligned} \tag{24}$$

where the subscript "$\mathbf{x}_{(\ell)}$" indicates the ℓ-th nearest neighbor to $\mathbf{x}_0$. Then, note the three terms in (24):

- The first term $\sigma^2 = \mathbb{E}[(Y - f(\mathbf{x}_0))^2]$ is the variance of the new test target and is irreducible;
- The second term is the squared bias, where

$$\text{Bias}[\hat{f}_k(\mathbf{x}_0)] := \mathbb{E}_{\mathcal{T}}[\hat{f}_k(\mathbf{x}_0)] - f(\mathbf{x}_0) \tag{25}$$

  and the expectation averages the randomness in the training data. This term is likely to increase with k;
- The variance term $\text{Var}_{\mathcal{T}}[\hat{f}_k(\mathbf{x}_0)]$ is the variance of an average and decreases with k.

The sum of the second term (the squared bias) and the third term (the variance) is the *mean squared error* of $\hat{f}_k(\mathbf{x}_0)$ in estimating $f(\mathbf{x}_0)$ and is under our control. As k varies, there is a bias-variance tradeoff between these two terms.

2. **Bias-Variance Tradeoff vs. Model Complexity:**
   - As the model complexity is increased, the variance tends to increase and the squared bias tends to decrease; and
   - as the model complexity is decreased, the variance tends to decrease and the squared bias tends to increase.

   Typically, we choose the model complexity to trade bias off with variance in such a way as to minimize the test error.

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.