Intro. to Supervised Learning

Zed

March 5, 2017

1 Statistical Decision Theory

1.1 Quantitative Dependent Variable

We want to firstly develop a general framework for supervised learning. We first consider quantitative output (label) $Y \in \mathbb{R}$ as a random variable. And $X \in \mathbb{R}^p$ as a p-random column vector for input variables (features).
We place ourselves in probability space $(\mathbb{R}^p \times \mathbb{R}, \mathcal{F}, \mathbb{P})$. The pair $(X, Y) \in \mathbb{R}^p \times \mathbb{R}$ has joint distribution $p_{X,Y}(x, y)$, and we will also use similar notations for marginal and conditional distributions. Our goal is to find a function $\hat{Y} = f(X)$ to predict the value of Y corresponding to given input. We proceed as follows.

Def. **Loss Function**: $L(Y, \hat{Y})$ is constructed for penalizing errors in prediction. By far we choose a simple *squared error loss*:

$$L(Y, f(X)) = \|Y - f(X)\|_{\mathcal{L}^2} = (Y - f(X))^2$$

Def. **Expected Prediction Error (EPE)**: We seek to find a function f that minimizes the expecion of L over the probability space we defined, which is:

$$EPE(f) := \mathbb{E}\left[L(Y, \hat{Y})\right]$$

If we use the squared error loss,

$$\begin{aligned} EPE(f) &= \mathbb{E}\left[(Y - f(X))^2\right] \\ &= \iint_{\mathbb{R}^p \times \mathbb{R}} (y - f(x))^2 p_{X,Y}(x, y) dy dx \end{aligned} \tag{1}$$

We can split the joint distribution in (1) to conditional distribution times the marginal, and rewrite the integral as

$$\begin{aligned} EPE(f) &= \iint_{\mathbb{R}^p \times \mathbb{R}} (y - f(x))^2 p_{X,Y}(x, y) dy dx \\ &= \int_X p_X(x) \left(\int_Y (y - f(x))^2 p_{Y|X}(y|x) dy \right) dx \\ &= \mathbb{E}_X\left[\mathbb{E}_{Y|X}\left[(Y - f(X))^2 | X = x\right]\right] \end{aligned} \tag{2}$$

And minimizing this expectation suffices to minimizing pointwise for any x:

$$f(x) = \operatorname*{argmin}_{\xi} \mathbb{E}_{Y|X}\left[(Y - \xi)^2 | X = x\right] \tag{3}$$

Take FOC:

$$\begin{aligned} \frac{\partial}{\partial \xi}\mathbb{E}_{Y|X}\left[(Y-\xi)^2|X=x\right] &= \mathbb{E}_{Y|X}\left[\frac{\partial}{\partial \xi}(Y-\xi)^2|X=x\right]=0 \\ \Rightarrow f(x)=\xi &= \mathbb{E}_{Y|X}\left[Y|X=x\right] \end{aligned} \tag{4}$$

So the solution to pointwise minimization problem of EPE is $f(x) = \mathbb{E}_{Y|X}\left[Y|X=x\right]$. Thus the best prediction of Y at point x is the the conditional expectation of Y, when ths 'best' is measured by square error. This is referred to as *regression function.*

Ex. The **k-Nearest Neighbour** methods attempt to implement this recepie directly, with

$$\hat{f}_{knn}(x) = \frac{1}{k}\sum_{x_j \in N_k(x)} y_j$$

Where $N_k(x) = \{x_1, ..., x_k; \|x - x_j\| \leq \|x - z\| \; j = 1, 2, ..., k, \forall z \notin N_k(x)\}$. It uses averaging to approximate expectation, and conditioning on 1 point is relaxed to conditioning on $N_k(x)$. It can be shown that with large training set of size N, $\hat{f}_{knn}(x) \to \mathbb{E}_{Y|X}\left[Y|X=x\right]$ as $N, k \to \infty$ with $k/N \to 0$. However, the rate of convergence decrease when dimension p increases.

Ex. **OLS Linear Regression** asssumes the function is linear in x, i.e.

$$\hat{f}_{ols}(x) = x^\top \beta$$

This is a model-based approach. We plug in this functional form into EPE:

$$EPE(\hat{f}_{ols}) = \mathbb{E}\left[(Y - X^\top\beta)^2\right] \tag{5}$$

Take FOC:

$$\begin{aligned} \frac{\partial}{\partial \beta}\mathbb{E}\left[(Y - X^\top\beta)^2\right] &= \mathbb{E}\left[\frac{\partial}{\partial \beta}(Y - X^\top\beta)^2\right] \\ &= \mathbb{E}\left[2X(Y - X^\top\beta)\right] = 0 \\ \Rightarrow \beta &= \mathbb{E}\left[XX^\top\right]^{-1}\mathbb{E}\left[XY\right] \end{aligned} \tag{6}$$

And then by replacing the expection by averaging over the dataset we obtain the familiar OLS estimator solution.

We have seen that both KNN and the OLS end up approximating conditional expectations by averages, but they differ in terms of model assumptions.

- $\hat{f}_{ols}(x)$ is assumed to be a globally linear function.
- $\hat{f}_{knn}(x)$ is assumed to be a locally constant function.

1.2 Categorical Dependent Variable

In this setting our dependent (random) varibale $G \in \{\mathcal{G}_1, \mathcal{G}_2, ..., \mathcal{G}_K\} =: \mathcal{G}$, i.e. it has K types in total. $\hat{G} = \hat{G}(X)$ is the prediction. We can use a $K \times K$ matrix $\boldsymbol{L}$ to represent the loss, with L_{kl} being the loss of classifying a $\mathcal{G}_k$ observation as $\mathcal{G}_l$. Usually we will use *zero-one* loss function, which charges 1 unit for all misclassifications uniformly, i.e.

$$L(G, \hat{G}(X)) = \begin{cases} 0 & G = \hat{G}(X), \\ 1 & G \neq \hat{G}(X) \end{cases} = \mathbb{1}_{\{G \neq \hat{G}(X)\}} \tag{7}$$

We proceed in the same fashion

$$\begin{aligned} EPE(\hat{G}) &= \mathbb{E}\left[L(G, \hat{G}(X))\right] \\ &= \int_X \left(\sum_{k=1}^{K} L(\mathcal{G}_k, \hat{G}(X))\mathbb{P}\left(G = \mathcal{G}_k, X = x\right)\right) dx \\ &= \int_X p_X(x) \left(\sum_{k=1}^{K} L(\mathcal{G}_k, \hat{G}(X)) p_{G|X}(\mathcal{G}_k|x)\right) dx \\ &= \mathbb{E}_X \left[\sum_{k=1}^{K} L(\mathcal{G}_k, \hat{G}(X)) p_{G|X}(\mathcal{G}_k|x)\right] \end{aligned} \tag{8}$$

And again it suffices to minimize EPE pointwise w.r.t. x,

$$\begin{aligned} \hat{G}(x) &= \underset{g\in\mathcal{G}}{\operatorname{argmin}} \sum_{k=1}^{K} L(\mathcal{G}_k, g) p_{G|X}(\mathcal{G}_k|x) \\ &= \underset{g\in\mathcal{G}}{\operatorname{argmin}} \sum_{k=1}^{K} \mathbb{1}_{\{g\neq\mathcal{G}_k\}}\mathbb{P}\left(G = \mathcal{G}_k|X = x\right) \\ &= \underset{g\in\mathcal{G}}{\operatorname{argmin}} \left(1 - \mathbb{P}\left(G = g|X = x\right)\right) \end{aligned} \tag{9}$$

In another word, $\hat{G}(x) = \mathcal{G}_k \iff$

$$\mathcal{G}_k = \underset{g\in\mathcal{G}}{\operatorname{argmax}}\, \mathbb{P}\left(G = g|X = x\right) \qquad \iff$$

$$\mathbb{P}\left(G = \mathcal{G}_k|X = x\right) = \max_{g\in\mathcal{G}} \mathbb{P}\left(G = g|X = x\right)$$

which says that, given $X = x$, $G = \mathcal{G}_k$ has the greatest conditional probability. This solution is known as **Bayes Classifier**. And the error rate is called the *Bayes Rate*. If we know the generating distribution of dataset, the Bayes classifier decision boundary can be specifed exactly.

The dummy variable regression approach fits in this framework, and is just another way of representing the Bayes classifier. Because we use $\hat{Y}(X) = \mathbb{E}\left[Y|X\right] = \mathbb{P}\left(\mathcal{G}_1|X\right)$ if $\mathcal{G}_1$ corresponds to $Y = 1$.

2 Bias-Variance Decomposition

The increase dimensionality p cast shadow on our established intuition, that we could always approximate the conditional expectation with k-nearest neighbour averaging.

2.1 Curse of Dimensionality

- **Neighbourhood not-so-local**: Consider inputs $X \sim \text{Uniform}([0, 1]^p)$, uniformly distributed in p-dim hypercube. We want to use hypercubical neighbourhood to capture a fraction r of all obs. The expected length of edge is $e(p) = r^{\frac{1}{p}}$, increases with p exponentially. In high dimension, we need to look at a wide range on each input variable to capture a desirable fraction of data. Such neibourhood is not "local" any more.
- **Sample close to boundary**: Each sample point becomes closer to boundary of the sample space. And the prediction is more difficult near the edges.

- **Sparsity of sample**: The sampling density is proportional to $N^{1/p}$. The number of obs required to form a desirably dense sample grows exponentially with dimensionality. Thus in high dimensions training samples *sparsely* populate the input space.

2.2 Bias-Variance Decomposition

We reconsider some familiar definition in the context of statistical learning. Consider random variables X, Y, Y can be a *deterministic function* of X, say

$$Y = f(X)$$

The **true** relationship between X and Y can also be *stochastic*, for example, in linear regression we have

$$Y = X^\top \beta + \epsilon$$

where ϵ is some kind of Gaussian random variable.
In the first setting, $f(\cdot)$ is a deterministic function, but **unknown**, i.e. we always estimate the function with $\hat{f}(\cdot)$, with the help of some kind of supervised learning algorithm, and a traing set $\mathcal{T}$ that contains the notion of randomness, because it is essentially a random sample of (X, Y), to which we have no complete control.

Hence the estimation $\hat{f}(\cdot)$ relies on the random sample $\mathcal{T}$. At a given point x, our estimator for Y is $\hat{Y}(x) = \hat{f}(x)$ is therefore also a random variable that relies on $\mathcal{T}$. When we take expectation of $\hat{Y}$, we are actually averaging over all possible training sets $\mathcal{T}$, which is the meaning of the notion

$$\mathbb{E}_{\mathcal{T}}\left[\hat{f}(x)\right] = \int_{\text{all } \mathcal{T}} \hat{f}(x) d\mathbb{P}$$

Def. **Bias (Supervised Learning)**: at a given point x, the bias of an estimator $\hat{f}(x)$ of the deterministic & true value $f(x)$ is

$$\text{Bias}(\hat{f}(x)) = \mathbb{E}_{\mathcal{T}}\left[\hat{f}(x)\right] - f(x)$$

Def. **Variance (Supervised Learning)**: at a given point x, the variance of a predictor $\hat{f}(x)$ of the deterministic & true value $f(x)$ is the centerred second moment wrt randomly sampled training set

$$\mathbb{V}\text{ar}_{\mathcal{T}}\left[\hat{f}(x)\right] = \mathbb{E}_{\mathcal{T}}\left[\left(\hat{f}(x) - \mathbb{E}_{\mathcal{T}}\left[\hat{f}(x)\right]\right)^2\right]$$

Def. **Mean Squared Error (MSE)**: deterministic model $Y = f(X)$, given point x,

$$MSE(\hat{f}(x)) := \mathbb{E}_{\mathcal{T}}\left[\left(\hat{f}(x) - f(x)\right)^2\right]$$

For simplicity we denote $\hat{Y} = \hat{Y}(x)$, $y = f(x)$ at a given x, the first is a random varible, the second is deterministic. We have

$$\begin{aligned}
MSE(\hat{Y}) &= \mathbb{E}_{\mathcal{T}}\left[\left(\hat{Y} - y\right)^2\right] \\
&= \mathbb{E}_{\mathcal{T}}\left[\left(\hat{Y} - \mathbb{E}_{\mathcal{T}}[\hat{Y}] + \mathbb{E}_{\mathcal{T}}[\hat{Y}] - y\right)^2\right] \qquad (10) \\
&= \mathbb{E}_{\mathcal{T}}\left[\left(\hat{Y} - \mathbb{E}_{\mathcal{T}}[\hat{Y}]\right)^2 + \left(\mathbb{E}_{\mathcal{T}}[\hat{Y}] - y\right)^2 + 2(\hat{Y} - \mathbb{E}_{\mathcal{T}}[\hat{Y}])(\mathbb{E}_{\mathcal{T}}[\hat{Y}] - y)\right] (\dagger)
\end{aligned}$$

Note that the second term is a constant wrt. expectation, the cross term is 0. Hence

$$\begin{aligned}(\dagger) &= \mathbb{E}_{\mathcal{T}}\left[\left(\hat{Y} - \mathbb{E}_{\mathcal{T}}[\hat{Y}]\right)^2\right] + \left(\mathbb{E}_{\mathcal{T}}[\hat{Y}] - y\right)^2 \\ &= \mathbb{V}\mathrm{ar}_{\mathcal{T}}[\hat{Y}] + \mathrm{Bias}^2[\hat{Y}]\end{aligned} \tag{11}$$

which is called the bias-variance decomposition. Note that in this setting the mapping $f(\cdot)$ is deterministic, so given x; y is also deterministic, the only factor of randomness is training set $\mathcal{T}$. Hence

$$MSE(\hat{Y}) = \mathbb{E}_{\mathcal{T}}\left[\left(\hat{Y} - y\right)^2\right] = \mathbb{E}\left[\hat{Y} - y\right] = EPE(\hat{Y})$$

We now consider the second setting

$$Y = X^\top \beta + \epsilon, \qquad \epsilon \sim \mathcal{N}(0, \sigma^2)$$

Here upper cases X, Y mean random varaible (scalar or vector). In the following text we use the bold notation $\boldsymbol{X}, \boldsymbol{y}$ to represent the matrix (training set). By OLS theory we have

$$\hat{\beta} = (\boldsymbol{X}^\top \boldsymbol{X})^{-1}\boldsymbol{X}^\top \boldsymbol{y} = (\boldsymbol{X}^\top \boldsymbol{X})^{-1}\boldsymbol{X}^\top(\boldsymbol{X}\beta + \epsilon) = \beta + (\boldsymbol{X}^\top \boldsymbol{X})^{-1}\boldsymbol{X}^\top \epsilon$$

So at a given point $\boldsymbol{x}_0$ (col vector, say $\boldsymbol{x}_0 = \boldsymbol{0}$), $\hat{y}_0 = \hat{f}(\boldsymbol{x}_0) = \boldsymbol{x}_0^\top \hat{\beta} = \boldsymbol{x}_0^\top \beta + \boldsymbol{x}_0^\top (\boldsymbol{X}^\top \boldsymbol{X})^{-1}\boldsymbol{X}^\top \epsilon$. Now the randomness comes from two sources: the sampling of trainging data $\boldsymbol{X}, \boldsymbol{y}$ and the noise ϵ within the mapping between X, Y, which affects the true value $y_0 = \boldsymbol{x}_0^\top \beta + \epsilon$.

Therefore, the expectation is splitted as $EPE(\hat{y}_0) = \mathbb{E}\left[(y_0 - \hat{y}_0)^2\right] = \mathbb{E}_{\mathcal{T}}\left[\mathbb{E}_{y_0|x_0}\left[(y_0 - \hat{y}_0)^2\right]\right]$. Since ϵ is independent to sampling $\mathcal{T}$, hence the expectation commutes: $\mathbb{E}_{\mathcal{T}}\mathbb{E}_{y_0|x_0} = \mathbb{E}_{y_0|x_0}\mathbb{E}_{\mathcal{T}}$. Further more, sampling $\mathcal{T}$ can be regarded as sampling $\boldsymbol{X}$ first, then obtain $\boldsymbol{y} = \boldsymbol{X}\beta + \boldsymbol{\epsilon}$. Hence moreover we have $\mathbb{E}_{\mathcal{T}} = \mathbb{E}_{\boldsymbol{X}}\mathbb{E}_{\boldsymbol{y}|\boldsymbol{X}}$.

Now we begin to deal with the $EPE(\hat{y}_0)$, we write

$$\begin{aligned}EPE(\hat{y}_0) &= \mathbb{E}_{\mathcal{T}}\left[\mathbb{E}_{y_0|x_0}\left[(y_0 - \hat{y}_0)^2\right]\right] \\ &= \mathbb{E}_{\mathcal{T}}\left[\mathbb{E}_{y_0|x_0}\left[\left(y_0 - \boldsymbol{x}_0^\top\beta + \boldsymbol{x}_0^\top\beta - \mathbb{E}_{\mathcal{T}}\hat{y}_0 + \mathbb{E}_{\mathcal{T}}\hat{y}_0 - \hat{y}_0\right)^2\right]\right] \\ &= \mathbb{E}_{\mathcal{T}}\left[\mathbb{E}_{y_0|x_0}\left[\left((y_0 - \boldsymbol{x}_0^\top\beta) + (\boldsymbol{x}_0^\top\beta - \mathbb{E}_{\mathcal{T}}\hat{y}_0) + (\mathbb{E}_{\mathcal{T}}\hat{y}_0 - \hat{y}_0)\right)^2\right]\right] \\ &= \mathbb{E}_{\mathcal{T}}\left[\mathbb{E}_{y_0|x_0}\left[(U_1 + U_2 + U_3)^2\right]\right] \quad (*)\end{aligned} \tag{12}$$

We consider the properties of these three terms:

- U_1: We note that $U_1 = \epsilon$ has nothing to do with $\boldsymbol{X}, \boldsymbol{y}$, so $\mathbb{E}_{\mathcal{T}}\left[U_1\right] = U_1$. And $\mathbb{E}_{y_0|x_0}(\epsilon) = \mathbb{E}(\epsilon) = 0$.
- U_2: The second factor U_2, by definition, is the Bias of $\hat{y}_0$ with respect to the deterministic part of y_0, i.e. $\boldsymbol{x}_0^\top\beta$. By econometrics theory, the OLS estimator $\hat{\beta}$ is unbiased, so we also expect $U_2 = 0$. Actually,

$$\begin{aligned}\mathbb{E}_{\mathcal{T}}\left[\hat{y}_0\right] &= \mathbb{E}_{\mathcal{T}}\left[\boldsymbol{x}_0^\top\beta + \boldsymbol{x}_0^\top(\boldsymbol{X}^\top\boldsymbol{X})^{-1}\boldsymbol{X}^\top\epsilon\right] \\ &= \boldsymbol{x}_0^\top\beta + \mathbb{E}_{\boldsymbol{X}}\mathbb{E}_{\boldsymbol{y}|\boldsymbol{X}}\left[\boldsymbol{x}_0^\top(\boldsymbol{X}^\top\boldsymbol{X})^{-1}\boldsymbol{X}^\top\epsilon\right] \\ &= \boldsymbol{x}_0^\top\beta + \mathbb{E}_{\boldsymbol{X}}\left[\boldsymbol{x}_0^\top(\boldsymbol{X}^\top\boldsymbol{X})^{-1}\boldsymbol{X}^\top \cdot \mathbb{E}_{\boldsymbol{y}|\boldsymbol{X}}\left(\epsilon\right)\right] \\ &= \boldsymbol{x}_0^\top\beta\end{aligned} \tag{13}$$

  Therefore $U_2 = 0$ indeed, and the estimator $\hat{y}_0$ is unbiased.

- U_3: Clearly $\mathbb{E}_{\mathcal{T}}[U_3] = \mathbb{E}_{\mathcal{T}}\hat{y}_0 - \mathbb{E}_{\mathcal{T}}\hat{y}_0 = 0$; and it has nothing to do with ϵ, i.e. $\mathbb{E}_{y_0|x_0}[U_3] = U_3$.

Since $U_2 = 0$, the cross terms $2U_1U_2$ and $2U_2U_3$ are zero. And the third cross term $\mathbb{E}_{\mathcal{T}}\left[\mathbb{E}_{y_0|x_0}[2U_1U_3]\right] = \mathbb{E}_{\mathcal{T}}\left[2U_3\mathbb{E}_{y_0|x_0}[U_1]\right] = 0$, using the fact that U_3 can be taken out from inner expectation and that $\mathbb{E}_{y_0|x_0}[U_1] = 0$.

Now we are almost done, only three squared terms (with $U_2^2 = 0$) remaining. We write

$$\begin{aligned}(*) &= \mathbb{E}_{y_0|x_0}U_1^2 + \mathbb{E}_{\mathcal{T}}U_2^2 + \mathbb{E}_{\mathcal{T}}U_3^2 = \mathbb{C}\text{ov}_{y_0|x_0}[y_0] + \text{Bias}^2(\hat{y}_0) + \mathbb{C}\text{ov}_{\mathcal{T}}[\hat{y}_0] \\ &= \mathbb{C}\text{ov}[\epsilon] + 0 + \mathbb{C}\text{ov}_{\mathcal{T}}[\hat{y}_0] = \sigma^2 + \mathbb{C}\text{ov}_{\mathcal{T}}[\hat{y}_0]\end{aligned} \tag{14}$$

Recall that $\hat{y}_0$ is unbiased to $\boldsymbol{x}_0^\top\beta$, hence

$$\begin{aligned}\mathbb{C}\text{ov}_{\mathcal{T}}[\hat{y}_0] &= \mathbb{E}_{\mathcal{T}}\left[(\mathbb{E}_{\mathcal{T}}\hat{y}_0 - \hat{y}_0)(\mathbb{E}_{\mathcal{T}}\hat{y}_0 - \hat{y}_0)^\top\right] = \mathbb{E}_{\mathcal{T}}\left[(\boldsymbol{x}_0^\top\beta - \hat{y}_0)(\boldsymbol{x}_0^\top\beta - \hat{y}_0)^\top\right] \\ &= \mathbb{E}_{\mathcal{T}}\left[\boldsymbol{x}_0^\top(\boldsymbol{X}^\top\boldsymbol{X})^{-1}\boldsymbol{X}^\top\epsilon\epsilon^\top\boldsymbol{X}(\boldsymbol{X}^\top\boldsymbol{X})^{-1}\boldsymbol{x}_0\right] \\ &= \sigma^2\boldsymbol{x}_0^\top\mathbb{E}_{\boldsymbol{X}}\left[(\boldsymbol{X}^\top\boldsymbol{X})^{-1}\right]\boldsymbol{x}_0\end{aligned} \tag{15}$$

where σ^2 comes out because $\mathbb{E}_{\boldsymbol{y}|\boldsymbol{X}}(\epsilon\epsilon^\top) = \mathbb{V}\text{ar}[\epsilon] = \sigma^2$, nothing to do with $\boldsymbol{X}$. Use a familiar technique in econometrics,

$$\sigma^2\boldsymbol{x}_0^\top\mathbb{E}_{\boldsymbol{X}}\left[(\boldsymbol{X}^\top\boldsymbol{X})^{-1}\right]\boldsymbol{x}_0 = \frac{1}{N}\sigma^2\boldsymbol{x}_0^\top\mathbb{E}_{\boldsymbol{X}}\left[\left(\frac{\boldsymbol{X}^\top\boldsymbol{X}}{N}\right)^{-1}\right]\boldsymbol{x}_0$$

when assuming the distribution that give rise to $\boldsymbol{X}, \boldsymbol{x}_0$ has mean 0, as $N \to \infty$, $\boldsymbol{X}^\top\boldsymbol{X}/N \to \mathbb{C}\text{ov}[\boldsymbol{X}] = \mathbb{C}\text{ov}[\boldsymbol{x}_0] =: \Sigma$, which is a $p \times p$ matrix.

$$\begin{aligned}\mathbb{E}_{x_0}\lim_{N\to\infty}\frac{1}{N}\sigma^2\boldsymbol{x}_0^\top\mathbb{E}_{\boldsymbol{X}}\left[\left(\frac{\boldsymbol{X}^\top\boldsymbol{X}}{N}\right)^{-1}\right]\boldsymbol{x}_0 &\approx \frac{1}{N}\sigma^2\mathbb{E}_{\boldsymbol{x}_0}\left[\boldsymbol{x}_0^\top\Sigma^{-1}\boldsymbol{x}_0\right] \\ &= \frac{1}{N}\sigma^2\text{trace}\left(\Sigma^{-1}\mathbb{E}_{\boldsymbol{x}_0}\left[\boldsymbol{x}_0\boldsymbol{x}_0^\top\right]\right) \\ &= \frac{1}{N}\sigma^2\text{trace}(\boldsymbol{I}_p) = \frac{p}{N}\sigma^2\end{aligned} \tag{16}$$

i.e.

$$\mathbb{E}_{\boldsymbol{x}_0}EPE(\hat{y}_0) \approx \sigma^2\left(1 + \frac{p}{N}\right) \tag{17}$$

Which indicates that the expectation (with respect to the fixed point we pick) of EPE grows **linearly** with p, as well as the variance of the estimator. This is a much better case, since we don't require the sample size N to grow exponentially an when N is large, the linear growth in variance is not a big deal.

This is reason why we say that OLS method is relatively *stable* compared with the 1-Nearest Neighbour case we discussed previously. When the true functional relation is linear, the bias is also small. But if the guess of linear structure is wrong, we will expect high bias. For example

1. $Y = X^\top\beta + \epsilon$, the linear model has no bias and small variance, and EPE slightly above $1\sigma^2$. In general settings it will dominate 1-Nearest Neighbour whose EPE is always above $2\sigma^2$. This ratio (EPE_{1nn}/EPE_{ols}) grows with the dimension.
2. $Y = \frac{1}{2}(X_1 + 1)^3$, the linear is biased this time, which moderates the ratio, but still increases with dimension. In low dimension, 1NN may dominates, and in high dimension, OLS dominates.

3. The case where linear assumption is seriously wrong, OLS bias is high, and 1-Nearest Neighbour may dominate OLS.

The moral of this example is that, by *imposing some heavy restriction/structures* on the class of models being fitted, we can avoid the curse of dimensionality and reduce variance. But when the structure is wrong, bias term may dominates. There is always a trade-off between bias and variance.

3 Statistical Models

3.1 A Statistical Model for Joint Dist $p_{X,Y}$

For *quantitative* output Y, suppose the true relation is

$$Y = f(X) + \epsilon$$

with $\mathbb{E}[\epsilon] = 0$, independent of X. We take expectation conditional on X: $\mathbb{E}[Y|X = x] = f(x)$ (rely on the assumption that error has zero mean, and indep.) And the additive error model ϵ is a useful approximation, it indicates that pair (X, Y) does not has deterministic relation, and there are other unmeasured variables that contributes to Y. The additive model assumes that we capture all of them via ϵ.

For *qualitative* output G, recall that we label an output as $\mathcal{G}_k$ if $\mathbb{P}(G = \mathcal{G}_k|X = x)$ is maximized. So we model the conditional probability $p_{G|X}(g|x)$ directly.

3.2 Function Approximation

We want to view the supervised learning problem as a function estimation problem. Usually we fit the function in a hypothesis space that contains a class of functions parametrized by θ. We view the parametrized function $f_\theta(x)$ as a surface in the $p + 1$ dimensional space $\mathbb{R}^p \times \mathbb{R}$. We observe noisy realizations $\mathcal{T}$, and find set of parameters such that the fitted surface gets as close to the observed points as possible, where close is measured by loss like residual sum of squres $RSS(\theta) = \sum_{i=1}^N (y_i - f_\theta(x_i))^2$, etc.

Another estimation method besides least square is the **Maximum Likelihood Estimation**.

Def. **Log Likelihood**: Given an i.i.d. random sample $y_i, i = 1, 2, ..., N$ generated from a probability density $p_Y(y; \theta)$, parametrized by θ. The log likelihood of the observed sample is

$$L(\theta) = \sum_{i=1}^N \log p(y; \theta) = \log \left(\mathbb{P}\left(\bigcap_{i=1}^N \{Y_i = y_i\}; \theta \right) \right)$$

This is exactly the log probability in which we see the sample $\{y_i\}$ conditional on the fact that the underlying density has parameter θ.

The maximum likelihood principle goes that we should choose theta such that this conditional probability of obtaining the observed sample is maximized.

Ex Suppose the true model is error (white noise) additive with parameter θ: $Y = f_\theta(X) + \epsilon$, then we obtain the conditional distribution of Y on parameter and inputs: $Y|X; \theta \sim \mathcal{N}(f_\theta(X), \sigma^2)$

$$\log \mathbb{P}(Y_i = y_i|X_i = x_i; \theta) = -\log \sqrt{2\pi}\sigma - \frac{(y_i - f_\theta(x_i))^2}{2\sigma^2}$$

$$L(\theta) = -N \log \sqrt{2\pi}\sigma - \frac{1}{2\sigma^2} \sum_{i=1}^N (y_i - f_\theta(x_i))^2$$

Ex. In the qualitative case we fit, for every class k, the conditional probability $\mathbb{P}\left(G = \mathcal{G}_k | X = x\right) = p_{G|X}(\mathcal{G}_k|x)$. When this is parametrized by θ we write

$$P_{k,\theta}(x) := \mathbb{P}\left(G = \mathcal{G}_k | X = x; \theta\right)$$

And for an i.i.d. sample $\{g_i\}_{i=1}^N$, the log conditional probability of seeing it on θ and $\boldsymbol{X}$ is

$$\begin{aligned} \log\left(\mathbb{P}\left(\bigcap_{i=1}^N \{G_i = g_i | X_i = x_i\}; \theta\right)\right) &= \log \prod_{i=1}^N \mathbb{P}\left(G_i = g_i | X_i = x_i; \theta\right) \\ &= \sum_{i=1}^N \log p_{g_i,\theta}(x_i) \end{aligned} \tag{18}$$

which is also referred to as the *cross-entropy*.

4 Structured Regression Models

There are infinitely many solutions of minimizing $RSS(\hat{f}) = \sum_{i=1}^N (y_i - \hat{f}(x_i))^2$. And any particular one might perform poorly on test set. Hence we must restrict the eligible solutions to a smaller family of functions.

Most of these restrictions can be considered as *complexity restrictions*. Which means that we assume the function to have regular behavior in small neighbourhoods of the input space. I.e. $\hat{f}$ exhibits special structure, like locally constant, linear or polynomial, for some x that are sufficiently close to each other.

The strength of the constraints is dictated by neighbourhood size, the larger the stronger. For example, locally constant on infinitesimally small neighbourhood is no constraint at all; but a global linear assumption is a strong one. Besides, some method directly specify the metric and size of neighbourhood, while others define these implicitly or adaptively. From previous discussion we know that all algorithms that attempts to produce functions in smally isotropic neighbourhoods will see problems in high dimensions; and those who overcome the dimensionality problems often have implicit or adaptive metrics for neighbourhood measures.

We now introduce some broad ideas of restrictions.

4.1 Roughness Penalty and Bayesian Methods

A class of loss function is controlled by explicitly penalizing RSS with a term

$$L(f;\lambda)_{PRSS} = RSS(f) + \lambda J(f)$$

where *functional* $J : \mathcal{F}(f) \to \mathbb{R}$ will be large for f that vary too rapidly over the neighbourhood. For example

$$L(f;\lambda)_{PRSS} = \sum_{i=1}^N \|y_i - f(x_i)\|_2 + \lambda \int (f''(x))^2 dx$$

By choosing λ we control the amount of penalty: $\lambda = 0$ no penalty, $\lambda = \infty$ only allow linear functions.

The penalty function (regularization method) expresses our *prior* belief that the class of funtion we seek exhibit a certain type of smooth behavior. In Bayesian framework, J corresponds to a log-prior, $L(f;\lambda)$ is log-posterior distribution.

4.2 Kernel Methods and Local Regression

These methods attempt to explicitly estimate functions by specifying the nature of local neighbourhood. A *kernel function* $K_\lambda(x_0, x)$ assigns weights to point x in a neighbourhood around x_0. For example, the Gaussian kernel

$$K_\lambda(x_0, x) = \frac{1}{\lambda} \exp\left(-\frac{\|x - x_0\|^2}{2\lambda}\right)$$

the weights assigned to x die exponentially with squared norm of distance from x_0. λ corresponds to variance, and controls the size of neighbourhood. A simplest estimator is weighted average $\hat{f}(x_0) = \sum_{i=1}^N K_\lambda(x_0, x_i)y_i / \sum_{i=1}^N K_\lambda(x_0, x_i)$ In general we define a local regression estimate with parameter θ, we seek to choose $\hat{\theta}$ that minimizes

$$RSS(f_\theta, x_0) = \sum_{i=1}^N K_\lambda(x_0, x_i)(y_i - f_\theta(x_i))^2$$

Pick $f_\theta(x) = \theta_0$, the solution is the weighted average estimate. And KNN method can be thought of using kernel $K_k(x, x_0) = \mathbb{1}_{\{\|x - x_0\| \leq \|x_{(k)} - x_0\|\}}$, where $x_{(k)}$ is the data with k-th rank in distance to x_0.

4.3 Basis Function and Dictionary Methods

Consider the model f as a linear expansion of basis functions

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

where each $h_m(\cdot)$ is a function of x with no implicit parameters, say $x^2, \sin(x)$, etc. An example is polynomial spline basis functions, where "spline" refers to a piece of continuous function between two knots, and joined up we have a piecewise polynomial function with continuity of certain degree. Other examples are

- *Radial basis functions*, which are symmetirc p-dimensional kernels located at particular centroids. we have model $f_\theta(x) = \sum_{m=1}^M K_{\lambda_m}(\mu_m, x)\theta_m$.
- *Activation basis functions* are those we used in neural network, consider a sigmoid activation $\sigma(x) = \frac{1}{1+e^{-x}}$, and we model estimator as $f_\theta(x) = \sum_{m=1}^M \beta_m \sigma(\alpha_m^\top x + b_m)$.

When one has a set/dictionary $\mathcal{D}$ of candidate basis functions, and build model by employing some search algorithm to adaptively choose basis; this is often referred to as *dictionary methods.*

5 Model Selection

All the structured models above includes some kind of *smoothing* or *complexity* hyperparameter to be manually determined,

- Multiplier λ of the penalty term in roughness penalty method.
- The width of the kernel in the kernel method.
- The number of basis functions in the dictionary method.

There are *competing forces* that affects the predictive ability of our estimators. We use a KNN example to illustrate.

Ex. At a test point x_0, our k-nearest neighbour fit is $\hat{f}_k(x_0) = \frac{1}{k}\sum_{l=1}^{k} y_{(l)}$, where $(x_{(l)}, y_{(l)})$ is the sample point ranked l-th in distance to $\|x - x_0\|$. Suppose the true relation is $Y = f(X) + \epsilon$ with $\mathbb{E}[\epsilon] = 0, \mathbb{E}\left[\epsilon^2\right] = \sigma^2$. For simplicity we assume no randomness in $\boldsymbol{X}$, i.e. the x_i in training set $\mathcal{T}$ is fixed in advance. Hence $\hat{f}_k(x_0) = \frac{1}{k}\sum_{l=1}^{k} f(x_{(l)}) + \epsilon_{(l)}$. $\mathbb{E}_{\mathcal{T}}\left[\hat{f}_k(x_0)\right] = \frac{1}{k}\sum_{l=1}^{k} f(x_{(l)})$; $\mathbb{V}\text{ar}_{\mathcal{T}}\left[\hat{f}_k(x_0)\right] = \sigma^2/k$.

We calculate the expected prediction error at x_0:

$$\begin{aligned}
EPE(\hat{f}_k(x_0)) &= \mathbb{E}_{\mathcal{T}}\left[(Y(X) - \hat{f}_k(X))^2 | X = x_0\right] = \mathbb{E}_{\mathcal{T}}\left[(\epsilon + f(x_0) - \hat{f}_k(x_0))^2\right] \\
&= \mathbb{E}_{\mathcal{T}}\left[\Big(\epsilon + (f(x_0) - \mathbb{E}_{\mathcal{T}}\hat{f}_k(x_0)) + (\mathbb{E}_{\mathcal{T}}\hat{f}_k(x_0) - \hat{f}_k(x_0))\Big)^2\right] \\
&= \sigma^2 + \text{Bias}^2(\hat{f}_k(x_0)) + \mathbb{V}\text{ar}_{\mathcal{T}}\left[\hat{f}_k(x_0)\right] \quad \text{(the cross terms die)} \qquad (19) \\
&= \sigma^2 + \left(f(x_0) - \frac{1}{k}\sum_{l=1}^{k} f(x_{(l)})\right)^2 + \frac{\sigma^2}{k}
\end{aligned}$$

- The first term σ^2 is the variance of the new test case $y(x_0)$, and is due to ϵ, because the relation is non-deterministic. This is call the *irreducible error.*
- The second term is the Bias of estimator, and is likely to increase with k, because large k yields a wider neighbourhood, it is likely that new neighbours introduced are faraway from the true value of $f(x_0)$, so does the average.
- The third term is the variance, clearly it is decreasing with k. Therefore, as k varies, there is a **Bias-Variance Trade Off**, as we have also seen in section 2. Generally, as the model complexity increases, the variance tends to increase and the bias reduced. For KNN the bias is controlled by k. Typically we want to trade these two off to minimize the test error as a whole. One must note that the *training error* $\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)$ cannot estimate test error, since it does not account for model complexity. Generally, training error always decreases with model complexity, i.e. whenever we fit the specific training data harder. And when the model adapts itself too much to the training set, it will not generalize well and have large **variance**, which we call as **overfit**. In other case when the model is not complex enough, it will have large **bias**, which we refer to as **underfit**.