MLISP: Machine Learning in Signal Processing

Lecture 4

Prof. V. I. Morgenshtern

Scribe: M. Solomon

Illustrations: The elements of statistical learning, Hastie, Tibshirani, Friedman

Agenda:

1. Basics of statistical decision theory
2. Bayesian optimality
3. Conditional expectation as the optimal predictor
4. Conditional expectation in the linear case
5. Maximum conditional probability as the optimal classifier

1 Basics of statistical decision theory

In this lecture we will touch upon the basics of statistical decision theory. Bayesian setup is the case in which the joint probability distribution from which our data was generated is known to us. We will see that in this case, there is an optimal solution to the regression and the classification problems. These are known as 'Bayes optimal predictors/classifiers'. In practice, the joint probability distribution is never known. We will see how the two algorithms, the nearest neighbors predictor and the linear model, attempt to approximate the unachievable optimal Bayesian solution.

2 Bayes optimal predictor: condition expectation

Let $\mathtt{x} \in \mathbb{R}^p$ denote a real valued random input vector, and $\mathtt{y} \in \mathbb{R}$ a real valued output vector. Assume that $(\mathtt{x}\,,\,\mathtt{y})$ pairs are drawn from a joint probability distribution $\mathbb{P}_{\mathtt{x},\mathtt{y}}(\cdot,\cdot)$.

Notation: we will use the notation and write $\mathtt{x}$ for random vectors, $\mathtt{y}$ for random scalars (not as bold as vectors), $\mathbf{x}$ for non-random vectors, y for non-random scalars.

Example: For example we might have the joint distribution $\mathbb{P}_{\mathtt{x},\mathtt{y}}(\cdot,\cdot)$ defined (implicitly) as follows. The house size, $\mathtt{x}_1 \sim \mathcal{U}[70, 200]$ sq. meters. The number of bedrooms, $\mathtt{x}_2 = \mathtt{x}_1/35 + \mathcal{U}[-1, 0, 1]$. The house price $\mathtt{y} = \alpha_1\mathtt{x}_1 + \alpha_2\mathtt{x}_2 + \mathcal{N}(0, 100000)$ US dollars. Here and in the future, $\mathcal{U}$ denotes the uniform distribution and $\mathcal{N}$ denotes the Gaussian distribution.

Recall, we seek a function $h(\cdot)$ for predicting $\mathtt{y}$ given $\mathtt{x}$. Let us use the squared error loss as before:

$$L(y, h(\mathbf{x})) = (y - h(\mathbf{x}))^2. \tag{1}$$

Then the expected prediction error is given by:

$$\begin{aligned} \mathrm{EPE}(h) &= \mathbb{E}\left\{(\mathtt{y} - h(\mathtt{x}))^2\right\} \\ &= \int (y - h(\mathbf{x}))^2 p(\mathbf{x}, y) d\mathbf{x} dy \end{aligned} \tag{2}$$

where $p(\cdot, \cdot) = p_{\mathtt{x},\mathtt{y}}(\cdot, \cdot)$ denotes the joint probability density function of $(\mathtt{x}, \mathtt{y})$. In the Bayesian formulation, the goal is to find $h(\cdot)$ that minimizes $\mathrm{EPE}(h)$.

Using the formula for conditional expectation:

$$\mathrm{EPE}(h) = \mathbb{E}_{\mathtt{x}} \mathbb{E}_{\mathtt{y}|\mathtt{x}} \left\{(\mathtt{y} - h(\mathtt{x}))^2 | \mathtt{x}\right\}. \tag{3}$$

Therefore, we can minimize EPE point-wise:

$$\hat{h}(\mathbf{x}) = \arg\min_{c} \mathbb{E}_{\mathtt{y}|\mathtt{x}} \left\{(\mathtt{y} - c)^2 | \mathtt{x} = \mathbf{x}\right\} \tag{4}$$

for each $\mathbf{x}$. The solution is $\hat{h}(\mathbf{x}) = \mathbb{E}\left\{\mathtt{y} | \mathtt{x} = \mathbf{x}\right\}$, which is also known as the *regression function*.

Proof: The minimum is achieved at the point where the derivative of the function is zero:

$$\frac{d}{dc} \mathbb{E}_{\mathtt{y}|\mathtt{x}} \left\{(\mathtt{y} - c)^2 | \mathtt{x} = \mathbf{x}\right\} = 0 \tag{5}$$

$$\Leftrightarrow \quad - \int p_{\mathtt{y}|\mathtt{x}}(y | \mathtt{x} = \mathbf{x}) 2(y - c) dy = 0 \tag{6}$$

$$\Leftrightarrow \quad c = \int_y p_{\mathtt{y}|\mathtt{x}}(y | \mathtt{x} = \mathbf{x}) y dy \tag{7}$$

$$\Leftrightarrow \quad c = \mathbb{E}\{\mathtt{y} | \mathtt{x} = \mathbf{x}\}. \tag{8}$$

2.1 Nearest neighbors as an approximation to optimal Bayes solution

In practice we cannot use the $\hat{h}(\mathbf{x}) = \mathbb{E}\{\mathtt{y} | \mathtt{x} = \mathbf{x}\}$ formula directly because we don't know the joint probability density $p_{\mathtt{x},\mathtt{y}}(\cdot, \cdot)$, which is the whole point of learning.

The nearest-neighbor methods attempt to directly implement the optimal recipe using the training data $(\mathtt{x}^{(i)}, \mathtt{y}^{(i)})$ to estimate $\mathbb{E}\{\mathtt{y} | \mathtt{x} = \mathbf{x}\}$. Specifically, at each point $\mathbf{x}$, we average all those $\mathtt{y}^{(i)}$'s with the corresponding inputs $\mathtt{x}^{(i)} \approx \mathbf{x}$:

$$\hat{h}_{\mathrm{NN}}(\mathbf{x}) = \mathrm{Ave}\Big(\mathtt{y}^{(i)} | \mathtt{x}^{(i)} \in N_k(\mathbf{x})\Big) = \frac{1}{\big|N_k(\mathbf{x})\big|} \sum_{i:\mathtt{x}^{(i)} \in N_k(\mathbf{x})} \mathtt{y}^{(i)} \tag{9}$$

where "Ave" denotes the average, and $N_k(\mathbf{x})$ is the neighborhood consisting of the k points in the training set that are the closest to $\mathbf{x}$.

If we have a lot of data: $n, k \to \infty$ such that $k/n \to 0$, then: $\hat{h}_{\mathrm{NN}}(\mathbf{x}) \to \mathbb{E}\{\mathtt{y} | \mathtt{x} = \mathbf{x}\} = \hat{h}(\mathbf{x})$.

To see this intuitively, consider a special case and assume that we only have two features, $\mathtt{x} \in [0, 1] \times [0, 1]$ and that $p_{\mathtt{x}}(\mathbf{x})$ is uniform over the set $\mathcal{X} = [0, 1] \times [0, 1]$. Let $|\mathcal{X}|$ denote the area of $\mathcal{X}$, $|\mathcal{X}| = 1$ in this case. By definition,

$$\mathbb{E}_{\mathtt{y}|\mathtt{x}}\{\mathtt{y} | \mathtt{x} = \mathbf{x}\} = \int_y y \, p_{\mathtt{y}|\mathtt{x}}(y | \mathtt{x} = \mathbf{x}) dy. \tag{10}$$

Let $N(\mathbf{x})$ be the area around $\mathbf{x}$ that is roughly equal to $N_k(\mathbf{x})$. Assume for simplicity that the neighborhoods $N_k(\mathbf{x})$ are all the same size for all $\mathbf{x}$ (follows from uniformity assumption). Then, starting from (10):

$$\begin{aligned}
\mathbb{E}_{\mathrm{y}|\mathbb{x}}\{\mathrm{y}|\mathbb{x}=\mathbf{x}\} &= \frac{1}{|N(\mathbf{x})|}\int_{\mathbf{x}'\in N(\mathbf{x})}\int_y y\, p_{\mathrm{y}|\mathbb{x}}(y|\mathbb{x}=\mathbf{x})p_{\mathbb{x}}(\mathbf{x}')d\mathbf{x}'dy \\
&\approx \frac{1}{|N(\mathbf{x})|}\int_{\mathbf{x}'\in N(\mathbf{x})}\int_y y\, p_{\mathrm{y}|\mathbb{x}}(y|\mathbb{x}=\mathbf{x}')p_{\mathbb{x}}(\mathbf{x}')d\mathbf{x}'dy \\
&= \frac{1}{|N(\mathbf{x})|}\int_{\mathbf{x}'\in N(\mathbf{x})}\int_y y\, p_{\mathrm{y},\mathbb{x}}(\mathbf{x}',y)d\mathbf{x}'dy \\
&\approx \mathrm{Ave}\Big(\mathrm{y}^{(i)}|\mathbb{x}^{(i)}\in N_k(\mathbf{x})\Big).
\end{aligned}$$

Above, in the second equality we assumed that the function $p_{\mathrm{y}|\mathbb{x}}(y|\mathbb{x}=\mathbf{x}')$ is a smooth function of $\mathbf{x}'$ and does not change much in the neighborhood $\mathbf{x}'\in N(\mathbf{x})$.

Isn't this an ideal universal learning method?

In low dimensions this is true. However when dimensionality gets large we run into very serious problems. As we will see soon, the metric size of k-neighborhood gets very large as the dimension gets large. Therefore, nearest neighborhood becomes a very poor surrogate for conditioning.

2.2 Linear regression as an approximation to Bayes solution in the linear setting

How does the least squares algorithm we discussed before fit into the Bayesian theory? Suppose that from our prior knowledge of the problem domain we decide to make an additional assumption on the functional form of the regression function $h(\cdot)$. Specifically, assume that $h(\cdot)$ is linear: $h(\mathbf{x})=\mathbf{x}^{\mathsf{T}}\boldsymbol{\theta}$.

Plugging this into (2) and differentiating to find the minimum of $\mathrm{EPE}(\boldsymbol{\theta})$, we can solve for $\boldsymbol{\theta}$:

$$\boldsymbol{\theta}^* = \Big[\mathbb{E}\{\mathbb{x}\mathbb{x}^{\mathsf{T}}\}\Big]^{-1}\mathbb{E}\{\mathbb{x}\mathrm{y}\}. \tag{11}$$

Proof:

$$\hat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} \mathbb{E}\{[\mathrm{y}-\mathbb{x}^{\mathsf{T}}\boldsymbol{\theta}]^2\} \tag{12}$$

$$\Leftrightarrow \quad \frac{d}{d\hat{\boldsymbol{\theta}}}\int p(\mathbf{x},y)[y-\mathbf{x}^{\mathsf{T}}\hat{\boldsymbol{\theta}}]^2 d\mathbf{x}dy = 0 \tag{13}$$

$$\Leftrightarrow \quad \int p(\mathbf{x},y)2\mathbf{x}[y-\mathbf{x}^{\mathsf{T}}\hat{\boldsymbol{\theta}}]d\mathbf{x}dy = 0 \tag{14}$$

$$\Leftrightarrow \quad \int p(\mathbf{x},y)\mathbf{x}\mathbf{x}^{\mathsf{T}}\hat{\boldsymbol{\theta}}d\mathbf{x}dy = \int p(\mathbf{x},y)\mathbf{x}y d\mathbf{x}dy \tag{15}$$

$$\Leftrightarrow \quad \mathbb{E}[\mathbb{x}\mathbb{x}^{\mathsf{T}}]\hat{\boldsymbol{\theta}} = \mathbb{E}[\mathbf{x}y] \tag{16}$$

$$\Rightarrow \quad \hat{\boldsymbol{\theta}} = \Big[\mathbb{E}\{\mathbb{x}\mathbb{x}^{\mathsf{T}}\}\Big]^{-1}\mathbb{E}\{\mathbb{x}\mathrm{y}\}. \tag{17}$$

Let's compare this to least-squares solution we derived in the previous lecture:

$$\hat{\boldsymbol{\theta}}_{\mathrm{LS}} = (\mathbb{X}^{\mathsf{T}}\mathbb{X})^{-1}\mathbb{X}^{\mathsf{T}}\mathbb{y} \tag{18}$$

where $\mathbb{X}$ is the data matrix:

$$\mathbb{X} = \begin{bmatrix} - & (\mathbb{x}^{(1)})^\mathsf{T} & - \\ - & (\mathbb{x}^{(2)})^\mathsf{T} & - \\ - & \vdots & - \\ - & (\mathbb{x}^{(n)})^\mathsf{T} & - \end{bmatrix}.$$

To see the relationship between the two formulas, consider the jth column of $\mathbb{X}^\mathsf{T}\mathbb{X}$:

$$[\mathbb{X}^\mathsf{T}\mathbb{X}]_j = \sum_{i=1}^{n} \mathbb{x}^{(i)} \mathbb{x}_j^{(i)} \tag{19}$$

where $\mathbb{x}_j^{(i)}$ is the jth variable in the ith example in the training set and $\mathbb{x}^{(i)} = [\mathbb{x}_0^{(i)}, \ldots, \mathbb{x}_{p-1}^{(i)}]^\mathsf{T}$. By the law of large numbers this converges (when the number of training examples, n becomes large) to $n\,\mathbb{E}\{\mathbb{x}\mathbb{x}_j\}$, which is n times the jth column of $\mathbb{E}\{\mathbb{x}\mathbb{x}^\mathsf{T}\}$.

Similarly, note that

$$\mathbb{X}^\mathsf{T}\mathbb{y} = \sum_{i=1}^{n} \mathbb{x}^{(i)} \mathbb{y}^{(i)}. \tag{20}$$

By the law of large numbers this converges (when the number of training examples, n becomes large) to $n\,\mathbb{E}\{\mathbb{x}\mathbb{y}\}$. We conclude that

$$\hat{\theta}_{\text{LS}} \approx \left[n\,\mathbb{E}\{\mathbb{x}\mathbb{x}^\mathsf{T}\}\right]^{-1} n\,\mathbb{E}\{\mathbb{x}\mathbb{y}\} = \hat{\theta}. \tag{21}$$

2.3 Summary

Both k-nearest neighbors and least squares end up approximating conditional expectation by average. But they differ dramatically in terms of model assumptions:

- Least Squares assumes that $h(\mathbf{x})$ is well approximated by a global linear function.
- k-nearest neighbors assumes that $h(\mathbf{x})$ is well approximated by a local constant function.

3 Bayes optimal classifier: maximum conditional probability

So far in this lecture we have considered a regression problem, let's return to the classification problem from the previous lecture:

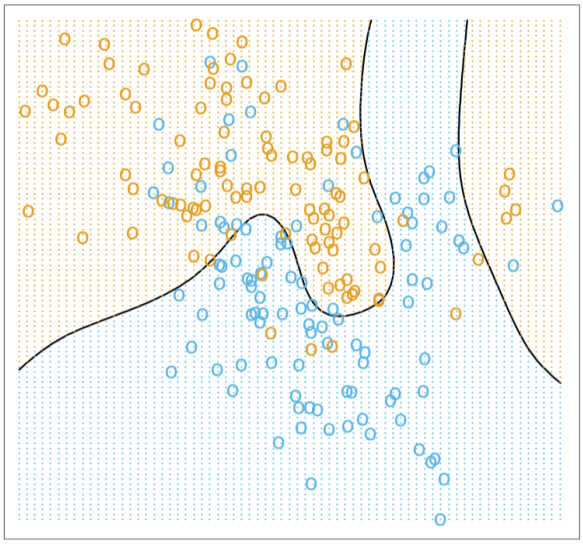


Figure 1: Bayes optimal classifier

We did not specify before how the data points in this example were generated. Now we reveal the secret:

- Generate 10 means $\mathbf{m}_k$ from $\mathcal{N}\left([0,1]^{\mathsf{T}}, \mathbf{I}\right)$ and label this class BLUE. Generated 10 means $\mathbf{m}_k$ from $\mathcal{N}\left([1,0]^{\mathsf{T}}, \mathbf{I}\right)$ and label this class ORANGE. These 20 points are fixed once and for all and are assumed nonrandom and known.
- For each class (BLUE and ORANGE), generate 100 observations as follows: pick an m_k at random with probability of $1/10$, then generate points according to $\mathcal{N}(\mathbf{m}_k, \mathbf{I}/5)$.

This is a Bayesian problem because we have specified the data-generating distribution precisely. Therefore, we can calculate the optimal solution by minimizing the expected prediction error:

$$EPE(h) = \mathbb{E}\left(L(\mathtt{y}, h(\mathtt{x}))\right) \tag{22}$$

where we use the misclassification loss function that is defined as follows:

$$L(\mathtt{y}, h(\mathtt{x})) = \begin{cases} 0, \text{ iff } h(\mathtt{x}) = \mathtt{y} \\ 1, \text{ iff } h(\mathtt{x}) \neq \mathtt{y}. \end{cases} \tag{23}$$

Similarly to the calculation we did for regression,

$$h(\mathbf{x}) = \arg\min_{c \in \{0,1\}} \mathbb{E}_{\mathrm{y}|\mathbf{x}}(L(\mathrm{y}, c)|\mathbf{x} = \mathbf{x}) \tag{24}$$

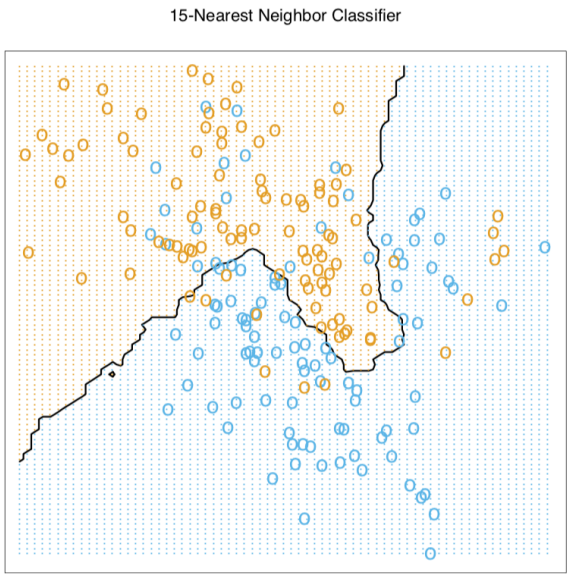
$$= \arg\min_{c \in \{0,1\}} \mathbb{P}(\mathrm{y} = 0|\mathbf{x} = \mathbf{x}) \cdot L(0, c) + \mathbb{P}(\mathrm{y} = 1|\mathbf{x} = \mathbf{x}) \cdot L(1, c) \tag{25}$$

$$= \arg\min_{c \in \{0,1\}} \mathbb{P}(\mathrm{y} \neq c|\mathbf{x} = \mathbf{x}) \tag{26}$$

$$= \arg\min_{c \in \{0,1\}} 1 - \mathbb{P}(\mathrm{y} = c|\mathbf{x} = \mathbf{x}) \tag{27}$$

$$= \arg\max_{c \in \{0,1\}} \mathbb{P}(\mathrm{y} = c|\mathbf{x} = \mathbf{x}). \tag{28}$$

Therefore, the optimal algorithm is simply to choose the class that has the largest conditional probability given the data. This is called the *Bayes rule* for classification. The Bayes-optimal decision boundary is shown in Figure 1. It is instructive to compare this decision boundary to the one produced by the 15-nearest neighbors classifier:



We observe that the decision boundaries produced by the two algorithms are very similar!