Lecture 2

IDS575: Statistical Models and Methods
Theja Tulabandhula

Notes derived from the book titled "Elements of Statistical Learning [2nd edition] (Sections 2.4-2.9, 3.1-3.2)

1 Statistical Decision Theory

We will now formally make the variables into random variables! Allowing us to bring in statistics and probability into the mix.

Let $Pr(X, Y)$ be a joint distribution. We want a function $f(X)$ for predicting Y, and its quality is measured using a *loss function* $L(Y, f(X))$. The loss function allows us to define what we mean by a 'best' model.

Example 1. Loss function: $(Y - f(X))^2$. □

So from the informal least squares objective, we can generalize to the following objective, called the *expected (squared) prediction error* (**EPE**):

$$\begin{aligned} EPE(f) &= E(Y - f(X))^2 \\ &= E_X E_{Y|X}([Y - f(X)]^2|X). \end{aligned}$$

We want to minimize this to get an f. One way to thin about it is to observe that the function value $f(x)$ at each point x is unrelated to other points. So you have one optimization problem per x which looks like:

$$\min_z E_{Y|X}([Y - z]^2|X = x).$$

This is minimized at $z = E(Y|X = x)$. Thus, the function defined by $f(x) = E(Y|X = x)$ is the optimal solution. It is called *the regression function.*

Lets revisit k-nearest-neighbor and linear modeling: they are in fact trying to estimate the regression function $E(Y|X = x)$!

1.1 Reinterpreting the Nearest-neighbor Method and the Linear Model

Nearest-neighbor: If we want to estimate $E(Y|X = x)$ at a point x, a natural thing to do is: average all y_i such that $x_i = x$.

Further relax this by taking those y_i into account for which $x_i \sim x$ (i.e., in the neighborhood).

For large number of points, we can expect that the neighbors are close to x so this approximation make sense.

The intuition here is that: (a) large N will get you many neighbors, and large k will get you stable estimation.

In fact, theoreticians have shown that for many different $Pr(X, Y)$, as long as N and k go to ∞, and k/N goes to 0, $\widehat{\mathrm{f}}(x) \to E(Y|X = x)$.

Is this the end of the story? Obviously no.

- The catch is, this is an *asymptotic* result.
- There is also another issue which will make Nearest-neighbor methods ineffective, which is called the ***curse of dimensionality***.

Linear model: What if we assume that the regression function $E(Y|X = x)$ is of the form $x^T\beta$? We are assuming something about the data then. If we make this assumption, it turns out that:

$$\beta = [E(XX^T)]^{-1}E(XY).$$

We have a single parameter β that does not depend on a specific value of x, it depends on the entire distribution of X (through the expectation).

If you compare this to the least squares solution, you will notice that the least squares solution is the empirical average version of the above.

Lets summarize Nearest-neighbor method and the linear model via least squares, now that we know about the expected prediction error EPE and $E(Y|X = x)$.

- Both methods do averaging
- Nearest neighbor assumes $E(Y|X)$ is locally constant.
- Least squares assumes a (globally) linear function for $E(Y|X)$.

Note 1. Instead of assuming $f(X) := E[Y|X] = X^T\beta$, one could assume $f(X) = \sum_{i=1}^{p} f_j(X_j)$. Can you see why this is more general than the linear model?

Note 2. What if we change the objective to $E|Y - f(X)|$? It turns out that the best fit is $f(x) = \text{median}(Y|X = x)$. It is not clear why one would choose one objective over the other a priori[1].

1.2 EPE for Classification

If we have categorical variable G (with K values) instead of Y then we can define an 'EPE' objective using a $K \times K$ matrix.

Example 2. This is the reasoning behind the zero-one loss function: a loss of 0 when correct prediction happens, and a loss of value 1 when incorrect prediction happens. □

The expected prediction error (EPE) in this case is:

$$EPE = E_X \sum_{k=1}^{K} L(G = k, \widehat{\mathrm{G}}(X)) Pr(G = k|X),$$

where $\widehat{\mathrm{G}}(x)$ is the classifier and the output variable takes K values between $1, ..., K$.

If we repeat our calculations as before and *use the zero-one loss*, we get the classifier:

$$\widehat{\mathrm{G}}(x) = \max_{k \in \{1,...,K\}} Pr(G = k|X = x).$$

Note 3. This is a very intuitive classifier and is known as the *Bayes classifier*. See Figure 1 for how it looks like.

Note 4. A nearest-neighbor classifier approximates this as well because it is essentially taking a 'majority vote'.

Note 5. As mentioned before, as a heuristic, one can convert this classification setting to a regression setting ($G \to Y$) and then do appropriate thresholding (say 0.5 for binary).

2 Curse of Dimensionality

Because it looks like no assumptions are needed for the nearest-neighbor methods, one should always try them first. Especially when there is large amount of data.

Not so correct intuition: With large data, one could approximate the conditional expectation very well because you can find a lot of neighbors for a point x.

The reason it breaks down is termed as the *curse of dimensionality*. How does it impact statistical modeling? Say you have three inputs and $N = 10000$, then the nearest-neighbor method may work well. But if you have 3000 inputs (because you collected more measurements), nearest-neighbor will most likely not work from a statistical modeling point of view!

[1] Perhaps, computational considerations may influence this choice.

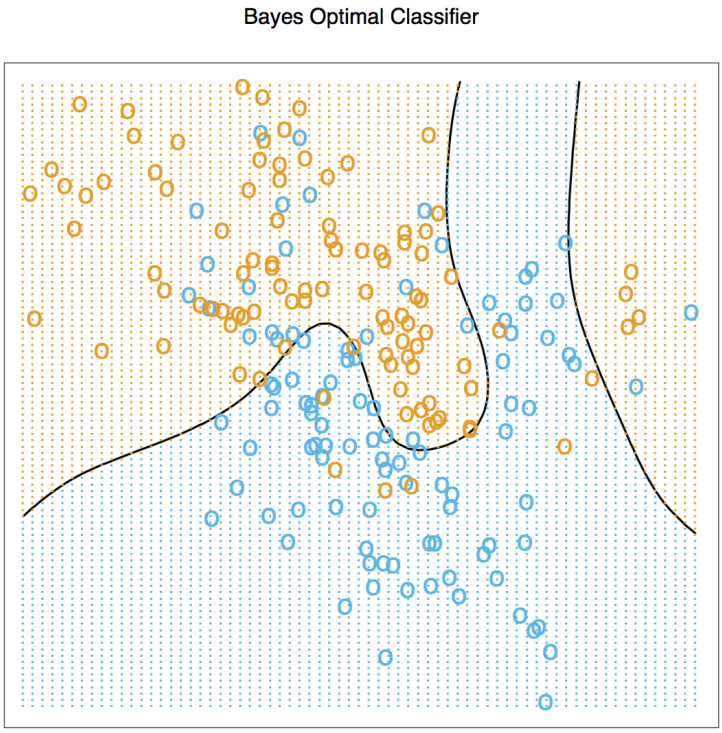

Figure 1: Bayes decision boundary.

Note 6. You may be worried about computing nearest neighbors in a decent amount of time as well. There are some solutions to the computational problem (such as approximately finding the nearest neighbors). Spotify has an open source implementation called Annoy that tries to address the computational issue (i.e., avoid linear time). In their own words: ... *"We use it at Spotify for music recommendations. After running matrix factorization algorithms, every user/item can be represented as a vector in f-dimensional space. This library helps us search for similar users/items. We have many millions of tracks in a high-dimensional space, so memory usage is a prime concern."*

Example 3. If you want to find a neighborhood-cube of side length l that covers r fraction (say 10%) of the volume of a unit hypercube in a p dimensional space, then the side length of such a cube is $(r)^{1/p}$ (say $0.1^{1/p}$). Figure 2 shows how long this is for even $p = 10$. Thus to capture a fixed percentage of data (if it is uniformly spread), you will need almost the full range of each variable coordinate! This is no longer local.

The consequence of all this is that doing statistical modeling for high dimensions is very different from low dimensions and we somehow need exponentially more observations in the training set.

Thus, nearest neighbor methods will fail because:

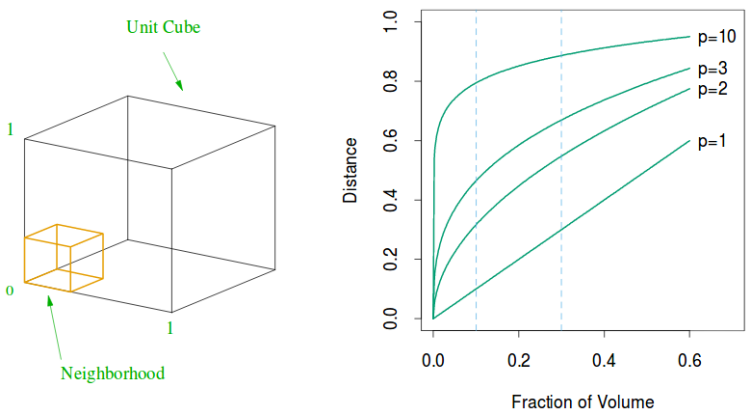

Figure 2: Large neighborhoods for the same fraction of volume.

- neighbors will not be close to the target point for which we are making a prediction, and
- if there is special structure in the data that you are aware of, significantly better prediction error can be obtained.

This is one of the key reasons why we want to develop different statistical modeling techniques!

Take for instance the linear model: if the strong assumption (conditional expectation is linear) holds, then the prediction error does not scale too badly compared to the nearest-neighbor methods. Figure 3 illustrate the relative errors between 1-nearest neighbor and the linear model for two specific datasets, showing that the latter is better. Note that such a trend can be easily reversed if we change the datasets.

3 Supervised Learning

3.1 Beyond Expected (squared) Prediction Error

We have only looked at k-nearest neighbor, linear models and squared loss function, But we can think of any function, parameterized by a parameter (say θ), and minimize the residual sum-of-squares (RSS, which is the empirical version of EPE):

$$RSS(\theta) = \sum_{i=1}^{N} (y_i - f_\theta(x_i))^2.$$

For $f_\theta(X) = X^T\theta$, we get a closed form expression. Otherwise, we may have to numerically optimize.

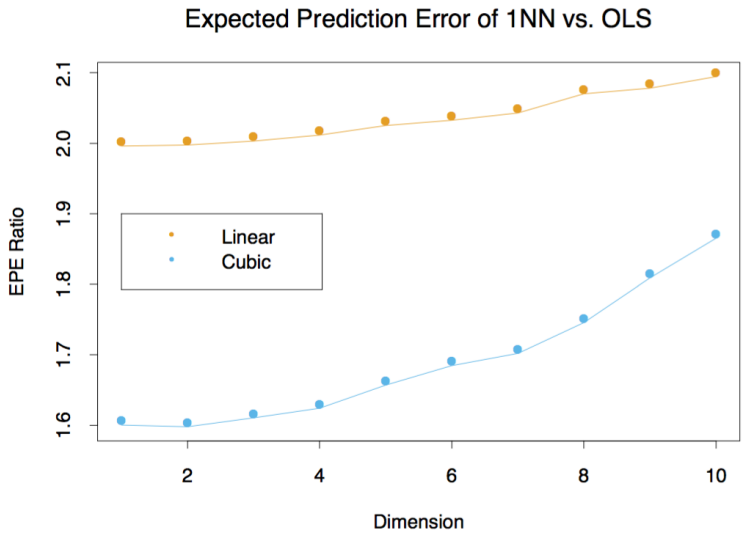

Figure 3: Linear model ($Y =$) doing better than 1-nearest-neighbor for two specific datasets (orange: $Y = X + \epsilon$, blue: $Y = \frac{1}{2}(X+1)^3$, X is 1-dimensional and $\epsilon \sim N(0,1)$). The y-axis is the EPE ratio of 1-neaerst-neighbor and linear modeling.

While least squares objective is great, there is a more general principle of estimation called *Maximum Likelihood Estimation* (**MLE**). Say random variable $Z \sim Pr_\theta(Z)$, where the density is indexed by parameter θ, then the log-probability of observing $z_1, ..., z_N$ is:

$$L(\theta) = \sum_{i=1}^{N} \log Pr_\theta(z_i).$$

The principle of MLE says that the most reasonable value for θ is that one, for which the probability of the observed sample is the largest.

Example 4. If $Y = f_\theta(X) + \epsilon$, $\epsilon \sim N(0, \sigma^2)$, then least squares is *equivalent* to MLE using $P(Y|X, \theta) = N(f_\theta(X), \sigma^2)$. The conditional log-likelihood of data is

$$L(\theta) \propto \sum_{i=1}^{N} (y_i - f_\theta(x_i))^2,$$

which is the same as $\text{RSS}(\theta)$.

Example 5. Consider qualitative output G. Let the model be $Pr(G = k|X = x) = p_{k,\theta}(x)$ for $1 \le k \le K$. Then the log-likelihood (also known as cross-entropy[2]) is $L(\theta) = \sum_{i=1}^{N} \log p_{g_i,\theta}(x_i)$.

[2]You may come across this term when you look at deep learning and neural network classifiers.

3.2 Different Model Families

For a fixed N, if we optimize $RSS(f)$ over all functions, we will have infinitely many solutions!

So we should restrict the search space. Such restrictions are based on additional knowledge and can be imposed using:

1. constraints on θ, or
2. implicitly though the learning method.

Example 6. For example, local linear fits in large neighborhoods is almost a globally linear model and is quite restrictive.

Note 7. Generally, any method that overcomes curse of dimensionality has an associated metric for defining the size of neighborhoods that are not small.

Lets list some methods here:

- Regularlized models: Here, the search space is controlled by defining Penalized RSS (PRSS): $PRSS(f, \lambda) = RSS(f) + \lambda J(f)$. These are also called penalty methods. And are equivalent to having a log-prior from a Bayesian point of view.
- Kernel methods: These explicitly specify the local neighborhood using a kernel function $K_\lambda(x_0, x)$.
  - $K_\lambda(x_0, x)$ assigns a weight to point x in a region around x_0.
  - Example: $K_\lambda(x_0, x) = \frac{1}{\lambda}\exp(-\frac{\|x-x_0\|_2^2}{2\lambda})$ assigns weights that die exponentially with squared Euclidean distance.
  - Given a $K_\lambda(x_0, x)$, an example model is:

$$\widehat{f}(x_0) = \frac{1}{\sum_{i=1}^N K_\lambda(x_0, x_i)} \sum_{i=1}^N K_\lambda(x_0, x_i) y_i,$$

    which resembles a weighted average!
  - $K_k(x_0, x) = 1[\|x - x_0\|_2 \leq \|x_{(k)} - x_0\|_2]$ gives us the k-nearest-neighbor method. Here, $x_{(k)}$ is the k^{th} nearest point to x_0.
- Dictionary methods: These are linear in the coefficients, but arbitrary in the inputs. A function in this family would be $f_\theta(x) = \sum_{m=1}^M \theta_m h_m(x)$. For instance, $h_m(x) = \exp(-\frac{1}{2\lambda_m}\|x - \mu_m\|_2^2)$. The are called dictionary methods, because we can choose from a set of candidate functions to define each $h_m()$.

4 Linear Regression

Linear regression: just means that $E(Y|X)$ is linear in the inputs $X_1, ..., X_p$.

Note 8. Linear methods can be applied to transformed inputs, considerably expanding their scope. This generalization leads to what are called *basis-function* methods.

Model:

$$Y = \sum_{j=1}^{p} X_j \beta_j$$

Here, X_js can be: (a) quantitative, (b) functions of raw data (e.g., log(), $\sqrt{}$ etc.), (c) or even encode ***interactions between*** variables (e.g., $X_3 = X_1 \cdot X_2$).

Note 9. X_j can also be categorical. Say it has K levels, then we can create a 1-hot encoding to get a group of input variables X_k, $k = 1, ..., K$. Here $X_k = 1[X_j = k]$.

Irrespective of how X_js came about, the model is linear in the parameter β.

4.1 Least Squares

Recall that $RSS(\beta) = \sum_{i=1}^{N}(y_i - \sum_{j=1}^{p} x_{ij}\beta_j)^2$. The geometry of least squares is illustrated in Figure 4.

Lets derive the closed form expression for $\widehat{\beta}$ that we saw before.

$$RSS(\beta) = (\mathbf{y} - \mathbf{X}\,\beta)^T(\mathbf{y} - \mathbf{X}\,\beta)$$

Here, $\mathbf{X}$ is $N \times p$. This is a quadratic function of β. If we differentiate with respect to β, we get:

$$\frac{\partial RSS}{\partial \beta} = -2\,\mathbf{X}^T(\mathbf{y} - \mathbf{X}\,\beta),$$

and

$$\frac{\partial^2 RSS}{\partial \beta \partial \beta^T} = 2\,\mathbf{X}^T\,\mathbf{X}\,.$$

If $\mathbf{X}$ has full column rank, then $\mathbf{X}^T\,\mathbf{X}$ is positive definite. Setting the first derivative equal to zero, i.e.,

$$\mathbf{X}^T(\mathbf{y} - \mathbf{X}\,\beta) = 0, \text{we get}$$
$$\Rightarrow \widehat{\beta} = (\mathbf{X}^T\,\mathbf{X})^{-1}\,\mathbf{X}^T\,\mathbf{y}\,.$$

Note 10. Does it happen that the rank of $\mathbf{X} = [\mathbf{x}_1 \ldots \mathbf{x}_p]$ is not p? The answer is yes. For instance, if $\mathbf{x}_2 = 4\,\mathbf{x}_1$. Another example: when we transform categorical variable X_j into 1-hot encoding.

Note 11. When $\mathbf{X}$ is not full rank, then $\widehat{\beta}$ is not unique! One fix is to drop redundant columns.

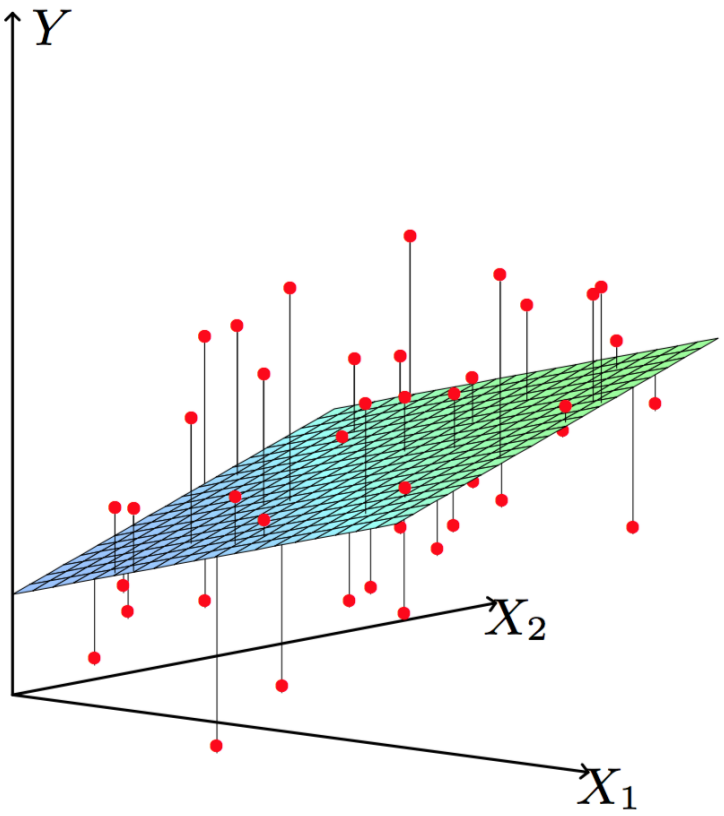

Figure 4: We seek a linear function of X that minimizes $RSS(\beta)$.

5 Summary

We learned the following things:

- Introduced probability while defining supervised learning problems.
- Curse of dimensionality motivated us to look for other statistical models (we will see them in future lectures!).
- Linear Regression: interaction terms, categorical variables.

In the next lecture, we will discuss subset selection and LASSO methods in regression. And move to the nuances of classification.

A Sample Exam Questions

- What is the regression function in the context of minimizing EPE?
- In what ways does k-nearest neighbor get affected by the curse of dimensionality?

- What is the equivalent maximum likelihood setup that justifies minimizing RSS?