Lecture 2

- Nearest-Neighbor Methods
- Linear Regression Model
- Curse of Dimensionality

Last time we have shown that in the regression settings ($X \in \mathbb{R}^p$, $Y \in \mathbb{R}$), the optimal (in the MSE sense) prediction of the output to input $X = x$ is given by the regression function: $f(x) = \mathbb{E}[Y|X=x]$.

This function is unknown and our goal is to estimate it.

Nearest-Neighbor Methods provide the most straightforward way to estimate $f(x)$.

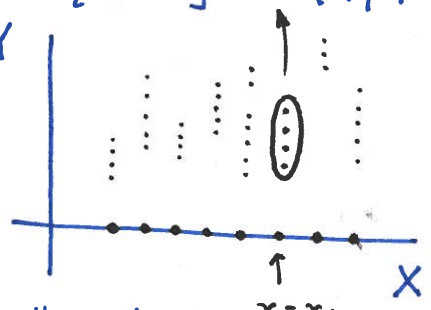
The main idea is to estimate $\mathbb{E}[Y|X=x]$ "nonparametrically" directly from the data.

Let $\mathcal{D} = \{(x_1, y_1), \dots, (x_N, y_N)\}$ be the training data.

Then we can estimate the expectation by the empirical average: $\mathbb{E}[Y|X=x] \approx \text{Ave}(y_i | x_i = x)$

The problem with this naive approach is that typically there is at most one observation at any $x = x_i$, that is the set $\{y_i | x_i = x\}$ consists of 1 point or empty.

This difficulty can be resolved by relaxing condition $x_i = x$



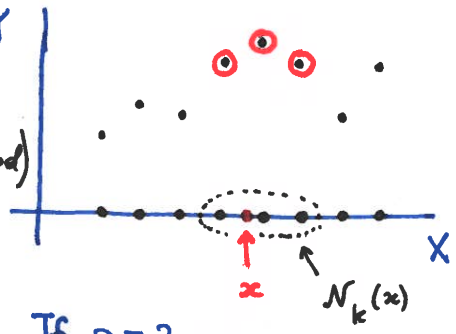
Let's replace $x_i = x$ with $x_i \in N_k(x)$, where $N_k(x)$ is the set of the k closest inputs x_i in the data

neighborhood of x

This leads to the following approximation of the regression function:

(#) $$f(x) \approx \hat{f}(x) = \frac{1}{k} \sum_{i : x_i \in N_k(x)} y_i$$

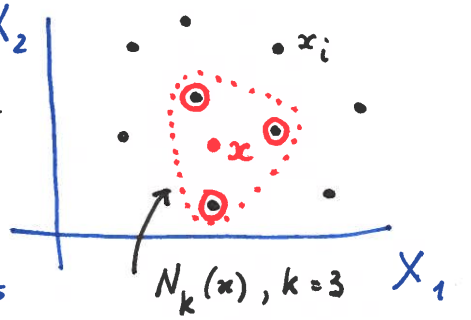
k-nearest neighbor method (k-NN method)



Two approximations are happening here:

1. Expectation is approximated by the empirical average.
   If $N, k \to \infty$, then approximation becomes better thanks to the LLN (law of large numbers)
2. Condition $x_i = x$ is relaxed by $x_i \in N_k(x)$.
   As $N \to \infty$, points in $N_k(x)$ become closer to x, and the corresponding approximation becomes better.

If $p = 2$



Important Fact: Under certain mild regularity conditions on the joint distribution of X and Y, it can be shown that

(*) $$\hat{f}(x) \to f(x) = \mathbb{E}[Y|X=x]$$ as $N, k \to \infty$ such that $\frac{k}{N} \to 0$.

Devroy at al (1996) "A probabilistic theory of pattern recognition."

From a theoretical point of view, the k-NN method looks very good and it seems we can use it as a universal solution to the regression problem.

But from a practical point of view, it has one drawback: the convergence (*) is a limiting result that holds when $N, k \to \infty$ (and $k/N \to 0$), but for finite N and k the approximation $f(x) \approx \hat{f}(x)$ in (#) may not be (and often is not) accurate.

This is especially the case if $p = \dim X$ is large: the k-NN method suffers from the curse of dimensionality.

- the convergence (*) still holds, but the rate of converges ↓ as p ↑.
- refers to phenomena that arise when analyzing data in high-dimensional spaces that do not occure in low-dim. spaces.

Let's consider several manifestations of the curse of dimensionality.

① The k-NN method is a local method: $f(x)$ is approximated based on the information in $N_k(x)$ that contains a fraction $\alpha = k/N$ of training sample.

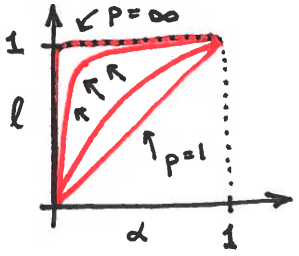
But in high dimensions the neighborhood $N_k(x)$ is not really "local".

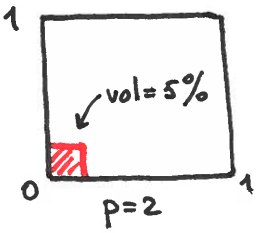
To illustrate this, consider inputs $x_1, \dots x_N$ uniformly distributed in $C_p = [0,1]^p$.

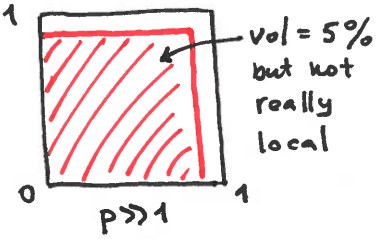
- p-dimensional unit hypercube.

To capture a fraction α of the sample by a hypercubical neighborhood $C_p(\ell) = [0,\ell]^p$, we need to choose ℓ such that $\text{vol}(C_p(\ell)) = \alpha$.

$$\Longrightarrow \ell^p = \alpha \iff \ell = \alpha^{1/p} \quad (\to 1 \text{ as } p \to \infty)$$





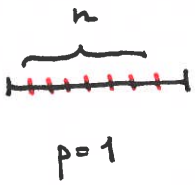


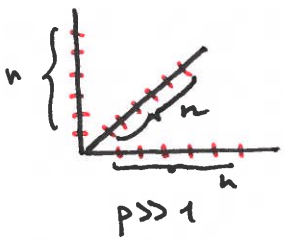
Remark: For example, if $p = 20$ and $\alpha = 0.01$, then $\ell \approx 0.8$. That is to capture 1% of the data to form a local average, we must cover about 80% of the range of each input variable in 20-dim space.

② Realistic training samples $x_1 \dots x_N$ sparsely populate the high-dimensional input spaces.

For example, if n measurements of $X_1 \in [0,1]$ represent a dense sample for a single input problem, then to get the same "resolution" (sampling density) of C_p, we need to use the sample size $N = \underbrace{n \times \dots \times n}_{p \text{ times}} = n^p$, which is not feasible.

- for instance if $n = 100$ and $p = 10$ ⇓ $N = 100^{10}$

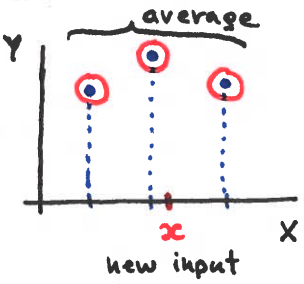




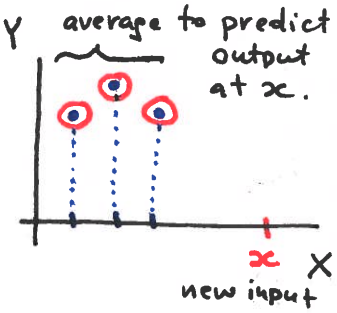
The problem is that the volume increases exponentially with adding extra dimensions.

③ An important consequence of the sparse sampling in high dimensions is that all sample points are close to the "boundary" of the sample. This is a problem for the k-NN method: instead of interpolating between nearest neighbors $x_i \in N_k(x)$ (as (#) suggests), we will have to extrapolate from them.

Schematically: instead of nice picture suggested by our low-dim. intuition

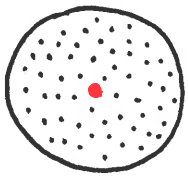


in high-dim, we will have the following picture



To illustrate this, consider N inputs uniformly distributed in B_p ← p-dimensional unit ball centered at the origin.

$X_1, \dots, X_N \sim \mathcal{U}(B_p)$



Low-dimensional intuition suggests that the we have:
Suppose we want to use a nearest-neighbor method to estimate the output at the origin.

Let D be the distance from the origin to the nearest neighbor (the closest X_i)

$\Rightarrow D = \min\{\|X_1\|, \dots, \|X_N\|\}$

Norms $\|X_1\|, \dots, \|X_N\|$ are i.i.d. Let's find their CDF $F_{\|X\|}(d)$, $X \sim \mathcal{U}(B_p)$

$$F_{\|X\|}(d) = \mathbb{P}(\|X\| \le d) = \frac{\text{vol}(B_p(d))}{\text{vol}(B_p)} = \frac{c \cdot d^p}{c \cdot 1^p} = d^p, \quad d \in [0,1].$$

$\text{vol}(B_p(d))$: p-dimensional ball of radius d centered at the origin.

Let's find the CDF of D:

$$F_D(d) = \mathbb{P}(D \le d) = 1 - \mathbb{P}(D > d)$$

$$= 1 - \mathbb{P}(\|X_1\| > d, \dots, \|X_N\| > d) = 1 - \prod_{i=1}^{N} \mathbb{P}(\|X_i\| > d)$$

$$= 1 - \prod_{i=1}^{N} \underbrace{(1 - \mathbb{P}(\|X_i\| \le d))}_{F_{\|X\|}(d)} = 1 - (1 - d^p)^N.$$

Remark: C is a constant that depends on p:

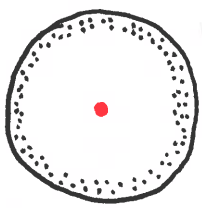
$$C = \frac{\pi^{p/2}}{\Gamma(p/2 + 1)}$$

Γ: Euler's gamma function

Let's use the median $\tilde{d}$ as a proxy for a typical value of D.

$$F_D(\tilde{d}) = \frac{1}{2} \Rightarrow \tilde{d} = \left(1 - \left(\frac{1}{2}\right)^{1/N}\right)^{1/p} \to 1 \text{ as } p \to \infty.$$

$\Rightarrow$ in high dimensions the picture looks like this:



For example, if $p = 20$, $N = 10^3 \Rightarrow \tilde{d} \approx 0.7$

$p = 50$, $N = 10^3 \Rightarrow \tilde{d} \approx 0.86$

Remark: It is possible to derive an expression for $\bar{d} = \mathbb{E}[D]$, but it is more complicated.

$\Rightarrow$ using k-NN at the center leads to extrapolation

So, if the sample size N is not extremely large and/or the dimension p of the input space is high, then the nearest neighbors $N_k(x)$ may not be close to the target input x, and this can result in large prediction errors.

This motivates us to look for alternative methods. Let's consider a completely different strategy: instead of estimating the regression function $f(x)$ directly from the data, let's model it, that is impose some structural assumptions on $f(x)$. ↷2

Suppose $f(X)$ is linear: $\boxed{f(X) = \beta_0 + \beta_1 X_1 + \dots + \beta_p X_p}$

Let $\beta = \begin{bmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_p \end{bmatrix}$ and $X = \begin{bmatrix} 1 \\ X_1 \\ \vdots \\ X_p \end{bmatrix}$ $\Rightarrow$ $\boxed{f(X) = X^T\beta}$ $\quad X \in \mathbb{R}^{p+1}$, $\beta \in \mathbb{R}^{p+1}$

Remark The linear model has been a mainstay of statistics for the past 50 years and remains one of the most important tools.

Remark The term β_0 is called bias in ML.

The mean squared error (the expected prediction error under the squared error loss function) is then:

$$MSE(f) = \mathbb{E}\left[(Y - \hat{Y})^2\right] = \mathbb{E}\left[(Y - X^T\beta)^2\right] = MSE(\beta) \leftarrow \text{function of } \beta.$$

To find the regression coefficients β that minimize MSE, we need to solve $\nabla_\beta MSE = 0$ for β.

$$\nabla_\beta MSE(\beta) = \nabla_\beta \mathbb{E}\left[(Y - X^T\beta)^2\right] = \mathbb{E}\left[\nabla_\beta (Y - X^T\beta)^2\right] = \mathbb{E}\left[2(Y - X^T\beta)\nabla_\beta (Y - X^T\beta)\right]$$

↑ integration w.r.t. X and Y commutes with differentiation w.r.t. β.

$$= -2\,\mathbb{E}\left[(Y - X^T\beta)\underbrace{\nabla_\beta (X^T\beta)}_{= X}\right] = -2\,\mathbb{E}\left[(Y - X^T\beta)X\right] = 0.$$

$= X$, since the gradient is a vector (and vectors are columns)

This leads to: $\mathbb{E}[YX] - \mathbb{E}[\underbrace{X^T\beta}_{\text{scalar}}\, X] = 0 \Rightarrow \mathbb{E}[XX^T\beta] = \mathbb{E}[YX]$.

Finally: $\mathbb{E}[XX^T]\beta = \mathbb{E}[YX] \Rightarrow \boxed{\beta = \mathbb{E}[XX^T]^{-1} \cdot \mathbb{E}[XY]}$ (!) ↷1

Equation (!) gives theoretically optimal value of the regression parameters.

Let's now estimate β using the training data $(x_1, y_1), \dots, (x_N, y_N)$.

Let's estimate the expected values in (!) by the corresponding empirical averages.

- $\mathbb{E}[XX^T] \approx \frac{1}{N}\sum_{i=1}^{N} x_i x_i^T = \frac{1}{N}\left(\overbrace{x_1 x_1^T}^{(p+1)\times(p+1)} + \dots + \overbrace{x_N x_N^T}^{(p+1)\times(p+1)}\right)$

$$= \frac{1}{N}\begin{bmatrix} | & & | \\ x_1 & \dots & x_N \\ | & & | \end{bmatrix} \cdot \left(\begin{bmatrix} - x_1^T - \\ 0 \end{bmatrix} + \dots + \begin{bmatrix} 0 \\ - x_N^T - \end{bmatrix} \right)$$

$$= \frac{1}{N}\begin{bmatrix} x_1 & \dots & x_N \end{bmatrix} \cdot \begin{bmatrix} x_1^T \\ \vdots \\ x_N^T \end{bmatrix} = \frac{1}{N} X^T \cdot X$$

$N \times (p+1)$

Remark: $XY = YX$ since $Y \in \mathbb{R}$

Remark: Since $X = \begin{bmatrix} 1 \\ X_1 \\ \vdots \\ X_p \end{bmatrix}$ $\in \mathbb{R}^{p+1}$, $x_i = \begin{bmatrix} 1 \\ x_{i1} \\ \vdots \\ x_{ip} \end{bmatrix} \in \mathbb{R}^{p+1}$

- $\mathbb{E}[XY] \approx \frac{1}{N}\sum_{i=1}^{N} x_i y_i = \frac{1}{N}\left(\begin{matrix}|\\ x_1\\ |\end{matrix} y_1 + \dots + \begin{matrix}|\\ x_N\\ |\end{matrix} y_N\right) = \frac{1}{N}\underbrace{\begin{bmatrix} | & & | \\ x_1 & \dots & x_N \\ | & & | \end{bmatrix}}_{X^T}\underbrace{\begin{bmatrix} y_1 \\ \vdots \\ y_N \end{bmatrix}}_{y} = \frac{1}{N} X^T y$

Thus, β can be estimated:

$$\beta \approx \hat{\beta} = \left(\frac{1}{N}X^TX\right)^{-1}\cdot\frac{1}{N}X^Ty = (X^TX)^{-1}X^Ty \Rightarrow \boxed{\hat{\beta} = (X^TX)^{-1}X^Ty} \quad (*)$$

So, if we assume that the relationship between the output Y and input X is approximately linear, $Y \approx f(X) = X^T\beta$, then we can estimate β using $(*)$ and use $\underbrace{\hat{Y} = \hat{f}(X) = X^T\hat{\beta}}$ for prediction. This is the linear regression method.

the best we can do under linear assumption.

If the linearity assumption is wrong, $Y \not\approx X^T\beta$, then the linear regression prediction $\hat{Y} = X^T\hat{\beta} \not\approx Y$ will obviously be not accurate.

Suppose that the assumption holds approximately: $Y = X^T\beta + \varepsilon$, where $\varepsilon \sim N(0, \sigma^2)$ (Gaussian noise)

Q: How accurate the linear regression in this case?
Does the accuracy suffer from the curse of dimensionality?

To answer these questions, let's derive the expected MSE of the method.

Let $X \in \mathbb{R}^{p+1}$ be a fixed input. The MSE at this X is

$$MSE(X) = \mathbb{E}[(Y-\hat{Y})^2] = \mathbb{E}[(X^T\beta + \varepsilon - X^T\hat{\beta})^2] \leftarrow \mathbb{E} \text{ wrt } \varepsilon \text{ and training data } \{X, y\}$$

- $\hat{\beta} = (X^TX)^{-1}X^Ty = (X^TX)^{-1}X^T(X\beta + e) = \beta + (X^TX)^{-1}X^Te$
- $y = \begin{bmatrix} y_1 \\ \vdots \\ y_N \end{bmatrix} = \begin{matrix} x_1^T\beta + \varepsilon_1 \\ \\ x_N^T\beta + \varepsilon_N \end{matrix} \Rightarrow y = \begin{bmatrix} x_1^T \\ \vdots \\ x_N^T \end{bmatrix}\beta + \underbrace{\begin{bmatrix} \varepsilon_1 \\ \vdots \\ \varepsilon_N \end{bmatrix}}_{e} = X\beta + e$

$1\times(p+1)$, $(p+1)\times(p+1)$, $(p+1)\times N$

$$\Rightarrow MSE(X) = \mathbb{E}\left[\left(X^T\beta + \varepsilon - X^T\beta - X^T(X^TX)^{-1}X^Te\right)^2\right] = \mathbb{E}\left[\left(\varepsilon - X^T(X^TX)^{-1}X^Te\right)^2\right]$$

$$= \mathbb{E}\left[\varepsilon^2 - 2\underbrace{\varepsilon\cdot X^T(X^TX)^{-1}X^Te}_{\text{independent, } \mathbb{E}[\varepsilon]=0} + \left(X^T(X^TX)^{-1}X^Te\right)^2\right] = \sigma^2 + \underbrace{\mathbb{E}\left[\left(X^T(X^TX)^{-1}X^Te\right)^2\right]}_{\text{let's compute this expectation } \mathbb{E}}$$

$$\mathbb{E} = \mathbb{E}\left[X^T(X^TX)^{-1}X^Te\cdot\underbrace{X^T(X^TX)^{-1}X^Te}_{\text{scalar, we can transpose it}}\right] = \mathbb{E}\left[X^T\underbrace{(X^TX)^{-1}X^Tee^TX(X^TX)^{-1}}_{\text{condition on } X}X\right] = \begin{matrix}\uparrow \text{fixed} \\ \\ \uparrow \text{fixed}\end{matrix}$$

$$= X^T\mathbb{E}\left[\mathbb{E}\left[(X^TX)^{-1}X^Tee^TX(X^TX)^{-1}\,|\,X\right]\right]X$$

$$= X^T\mathbb{E}\left[(X^TX)^{-1}X^T\underbrace{\mathbb{E}[ee^T|X]}_{=\mathbb{E}[ee^T]=\sigma^2I_N}X(X^TX)^{-1}\right]X = X^T\mathbb{E}\left[\sigma^2\overbrace{(X^TX)^{-1}X^TX}^{I_{p+1}}(X^TX)^{-1}\right]X$$

$\uparrow$ since e and X are independent

$= \mathbb{E}[ee^T] = \sigma^2 I_N$, where I_N is $N\times N$ identity matrix.

So, $\mathbb{E} = \sigma^2 X^T \mathbb{E}\left[(X^TX)^{-1}\right] X$

$$\Rightarrow \boxed{MSE(X) = \sigma^2 + \sigma^2 X^T \mathbb{E}\left[(X^TX)^{-1}\right] X}$$

At the bottom of page 9, we saw that $\frac{1}{N} X^TX \approx \mathbb{E}[XX^T]$.

Remark: By the law of large numbers $\frac{1}{N} X^TX \longrightarrow \mathbb{E}[XX^T]$ as $N \to \infty$.

Therefore, $(X^TX)^{-1} \approx \frac{1}{N} (\mathbb{E}[XX^T])^{-1}$

Remark: If $\mathbb{E}[X] = 0$, then $Cov(X) \overset{def}{=} \mathbb{E}\left[(X - \mathbb{E}[X])(X - \mathbb{E}[X])^T\right] = \mathbb{E}[XX^T]$

And, therefore, $(X^TX)^{-1} \approx \frac{1}{N} Cov(X)^{-1}$.

$$\Rightarrow \boxed{MSE(X) \approx \sigma^2 + \sigma^2 X^T (\mathbb{E}[XX^T])^{-1} X / N}$$

Now we can find the expected MSE:

useful trick

trace operator $tr\, A = \sum_{i=1}^{n} A_{ii}$

$$\mathbb{E}[MSE(X)] \approx \sigma^2 + \frac{\sigma^2}{N} \mathbb{E}\left[\underbrace{X^T}_{1\times(p+1)} \underbrace{\mathbb{E}[XX^T]^{-1}}_{(p+1)\times(p+1)} \underbrace{X}_{(p+1)\times 1}\right] = \sigma^2 + \frac{\sigma^2}{N} \mathbb{E}\left[tr\left(X^T \mathbb{E}[XX^T]^{-1} X\right)\right]$$

Under trace operator, matrix multiplication is a commutative operation:

$tr(AB) = tr(BA)$

Although $AB \neq BA$

$$= \sigma^2 + \frac{\sigma^2}{N} \mathbb{E}\left[tr\left(XX^T \cdot \mathbb{E}[XX^T]^{-1}\right)\right]$$

$\mathbb{E}$ and tr commute: $\mathbb{E}[tr(*)] = tr\,\mathbb{E}[*]$

$$= \sigma^2 + \frac{\sigma^2}{N} tr\left(\mathbb{E}\left[XX^T \cdot \mathbb{E}[XX^T]^{-1}\right]\right)$$

$$= \sigma^2 + \frac{\sigma^2}{N} tr\left(\mathbb{E}[XX^T] \cdot \mathbb{E}[XX^T]^{-1}\right) = \sigma^2 + \frac{\sigma^2}{N} tr\, I_{p+1} = \sigma^2 + \frac{\sigma^2}{N}(p+1)$$

Thus, we have: $\boxed{\mathbb{E}[MSE(X)] \approx \sigma^2 + \frac{\sigma^2}{N} p}$

As intuitively expected, prediction accuracy ↑ as the sample size N ↑

↓ as the dimension p ↑

↓ as "noise strength" σ ↑

Important observation:

We can suppress the curse of dimensionality by making σ^2/N small enough.

If σ^2/N is small, then the growth in the prediction error associated with the increase of p is negligible.

(but this is only if $Y \approx X^T\beta$)