

# Optimization and Machine Learning SI151

Lu Sun

School of Information Science and Technology

ShanghaiTech University

February 25, 2020

Today:

- Overview of supervised learning II
  - Statistical decision theory
  - Local methods in high dimensions
  - Statistical models
  - Model selection

Readings:

- The Elements of Statistical Learning (ESL), Chapter 2

# Overview of Supervised Learning II

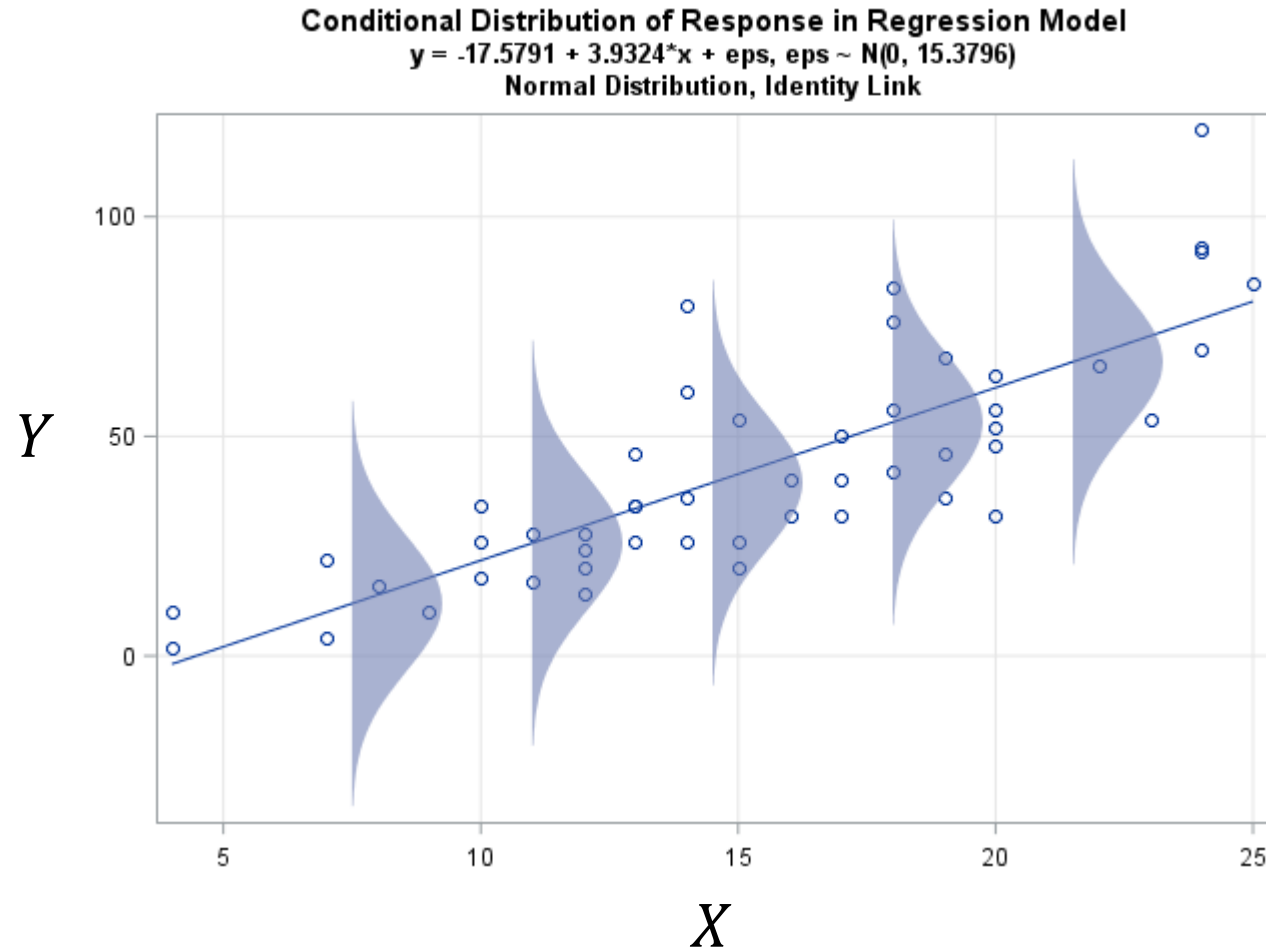
--- Statistical Decision Theory

# Statistical Decision Theory

- **Given:**
  - random input vector  $X \in \mathbb{R}^p$ ,
  - random output variable  $Y \in \mathbb{R}$ ,
  - joint distribution  $\Pr(X, Y)$ ,
- **Goal:** we seek a function  $f(X)$  for predicting  $Y$  given values of  $X$ .
- To penalize prediction errors, we introduce the *loss function*  $L(Y, f(X))$ .
- **Squared error loss:**
$$L(Y, f(X)) = (Y - f(X))^2.$$
- **Expected prediction error (EPE):**
$$\begin{aligned} \text{EPE}(f) &= E(Y - f(X))^2 \\ &= \int (y - f(x))^2 \Pr(dx, dy). \end{aligned}$$
- Since  $\Pr(X, Y) = \Pr(Y|X) \Pr(X)$ , EPE can also be written as
$$\text{EPE}(f) = E_X E_{Y|X}([Y - f(X)]^2 | X).$$
- Thus, it suffices to minimize EPE *pointwise*:
$$f(x) = \operatorname{argmin}_c E_{Y|X}([Y - c]^2 | X = x)$$

Regression function:  $f(x) = E(Y|X = x)$ .

# Statistical Decision Theory



Regression function:  $f(x) = E(Y|X = x)$ .

# Statistical Decision Theory

- **Nearest neighbor methods** try to directly implement this recipe
$$\hat{f}(x) = \text{Ave}(y_i | x_i \in N_k(x)).$$
- Two approximations:
  - expectation is approximated by **averaging over sample data**;
  - conditioning at a point is relaxed to **conditioning on neighborhood**.
- As  $N, k \rightarrow \infty$  and  $\frac{k}{N} \rightarrow 0$ , we have
$$\hat{f}(x) \rightarrow E(Y|X = x).$$
- But usually we do not have very large samples.
  - By making assumptions (**linearity**), we can reduce the required number of observations greatly.
- As increasing the number  $p$  of dimensions, the number  $N$  of observations required in the training data set **increases exponentially**.
  - Thus the *rate of convergence* to the true estimator (with increasing  $p$ ) decreases.

Regression function:  $f(x) = E(Y|X = x)$ .

# Statistical Decision Theory

- **Linear regression** assumes that the regression function is approximately linear

$$f(x) \approx x^T \beta.$$

- This is a **model-based approach**.
- Plugging this  $f(x)$  into EPE,

$$\begin{aligned} \text{EPE}(f) &= E(Y - f(X))^2 \\ &= E((Y - X^T \beta)^T (Y - X^T \beta)) \end{aligned}$$

- Differentiating w.r.t.  $\beta$ , leads to

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

- Again, linear regression **replaces the theoretical expectation** by averaging over the observed data

$$\text{RSS}(\beta) = \sum_{i=1}^N (y_i - x_i^T \beta)^2$$

$$\hat{\beta} = (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{y}$$

- Summary – approximation of  $f(X)$ 
  - **Least squares**:  
globally linear function
  - **Nearest neighbors**:  
locally constant function.

Regression function:  $f(x) = E(Y|X = x)$ .

# Statistical Decision Theory

- **Additional methods** in our course are often model-based but more flexible than the linear model.
- For example, **additive models**

$$f(X) = \sum_{j=1}^p f_j(X_j)$$

- Coordinate function  $f_j$  is arbitrary.
- Approximate *univariate* conditional expectations *simultaneously* for each  $f_j$ .
- Model assumption: **additivity**.

- What happens if we use **absolute loss function**?

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

- In this case,
$$\hat{f}(x) = \text{median}(Y|X = x)$$
- More **robust** than the conditional mean.
- Summary:
  - $L_1$  criterion **not differentiable**.
  - Squared error is the most popular.

**Q:** How to obtain the  $\hat{f}(x)$  when absolute loss is used?

# Statistical Decision Theory

- Procedure for **categorical output variable**  $G$  with values from  $\mathcal{G}$ .
- **Loss function** is  $K \times K$  matrix  $\mathbf{L}$ , where  $K = \text{card}(\mathcal{G})$
- $\mathbf{L}(k, l)$  is the price paid for misclassifying an observation belonging to class  $\mathcal{G}_k$  as class  $\mathcal{G}_l$
- $\mathbf{L}$  is zero on the diagonal
- We often use the **zero-one loss** function

$$\mathbf{L}(k, l) = 1 - \delta_{kl}$$

where  $\delta_{kl} = 1$  if  $k = l$ , otherwise  $\delta_{kl} = 0$

- **Expected prediction error (EPE)**

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

where expectation taken w.r.t.  $\Pr(G, X)$

- Conditioning on  $X$  yields


$$\text{EPE} = \mathbb{E}_X \sum_{k=1}^K L[\mathcal{G}_k, \hat{G}(X)] \Pr(\mathcal{G}_k | X)$$

- Again, it suffices to pointwise minimization

$$\hat{G}(x) = \operatorname{argmin}_{g \in \mathcal{G}} \sum_{k=1}^K L(\mathcal{G}_k, g) \Pr(\mathcal{G}_k | X = x)$$

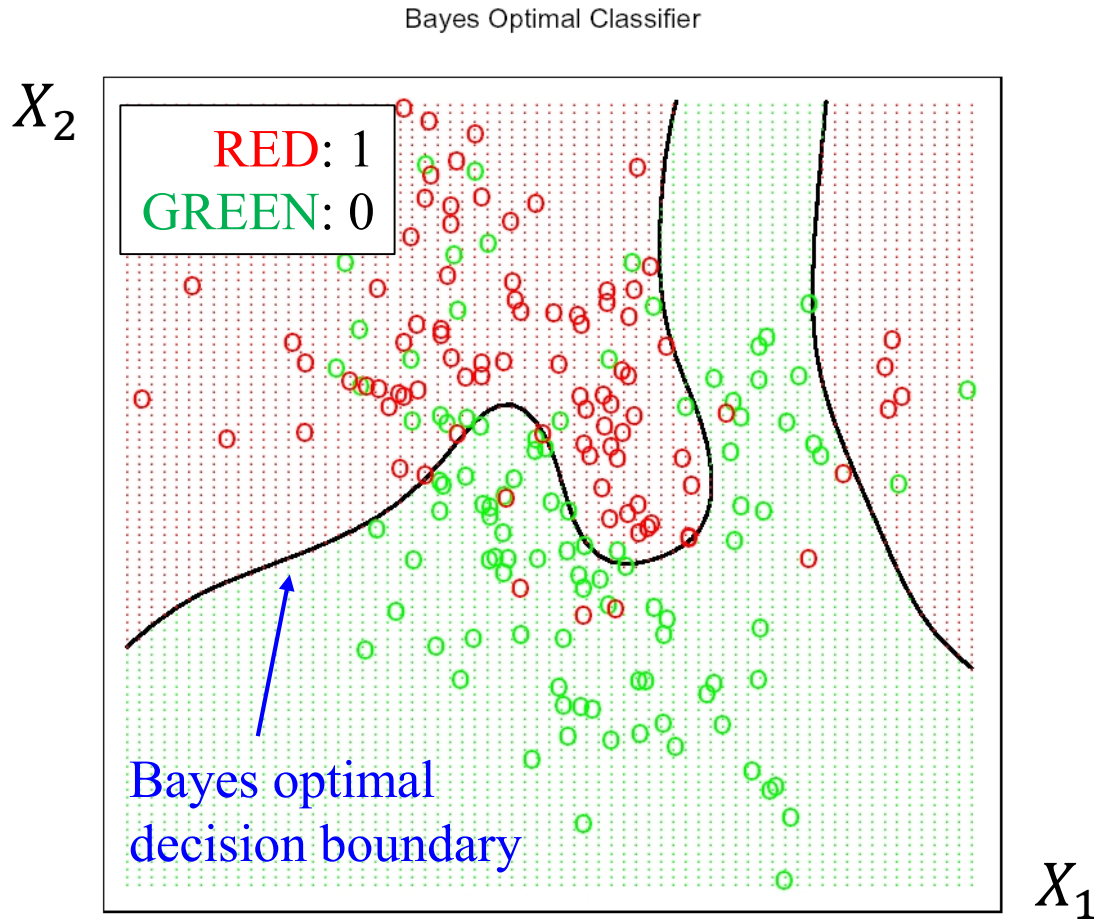
- Or simply

$$\hat{G}(x) = \operatorname{argmax}_{g \in \mathcal{G}} \Pr(g | X = x)$$

 **Bayes classifier**



# Statistical Decision Theory



Since the generating density is known for each class, this boundary can be calculated exactly.

- Expected prediction error (EPE)

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

where expectation taken w.r.t.  $\Pr(G, X)$

- Conditioning on  $X$  yields

$$\text{EPE} = \mathbb{E}_X \sum_{k=1}^K L(\mathcal{G}_k, \hat{G}(X)) \Pr(\mathcal{G}_k | X)$$

- Again, it suffices to pointwise minimization

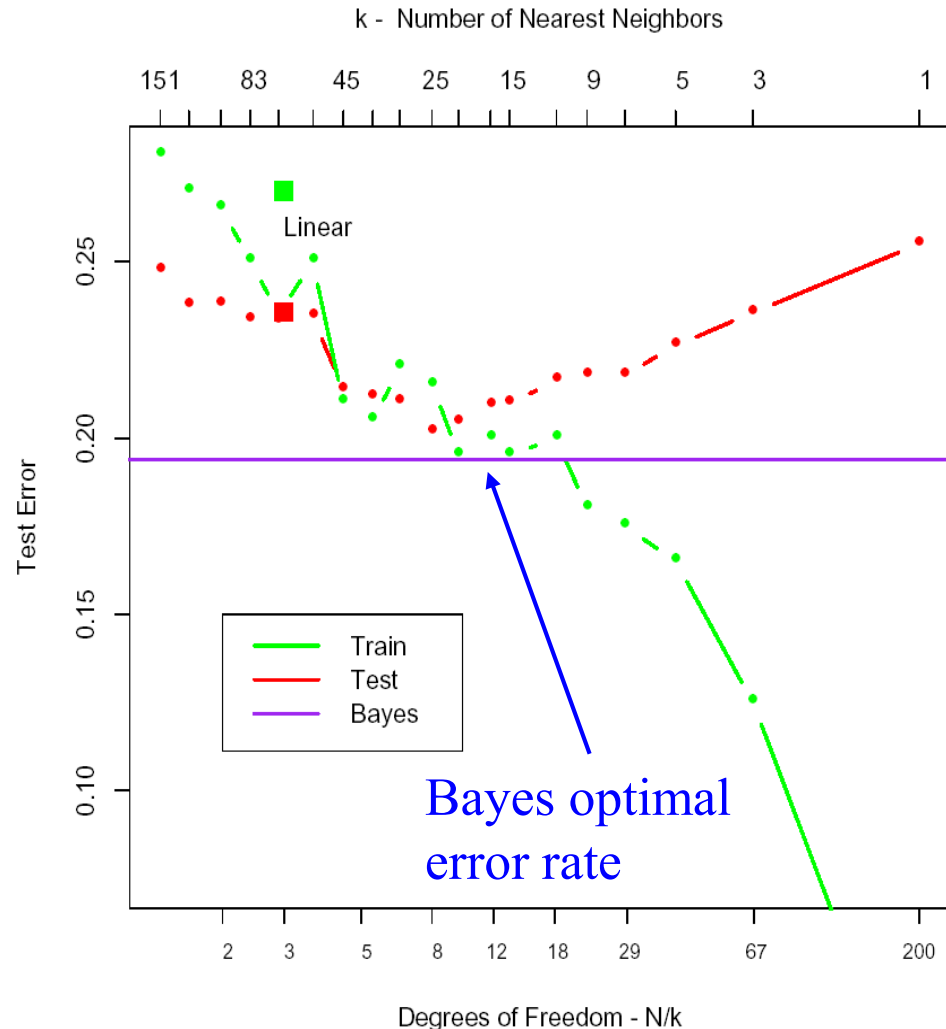
$$\hat{G}(x) = \operatorname{argmin}_{g \in \mathcal{G}} \sum_{k=1}^K L(\mathcal{G}_k, g) \Pr(\mathcal{G}_k | X = x)$$

- Or simply

$$\hat{G}(x) = \operatorname{argmax}_{g \in \mathcal{G}} \Pr(g | X = x)$$

Bayes classifier

# Statistical Decision Theory



- Expected prediction error (EPE)

$$\text{EPE} = E[L(G, \hat{G}(X))]$$

where expectation taken w.r.t.  $\Pr(G, X)$

- Conditioning on  $X$  yields

$$\text{EPE} = E_X \sum_{k=1}^K L[\mathcal{G}_k, \hat{G}(X)] \Pr(\mathcal{G}_k | X)$$

- Again, it suffices to pointwise minimization

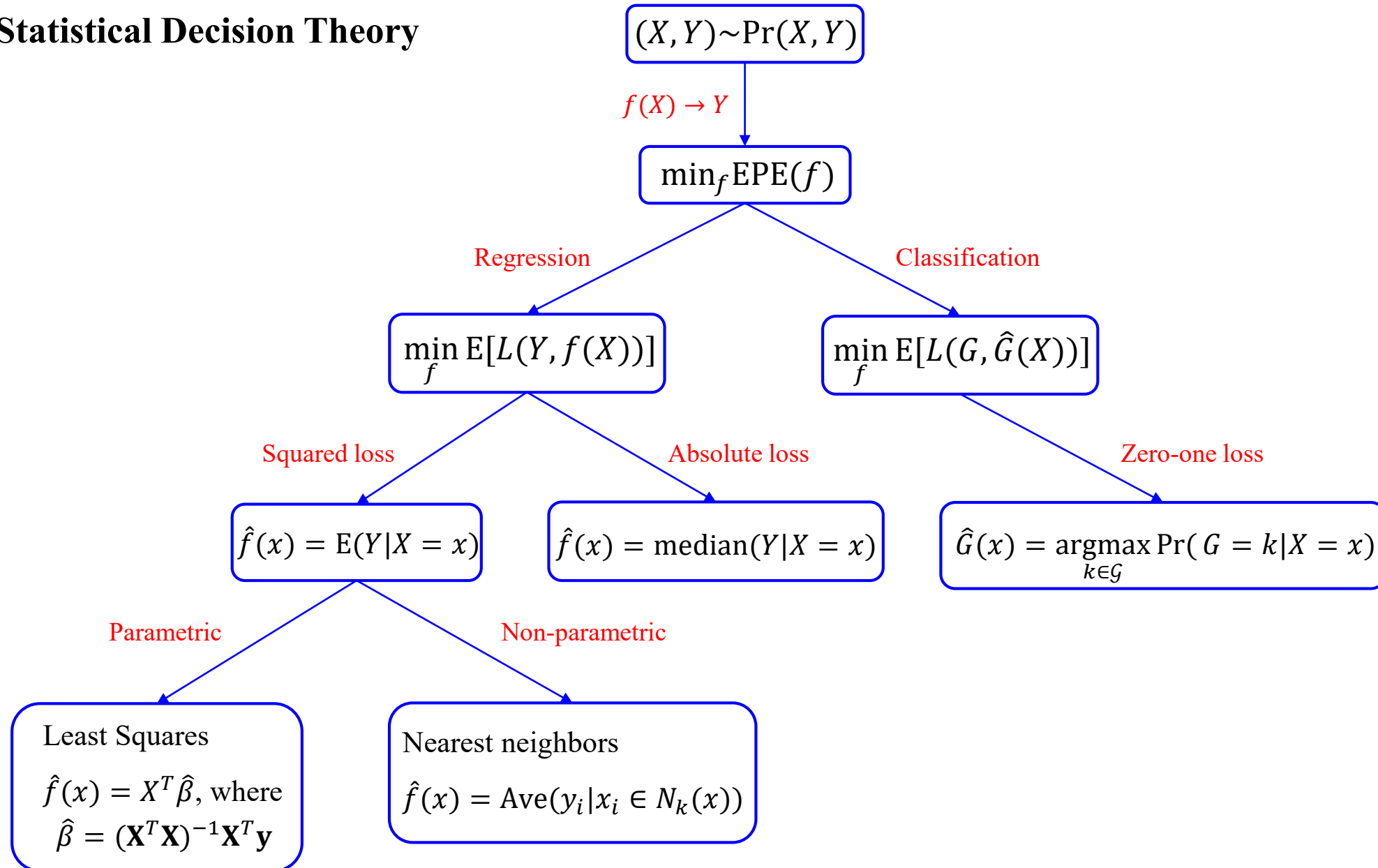
$$\hat{G}(x) = \operatorname{argmin}_{g \in G} \sum_{k=1}^K L(\mathcal{G}_k, g) \Pr(\mathcal{G}_k | X = x)$$

- Or simply

$$\hat{G}(x) = \operatorname{argmax}_{g \in G} \Pr(g | X = x)$$

Bayes classifier

# Statistical Decision Theory

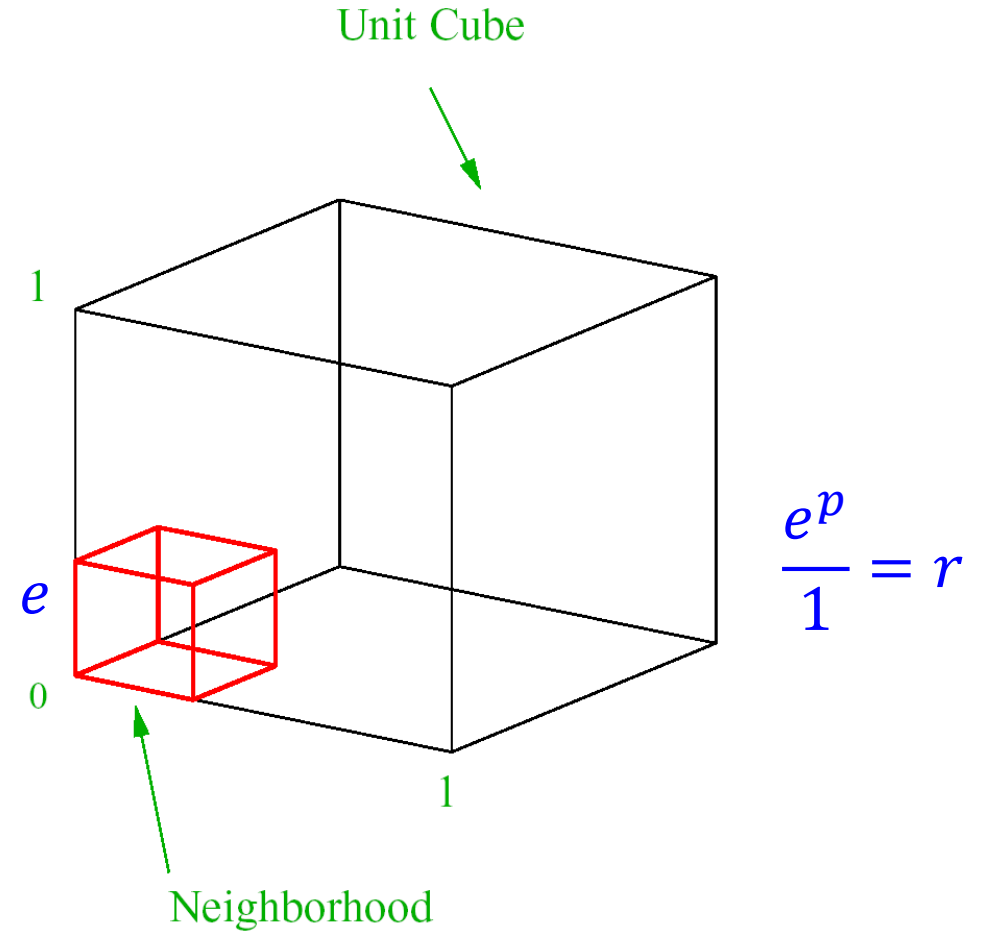


# Overview of Supervised Learning II

--- Local Methods in High Dimensions

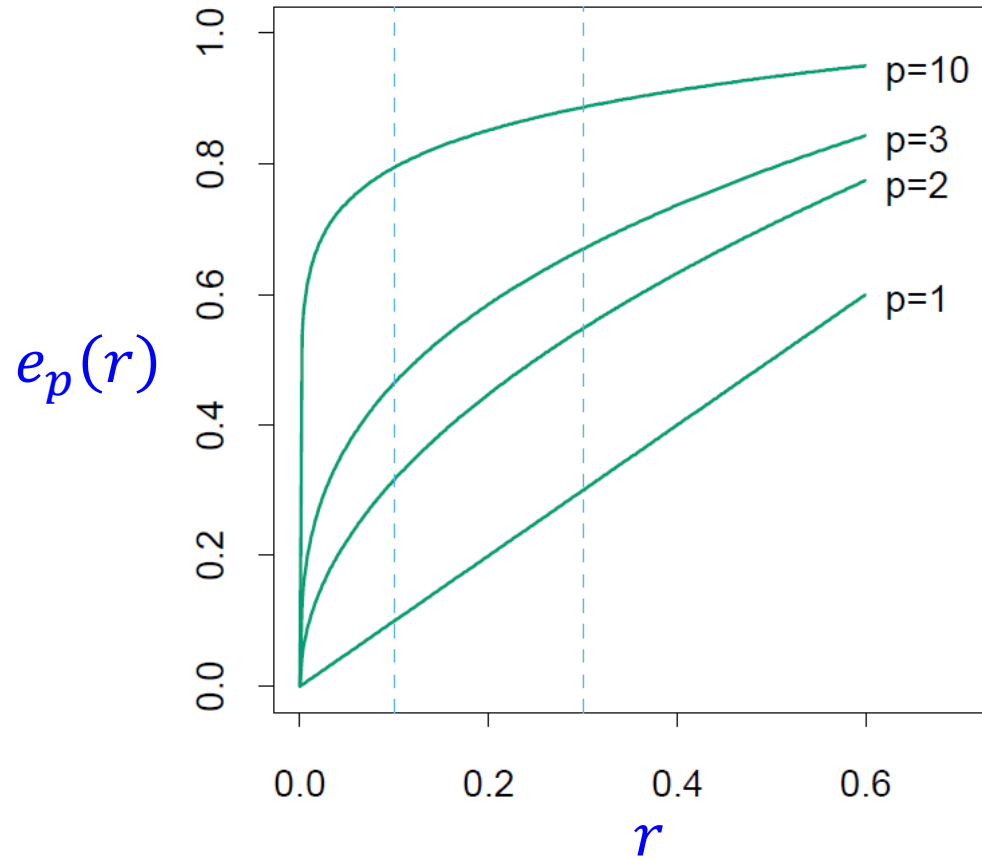
# Local Models in High Dimensions

- **Curse of Dimensionality:**  
Local neighborhoods become increasingly global, as the number of dimension increases
- **Example:**  
Points uniformly distributed in a  $p$ -dimensional unit hypercube.
- Hypercubical neighborhood in  $p$  dimensions that captures a fraction  $r$  of the data
  - edge length:  $e_p(r) = r^{\frac{1}{p}}$
  - $e_{10}(0.01) = 0.63$
  - $e_{10}(0.1) = 0.80$

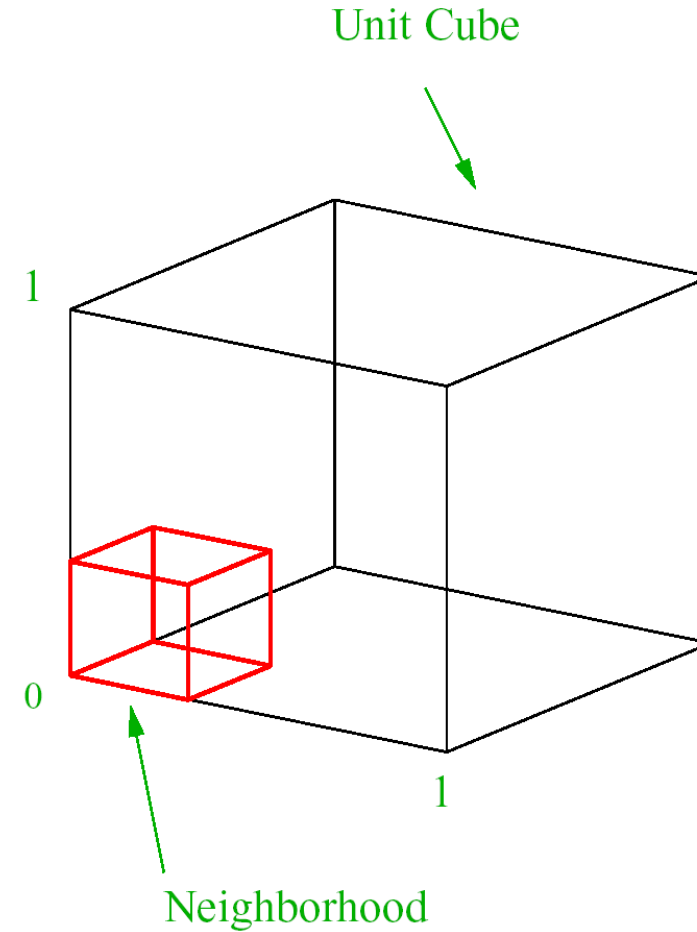


In **ten** dimensions we need to cover **63%** (**80%**) of the range of each coordinate to capture **1%** (**10%**) of the data.

# Local Models in High Dimensions



Reducing  $r$  reduces the number of observations and thus the stability.



In **ten** dimensions we need to cover **63%** (**80%**) of the range of each coordinate to capture **1%** (**10%**) of the data.

# Local Models in High Dimensions

- In high dimensions, all sample points are close to the edge of the sample
- $N$  data points uniformly distributed in a  $p$ -dimensional unit ball centered at the origin
- **Median distance** from the closest point to the origin

$$d(p, N) = \left(1 - \frac{1}{2}^{1/N}\right)^{1/p}$$

- $d(10,500) \approx 0.52$ : more than **half** the way to the boundary

$$(1) \quad \prod_{i=1}^N \Pr(\|x_i\| > r) = \frac{1}{2}$$

$$(2) \quad \Pr(\|x_i\| > r) = 1 - \Pr(\|x_i\| \leq r) = 1 - r^p$$

$$(3) \quad (1 - r^p)^N = \frac{1}{2}$$

$$\text{Volume of a } p\text{-ball: } V_p(r) = \frac{\pi^{\frac{p}{2}}}{\Gamma(\frac{p}{2}+1)} r^p$$

# Local Models in High Dimensions

- In high dimensions, all sample points are close to the edge of the sample
- $N$  data points uniformly distributed in a  $p$ -dimensional unit ball centered at the origin
- **Median distance** from the closest point to the origin
- Sampling density is proportional to  $N^{1/p}$
- If  $N_1 = 100$  is a dense sample for one input, then  $N_{10} = 100^{10}$  is an equally dense sample for 10 inputs.
- Thus in high dimensions all feasible training samples **sparsely populate** the input space.

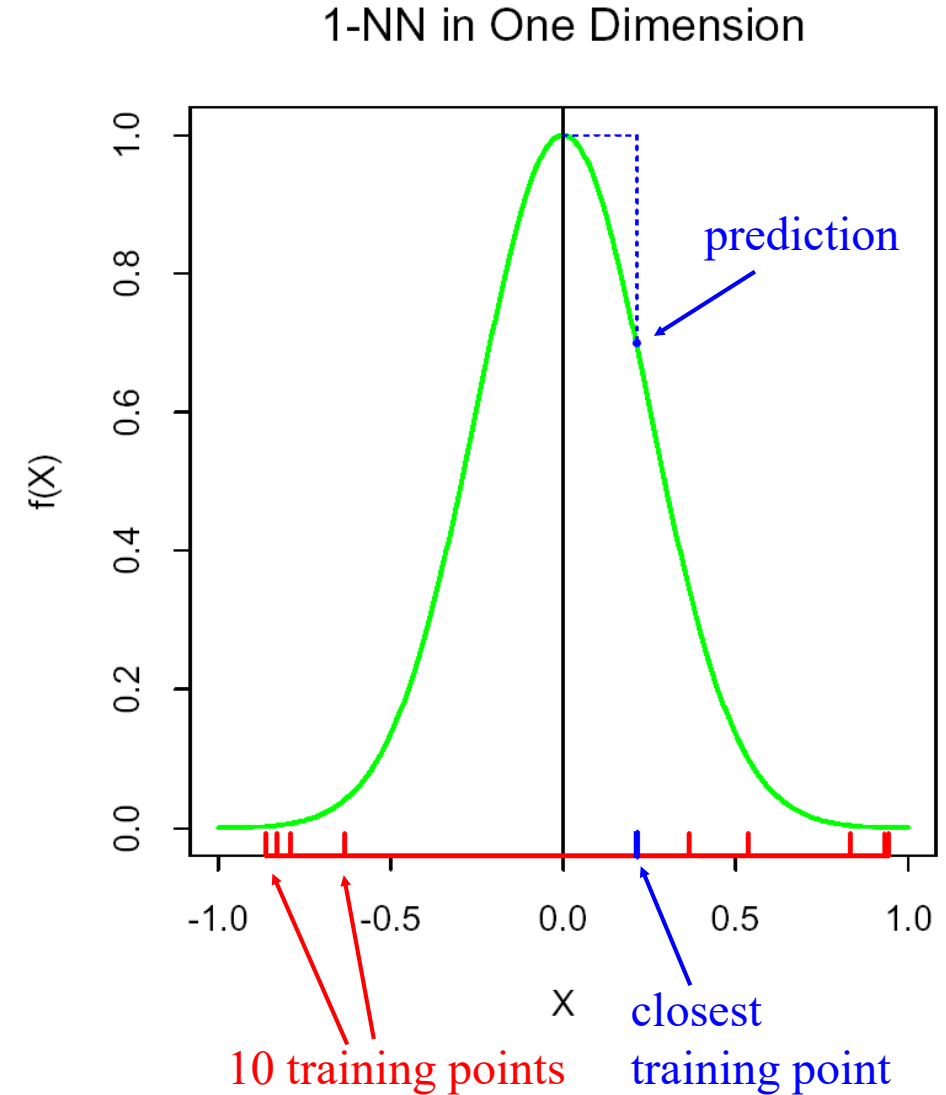
$$d(p, N) = \left(1 - \frac{1}{2}^{1/N}\right)^{1/p}$$

- $d(10, 500) \approx 0.52$ : more than **half** the way to the boundary



# Local Models in High Dimensions

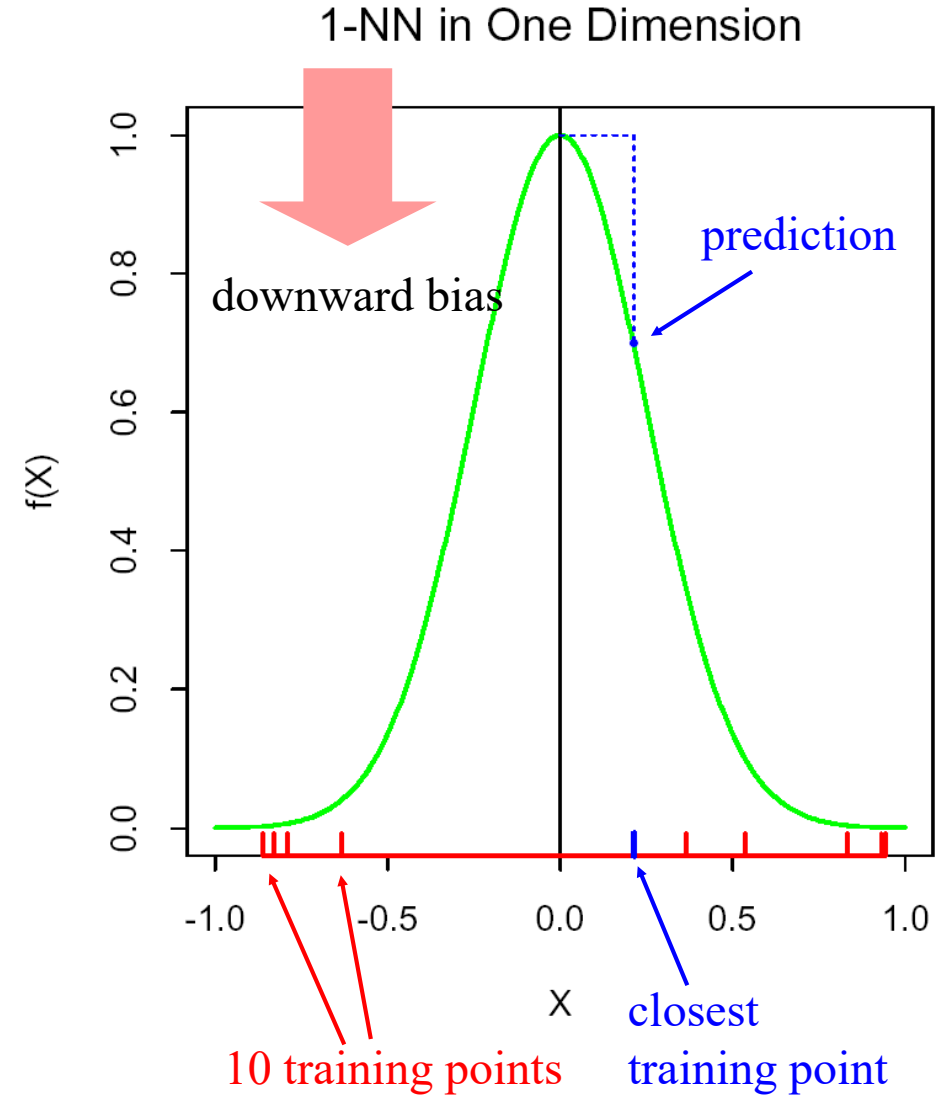
- Another example
- $\mathcal{T}$ : set of training points  $x_i$  generated uniformly in  $[-1,1]^p$  (red)
- Functional relationship between  $X$  and  $Y$  (green)
$$Y = f(X) = e^{-8\|X\|^2}$$
- No measurement error
- Error of a 1-nearest neighbor classifier in estimating  $f(0)$  (blue)



# Local Models in High Dimensions

- Another example
- Problem deterministic:  
Prediction error is the **mean-squared error** for estimating  $f(0)$

$$\begin{aligned}\text{MSE}(x_0) &= \mathbb{E}_{\mathcal{T}}[f(x_0) - \hat{y}_0]^2 \\ &= \mathbb{E}_{\mathcal{T}}[\hat{y}_0 - \mathbb{E}_{\mathcal{T}}(\hat{y}_0)]^2 \\ &\quad + [\mathbb{E}_{\mathcal{T}}(\hat{y}_0) - f(x_0)]^2 \\ &= \text{Var}_{\mathcal{T}}(\hat{y}_0) + \text{Bias}^2(\hat{y}_0)\end{aligned}$$



# Local Models in High Dimensions

$$\begin{aligned}\text{MSE}(x_0) &= E_{\mathcal{T}}[f(x_0) - \hat{y}_0]^2 \\ &= E_{\mathcal{T}}[\hat{y}_0 - E_{\mathcal{T}}(\hat{y}_0) + E_{\mathcal{T}}(\hat{y}_0) - f(x_0)]^2 \\ &= E_{\mathcal{T}} \left[ (\hat{y}_0 - E_{\mathcal{T}}(\hat{y}_0))^2 + \underbrace{2(\hat{y}_0 - E_{\mathcal{T}}(\hat{y}_0))(E_{\mathcal{T}}(\hat{y}_0) - f(x_0))}_{E_{\mathcal{T}}(\hat{y}_0 - E_{\mathcal{T}}(\hat{y}_0))(E_{\mathcal{T}}(\hat{y}_0) - f(x_0)) = 0} + \underbrace{(E_{\mathcal{T}}(\hat{y}_0) - f(x_0))^2}_{\text{Constant}} \right] \\ &= E_{\mathcal{T}} \left[ (\hat{y}_0 - E_{\mathcal{T}}(\hat{y}_0))^2 \right] + (E_{\mathcal{T}}(\hat{y}_0) - f(x_0))^2 \\ &= \text{Var}_{\mathcal{T}}(\hat{y}_0) + \text{Bias}^2(\hat{y}_0)\end{aligned}$$

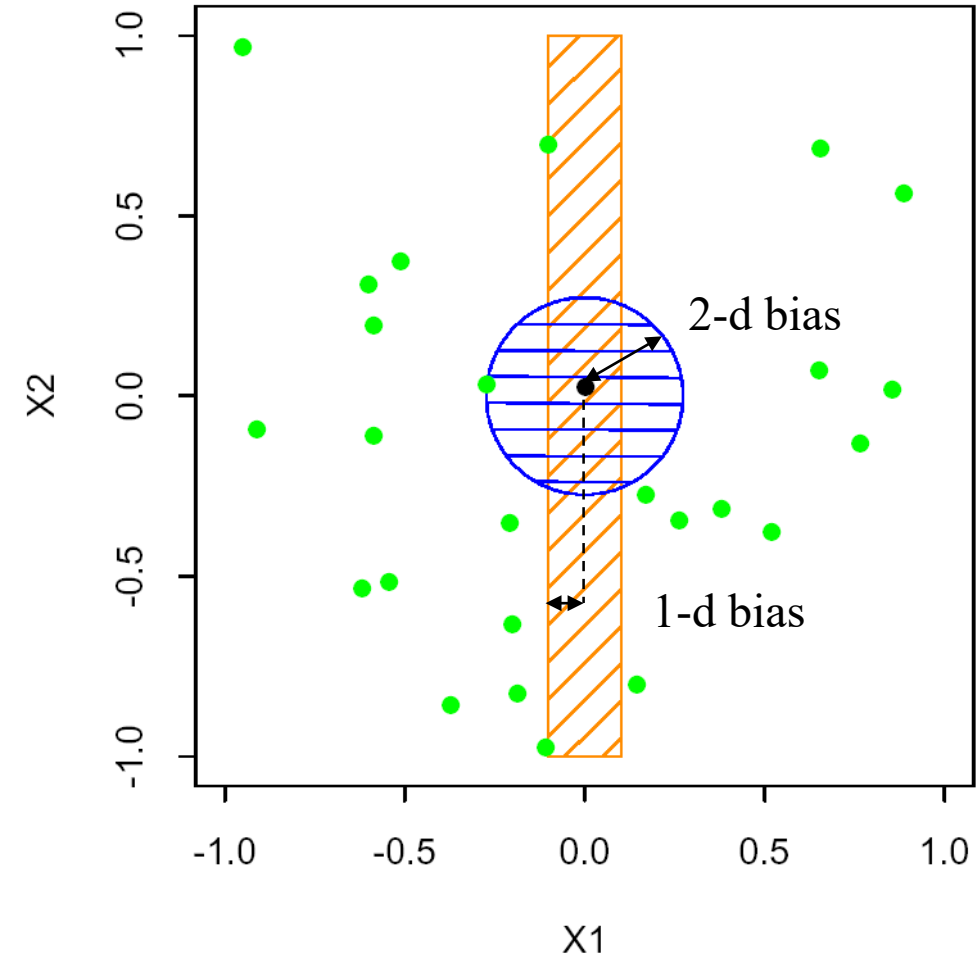
This is known as **the bias-variance decomposition**.

# Local Models in High Dimensions

- Another example
- 1-d (red) vs 2-d (blue)
- As  $p$  increases, the bias increases

$$\begin{aligned}\text{MSE}(x_0) &= \mathbb{E}_{\mathcal{T}}[f(x_0) - \hat{y}_0]^2 \\ &= \mathbb{E}_{\mathcal{T}}[\hat{y}_0 - \mathbb{E}_{\mathcal{T}}(\hat{y}_0)]^2 \\ &\quad + [\mathbb{E}_{\mathcal{T}}(\hat{y}_0) - f(x_0)]^2 \\ &= \text{Var}_{\mathcal{T}}(\hat{y}_0) + \text{Bias}^2(\hat{y}_0)\end{aligned}$$

1-NN in One vs. Two Dimensions

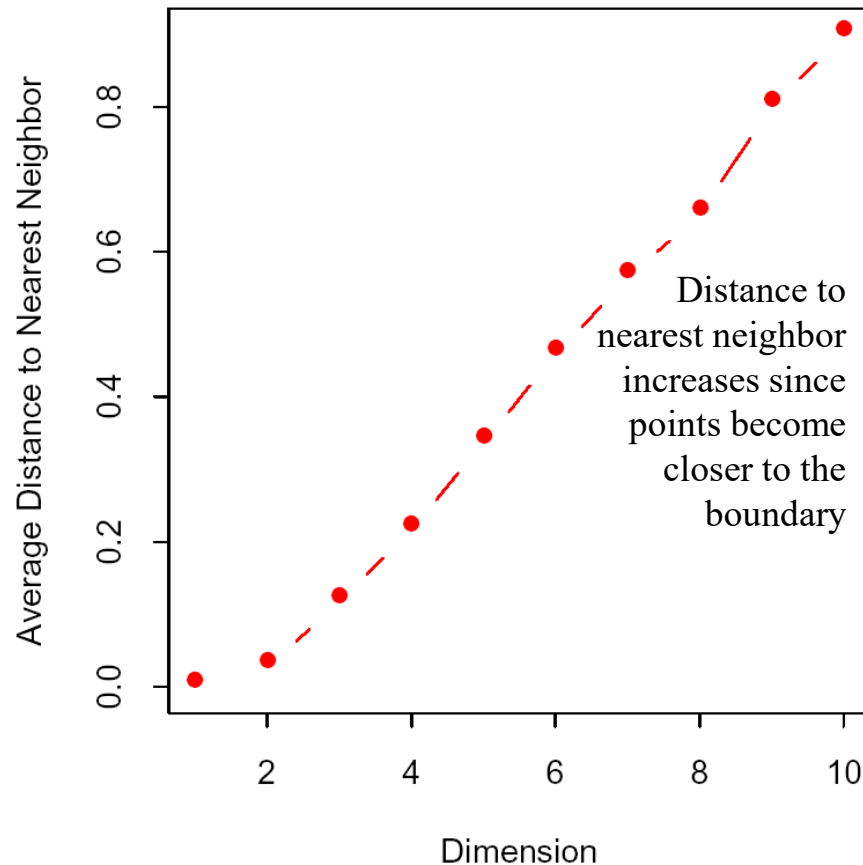


# Local Models in High Dimensions

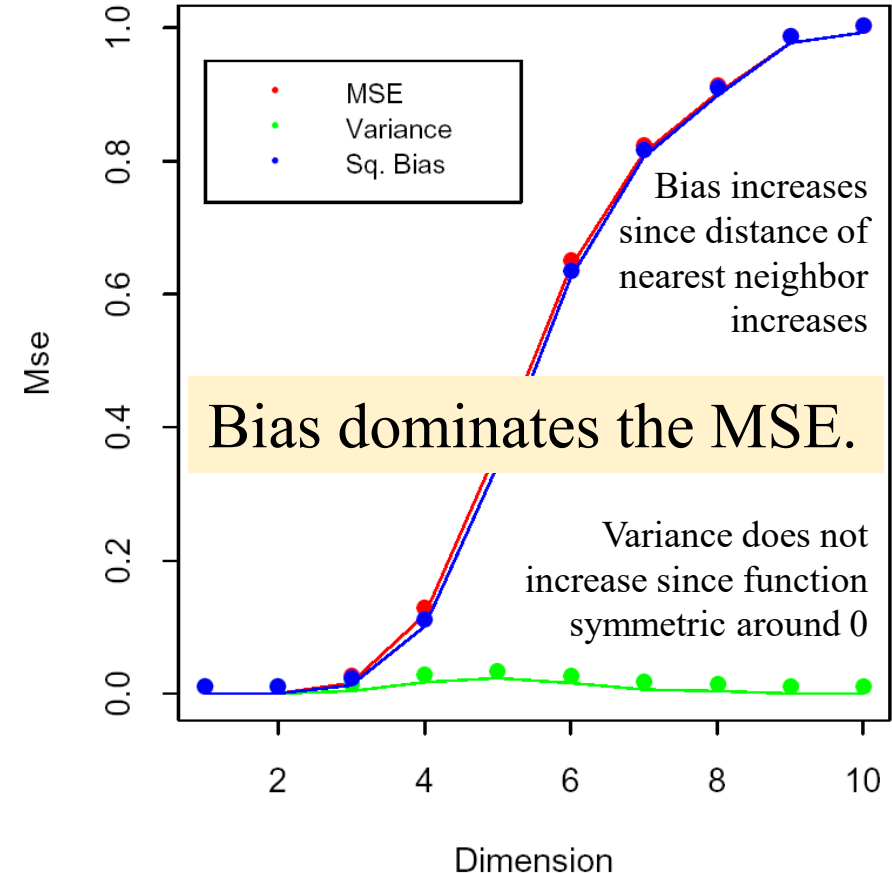
$$Y = f(X) = e^{-8\|X\|^2}$$

- The case on  $N=1000$  training points

Distance to 1-NN vs. Dimension



MSE vs. Dimension

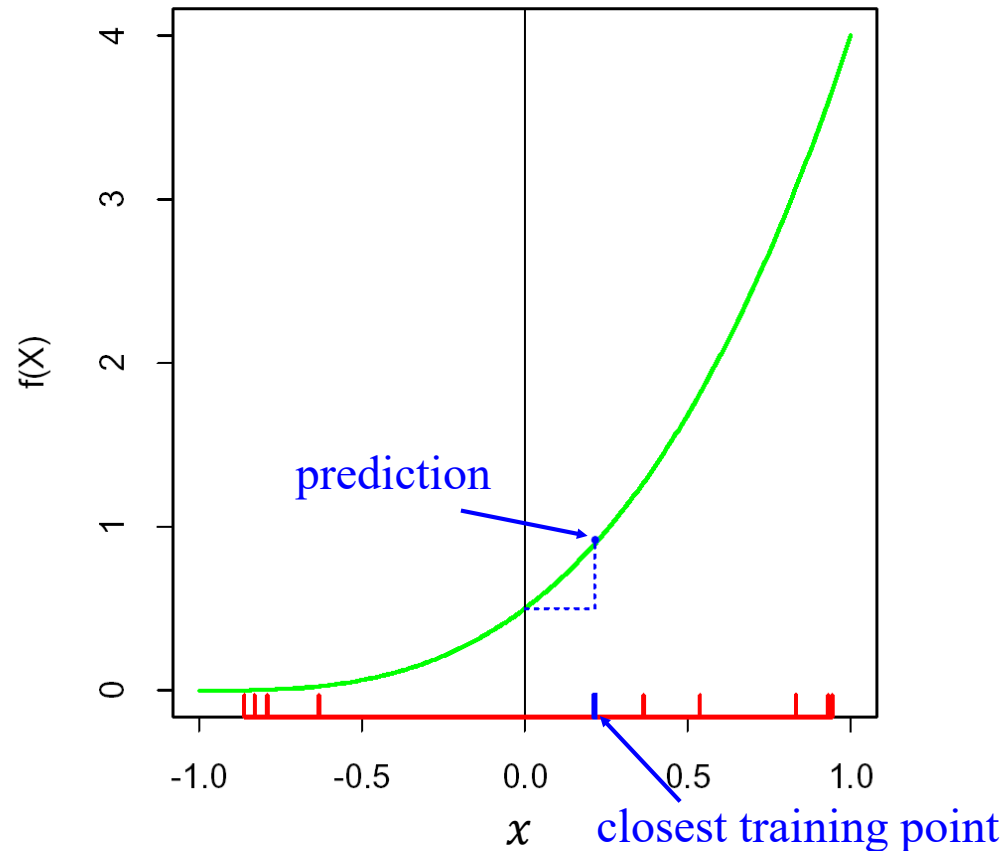


# Local Models in High Dimensions

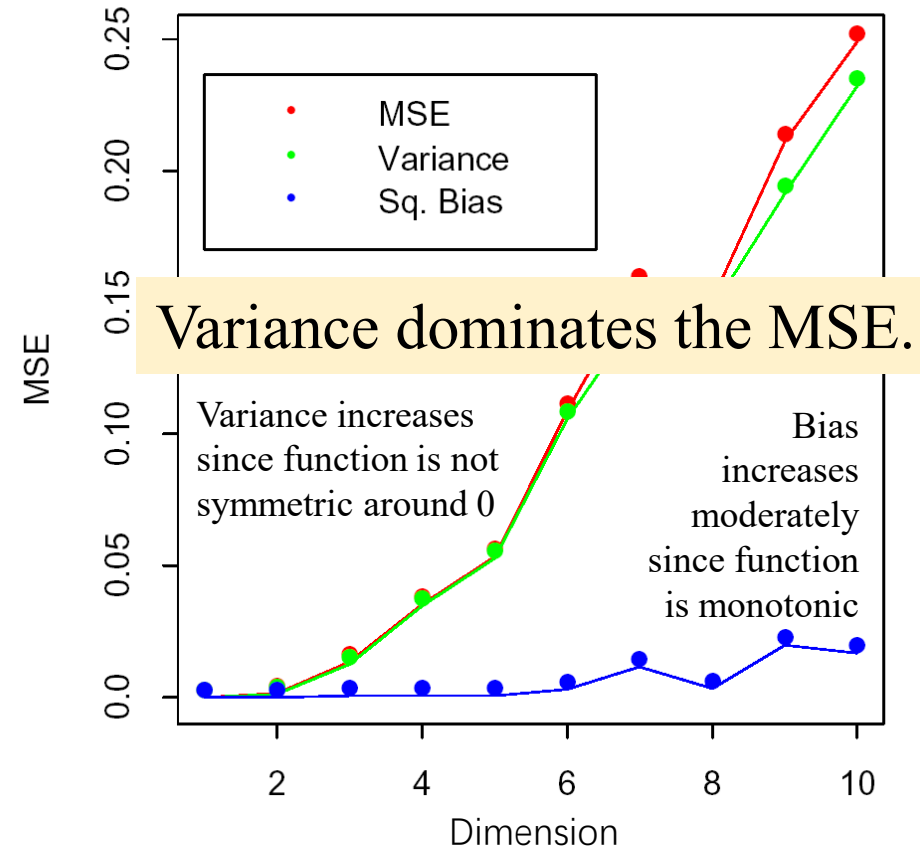
- Yet another example

$$Y = f(X) = \frac{1}{2}(X_1 + 1)^3$$

1-NN in One Dimension



MSE vs. Dimension



# Local Models in High Dimensions

- Suppose a linear relationship with measurement error

$$Y = X^T \beta + \varepsilon, \quad \varepsilon \sim \mathcal{N}(0, \sigma^2)$$

- If the model is fitted by **least squares**, we find that

$$\text{EPE}(x_0) = \sigma^2 + \mathbb{E}_{\mathcal{J}}[x_0^T (\mathbf{X}^T \mathbf{X})^{-1} x_0] \sigma^2$$

- Additional variance  $\sigma^2$  originates from the **nondeterministic part**
- Variance depends on  $x_0$
- No bias

- If  $N$  is large, we get

$$\mathbb{E}_{x_0} \text{EPE}(x_0) \sim \frac{\sigma^2}{N} p + \sigma^2$$

- As  $p$  increases, variance grows negligible for large  $N$  or small  $\sigma^2$
- Curse of dimensionality **controlled**

# Local Models in High Dimensions

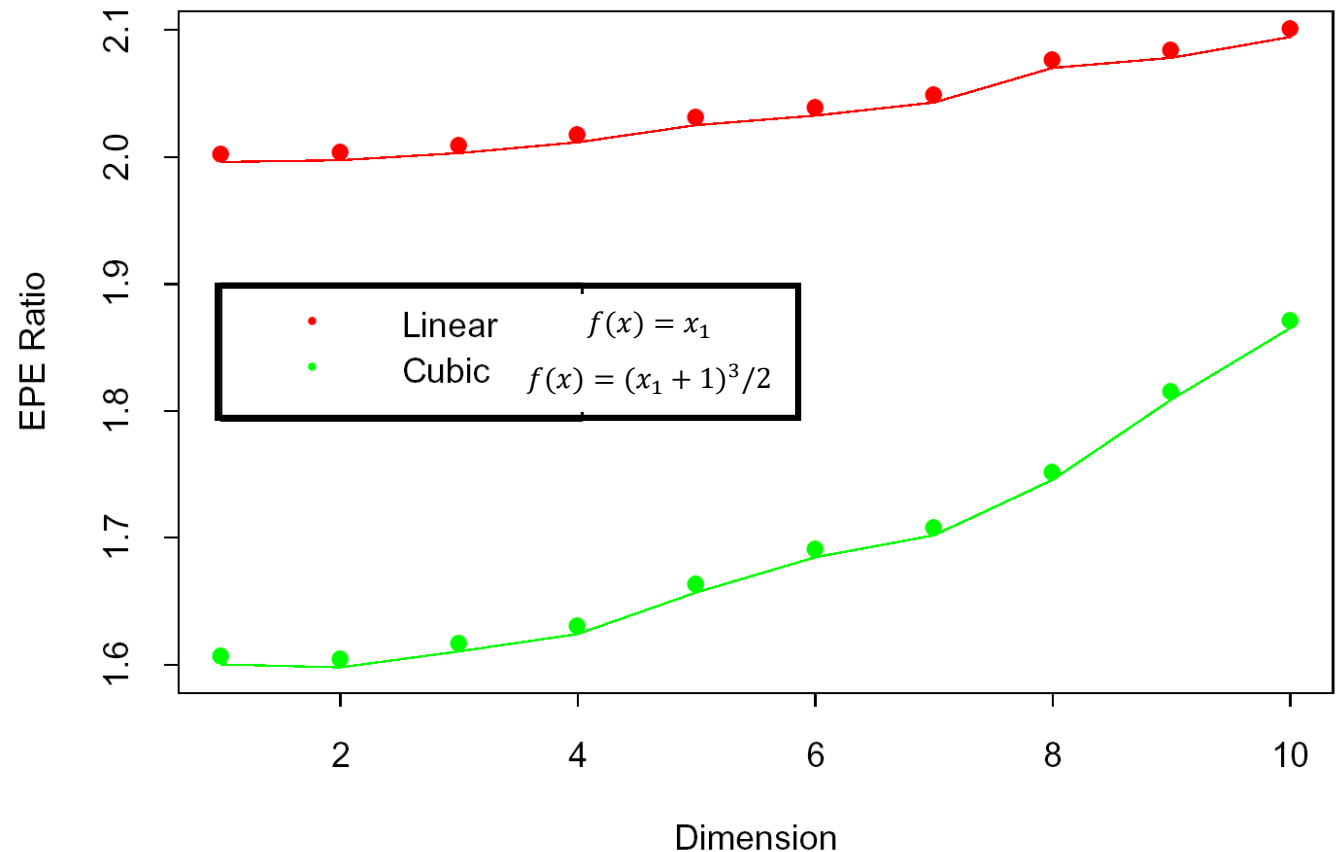
- More generally

$$Y = f(X) + \varepsilon,$$

$X$  uniform,  $\varepsilon \sim \mathcal{N}(0,1)$

- Sample size:  $N = 500$
- Linear case
  - EPE (Least Squares) is slightly above 1, no bias
  - EPE (1-NN) always above 2, grows slowly as nearest training point strays from target

$$\text{EPE ratio} = \frac{\text{EPE (1-NN)}}{\text{EPE (least squares)}}, \text{ at } x_0 = 0$$





# Local Models in High Dimensions

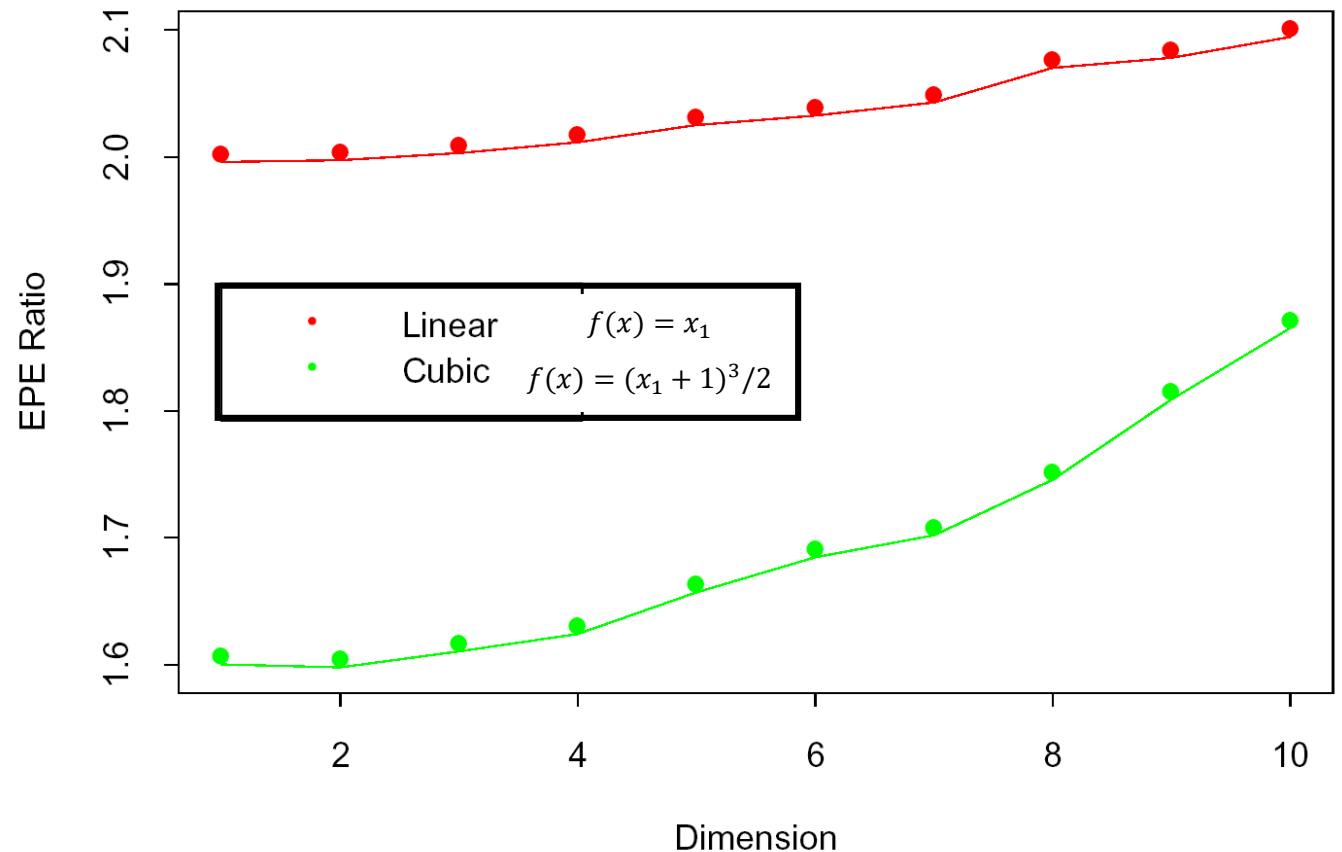
- More generally

$$Y = f(X) + \varepsilon,$$

$X$  uniform,  $\varepsilon \sim \mathcal{N}(0,1)$

- Sample size:  $N = 500$
- Cubic case
  - EPE (Least Squares) is biased, thus ratio is smaller

$$\text{EPE ratio} = \frac{\text{EPE (1-NN)}}{\text{EPE (least squares)}}, \text{ at } x_0 = 0$$



# Local Models in High Dimensions – Summary

- **Curse of Dimensionality**
  1. Local neighborhoods become **increasingly global**, as the number of dimension increases
  2. In high dimensions, all samples are **close to the edge** of the sample
  3. Samples **sparsely populate** the input space
- **The bias-variance decomposition**
  1. **Deterministic** case
$$\begin{aligned}\text{EPE}(x_0) &= \text{MSE}(x_0) \\ &= \text{Var}_{\mathcal{T}}(\hat{y}_0) + \text{Bias}^2(\hat{y}_0)\end{aligned}$$
  2. **Non-deterministic** case
$$\begin{aligned}\text{EPE}(x_0) &= \text{MSE}(x_0) + \sigma^2 \\ &= \text{Var}_{\mathcal{T}}(\hat{y}_0) + \text{Bias}^2(\hat{y}_0) + \sigma^2\end{aligned}$$
- **Least squares**
  - **Linear** case: non-biased, negligible variance for large  $N$
  - **Non-linear** case: biased
- **Nearest neighbors**
  - **Symmetric** on  $x_0$ :  $\text{Bias}^2(\hat{y}_0)$  dominates
  - **Monotonic** on  $x_0$ :  $\text{Var}_{\mathcal{T}}(\hat{y}_0)$  dominates

# Overview of Supervised Learning II

--- Statistical Models

# Statistical Models – Function Approximation

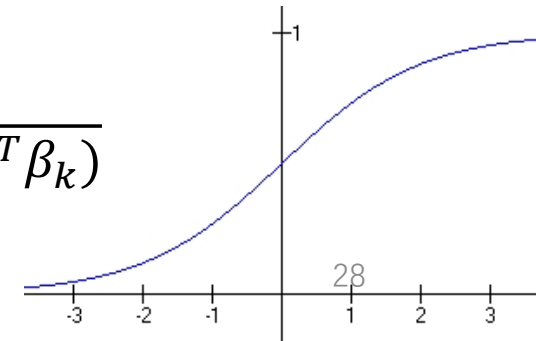
- **Data:** pairs  $(x_i, y_i)$  that are points in  $(p + 1)$ -dimensional Euclidean space, we fit  $f: \mathbb{R}^p \rightarrow \mathbb{R}$  by
- **Linear basis expansions** have the more general form

$$y_i = f(x_i) + \varepsilon_i$$

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

- **Goal:** a good approximation of  $f(x)$  in some region of input space, given the training set  $\mathcal{T}$
- Many models have certain parameters  $\theta$ 
  - E.g. for the linear model  $f(x) = x^T \beta$  and  $\theta = \beta$
- $h_k$ : a suitable set of functions or transformations of the input vector  $x$ .
- Examples:
  - Polynomial expansions:  $h_k(x) = x_1 x_2^2$
  - Trigonometric expansions:  $h_k(x) = \cos(x_1)$
  - Sigmoid expansion:

$$h_k(x) = \frac{1}{1 + \exp(-x^T \beta_k)}$$



# Statistical Models – Function Approximation

- Approximating  $f_\theta$  by minimizing the **residual sum of squares**

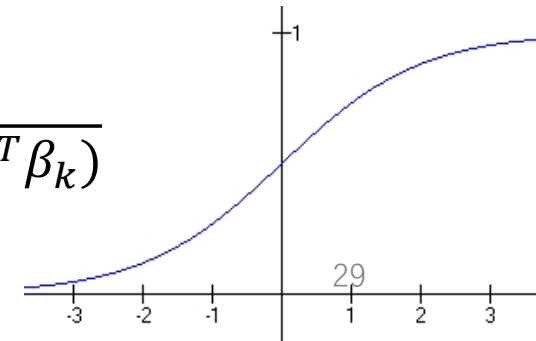
$$\text{RSS}(\theta) = \sum_{i=1}^N (y_i - f_\theta(x_i))^2$$

- Linear basis expansions** have the more general form

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

- $h_k$ : a suitable set of functions or transformations of the input vector  $x$ .
- Examples:
  - Polynomial expansions:  $h_k(x) = x_1 x_2^2$
  - Trigonometric expansions:  $h_k(x) = \cos(x_1)$
  - Sigmoid expansion:

$$h_k(x) = \frac{1}{1 + \exp(-x^T \beta_k)}$$



# Statistical Models – Function Approximation

- Approximating  $f_\theta$  by minimizing the residual sum of squares

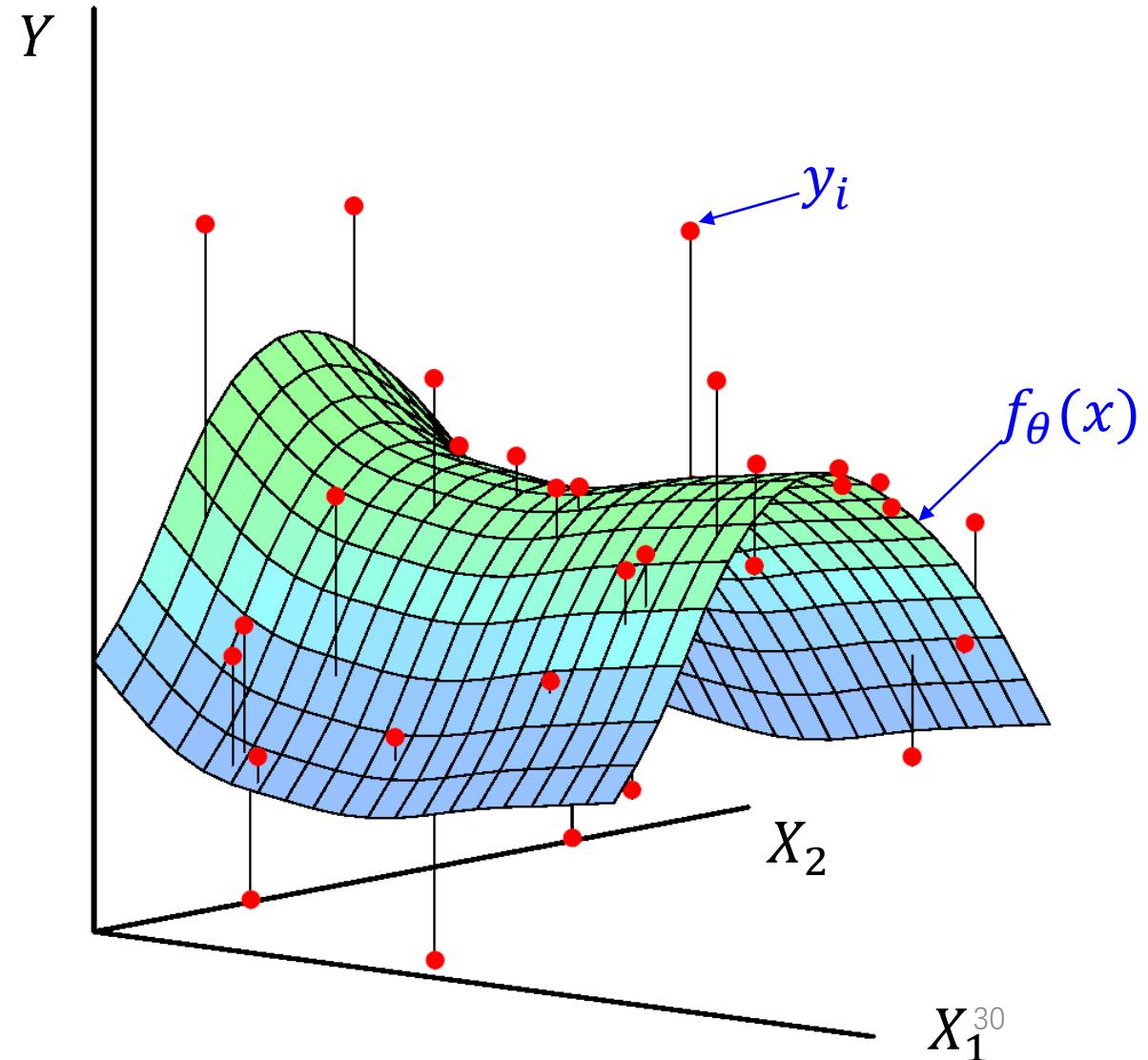
$$\text{RSS}(\theta) = \sum_{i=1}^N (y_i - f_\theta(x_i))^2$$

- Intuition

- $f$  surface in  $(p + 1)$  –space
- Observe noisy realizations
- Want **fitted surface as close to the observed points as possible**
- Distance measured by RSS

- Methods

- **Closed form**: if basis function have no hidden parameters
- **Iterative**: otherwise



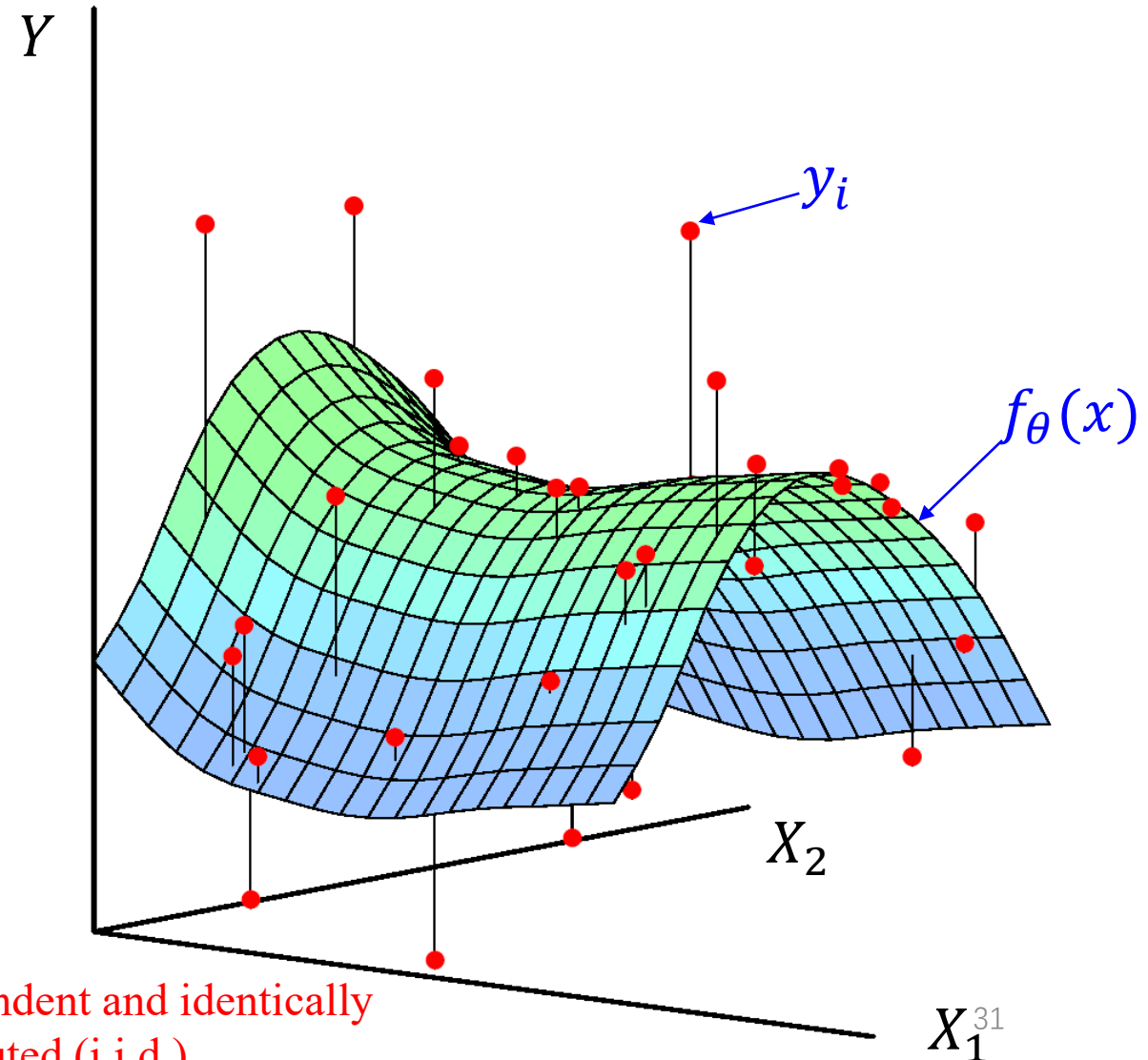
# Statistical Models – Function Approximation

- Approximating  $f_\theta$  by **maximum likelihood estimation (MLE)**
- Assume an independently drawn random sample  $y_i, i = 1, \dots, N$  from a probability density  $\text{Pr}_\theta(y)$ .
- The log-probability of observing the sample is

$$L(\theta) = \sum_{i=1}^N \log \text{Pr}_\theta(y_i)$$

$$\begin{aligned} L(\theta) &= \log \text{Pr}_\theta(\underline{y_1, y_2, \dots, y_N}) \\ &= \log \prod_{i=1}^N \text{Pr}_\theta(y_i) \end{aligned}$$

independent and identically distributed (i.i.d.)



# Statistical Models – Function Approximation

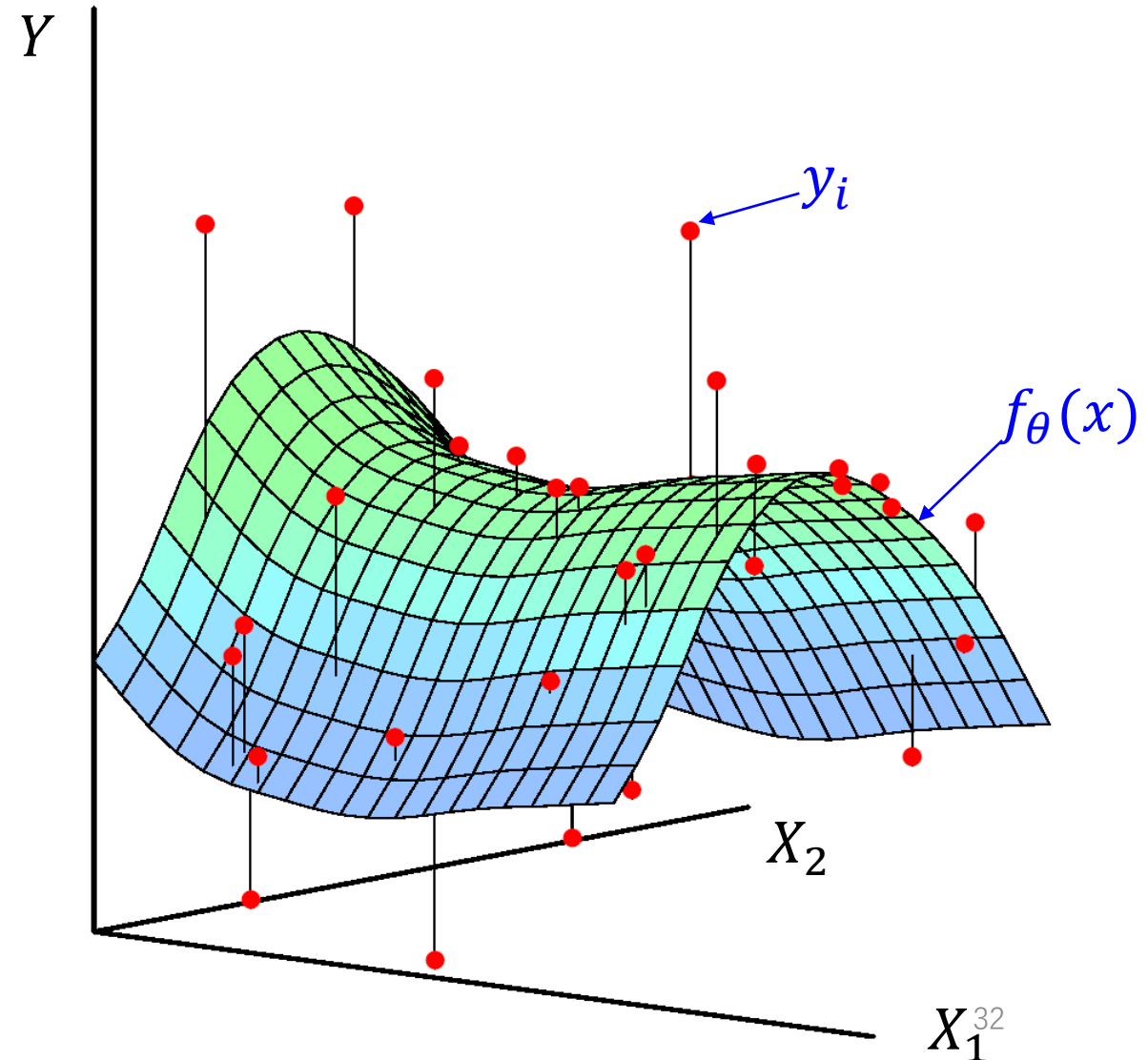
- Approximating  $f_\theta$  by **maximum likelihood estimation (MLE)**
- Assume an independently drawn random sample  $y_i, i = 1, \dots, N$  from a probability density  $\text{Pr}_\theta(y)$ .
- The log-probability of observing the sample is

$$L(\theta) = \sum_{i=1}^N \log \text{Pr}_\theta(y_i)$$

- **Set  $\theta$  to maximize  $L(\theta)$**

## Intuition:

Under the assumed statistical model, the observed data is most probable.





# Statistical Models – Function Approximation

- Approximating  $f_\theta$  by **maximum likelihood estimation (MLE)**
- Assume an independently drawn random sample  $y_i, i = 1, \dots, N$  from a probability density  $\Pr_\theta(y)$ .
- The log-probability of observing the sample is

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

- **Set  $\theta$  to maximize  $L(\theta)$**

$$\Pr_\theta(y|X = x) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{1}{2}\left(\frac{y - f_\theta(x)}{\sigma}\right)^2\right)$$

- **Least squares with additive error model**

$$Y = f_\theta(X) + \varepsilon$$
$$\varepsilon \sim \mathcal{N}(0, \sigma^2)$$

is **equivalent to maximum likelihood** with the conditional likelihood

$$\Pr_\theta(Y|X) = \mathcal{N}(f_\theta(X), \sigma^2)$$

- This is, because in this case the *log-likelihood* of data is

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

# Statistical Models – Function Approximation

- Approximating  $f_\theta$  by **maximum likelihood estimation (MLE)**
- Assume an independently drawn random sample  $y_i, i = 1, \dots, N$  from a probability density  $\text{Pr}_\theta(y)$ .
- The log-probability of observing the sample is

$$L(\theta) = \sum_{i=1}^N \log \text{Pr}_\theta(y_i)$$

- Set  $\theta$  to maximize  $L(\theta)$**

$$\text{argmax}_\theta L(\theta) = \text{argmin}_\theta \text{RSS}(\theta) = \text{argmin}_\theta \sum_{i=1}^N (y_i - f_\theta(x_i))^2$$

- Least squares with additive error model**

$$Y = f_\theta(X) + \varepsilon$$
$$\varepsilon \sim \mathcal{N}(0, \sigma^2)$$

is **equivalent to maximum likelihood** with the conditional likelihood

$$\text{Pr}_\theta(Y|X) = \mathcal{N}(f_\theta(X), \sigma^2)$$

- This is, because in this case the *log-likelihood* of data is

$$L(\theta) = -\frac{N}{2} \log(2\pi) - N \log \sigma$$

Proportional to RSS

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

# Overview of Supervised Learning II

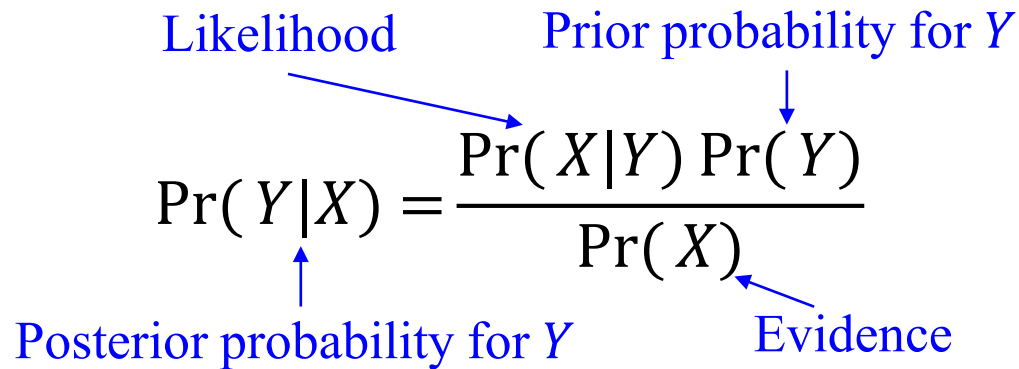
--- Bayesian Methods and Roughness Penalty

# Bayesian Methods and Roughness Penalty

- Bayesian methods
- Formula for joint probabilities

$$\begin{aligned}\Pr(X, Y) &= \Pr(Y|X) \Pr(X) \\ &= \Pr(X|Y) \Pr(Y)\end{aligned}$$

- Bayes's theorem



The diagram shows the formula  $\Pr(Y|X) = \frac{\Pr(X|Y) \Pr(Y)}{\Pr(X)}$  with blue arrows pointing to each term from a label. 'Likelihood' points to  $\Pr(X|Y)$ , 'Prior probability for Y' points to  $\Pr(Y)$ , 'Posterior probability for Y' points to  $\Pr(Y|X)$ , and 'Evidence' points to  $\Pr(X)$ .

$$\Pr(Y|X) = \frac{\Pr(X|Y) \Pr(Y)}{\Pr(X)}$$

$$\text{Posterior} \propto \text{Likelihood} \times \text{Prior}$$

- RSS is penalized with a roughness penalty

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

- $J(f)$  is large for ragged functions
  - E.g. cubic smoothing spline is the solution for the least-squares problem

$$\begin{aligned}\text{PRSS}(f; \lambda) &= \sum_{i=1}^N (y_i - f(x_i))^2 \\ &\quad + \lambda \int [f''(x)]^2 dx\end{aligned}$$

- Large second derivative is penalized

# Bayesian Methods and Roughness Penalty

- Introducing penalty functions is a type of **regularization**
  - It works against **overfitting**
  - It implements beliefs about unseen parts of the problem
- In a **Bayesian** framework
  - Penalty  $J$  is the **log-prior** (probability distribution)
  - PRSS is the **log-posterior** (probability distribution)
- RSS is penalized with a **roughness penalty**
$$\text{PRSS}(f ; \lambda) = \text{RSS}(f) + \lambda J(f)$$
- $J(f)$  is large for ragged functions
  - E.g. **cubic smoothing spline** is the solution for the least-squares problem
$$\text{PRSS}(f ; \lambda) = \sum_{i=1}^N (y_i - f(x_i))^2 + \lambda \int [f''(x)]^2 dx$$
  - Large second derivative is penalized

$$\text{Posterior} \propto \text{Likelihood} \times \text{Prior}$$

# Overview of Supervised Learning II

--- Model Selection

# Model Selection

- Smoothing and complexity parameters
  - Coefficient of the penalty term
  - Width of the kernel
  - Number of basis functions
- The setting of the parameters implements a trade-off between bias and variance

- Example:  $k$ -NN methods

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

$$E(\varepsilon) = 0$$

$$\text{Var}(\varepsilon) = \sigma^2$$

- Generalization error

$$\begin{aligned} \text{EPE}_k(x_0) &= E[Y - \hat{f}_k(x_0) | X = x_0] \\ &= \sigma^2 + [\text{Bias}^2(\hat{f}_k(x_0)) + \text{Var}_{\mathcal{T}}(\hat{f}_k(x_0))] \\ &= \underbrace{\sigma^2}_{\text{irreducible error}} + \underbrace{\left[ f(x_0) - \frac{1}{k} \sum_{\ell=1}^k f(x_{(\ell)}) \right]^2 + \frac{\sigma^2}{k}}_{\text{mean-square error}} \end{aligned}$$

