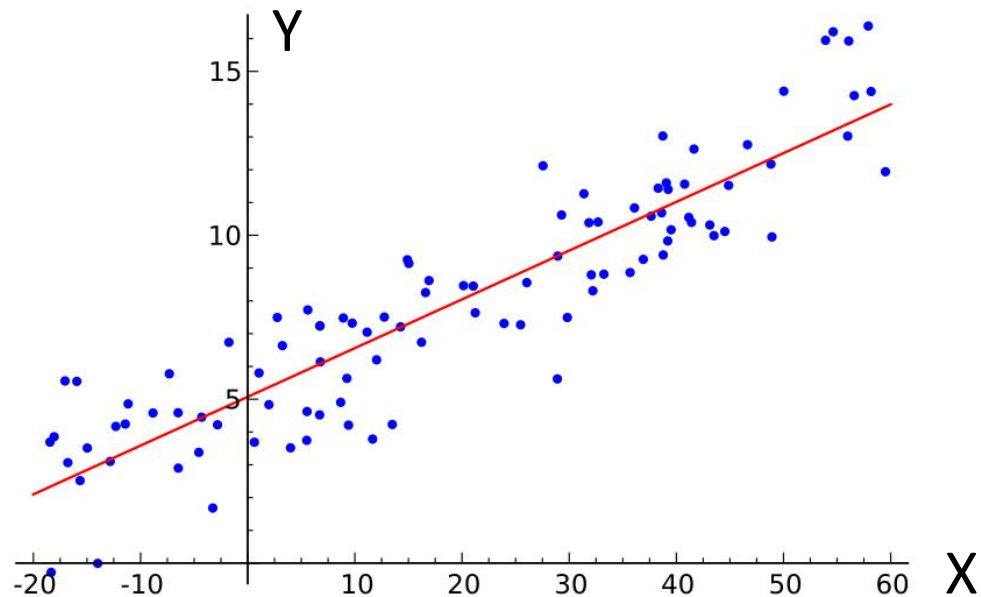# Linear Regression and the Bias Variance Tradeoff

Lecture by Shangsong Liang
Sun Yat-sen University

Thanks to Joseph E. Gonzalez, Tom Mitchell

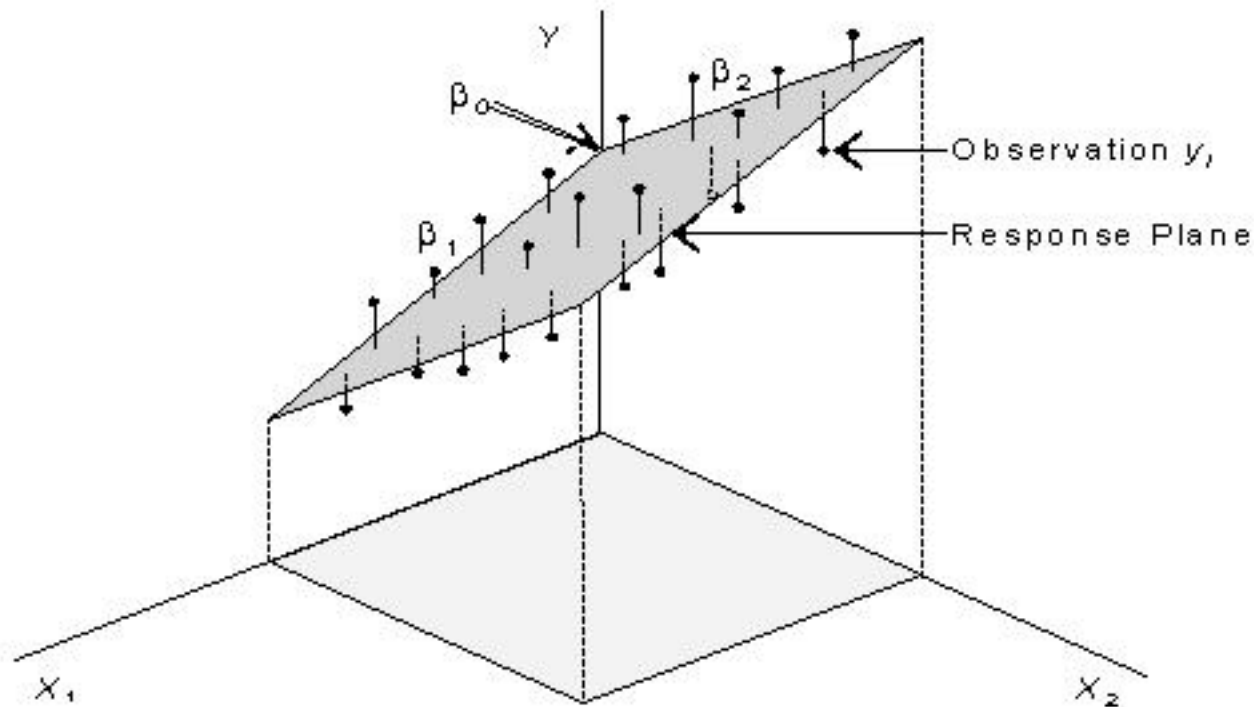# Simple Linear Regression

Linear Model:

Response Variable

Covariate

$$Y = mX + b$$
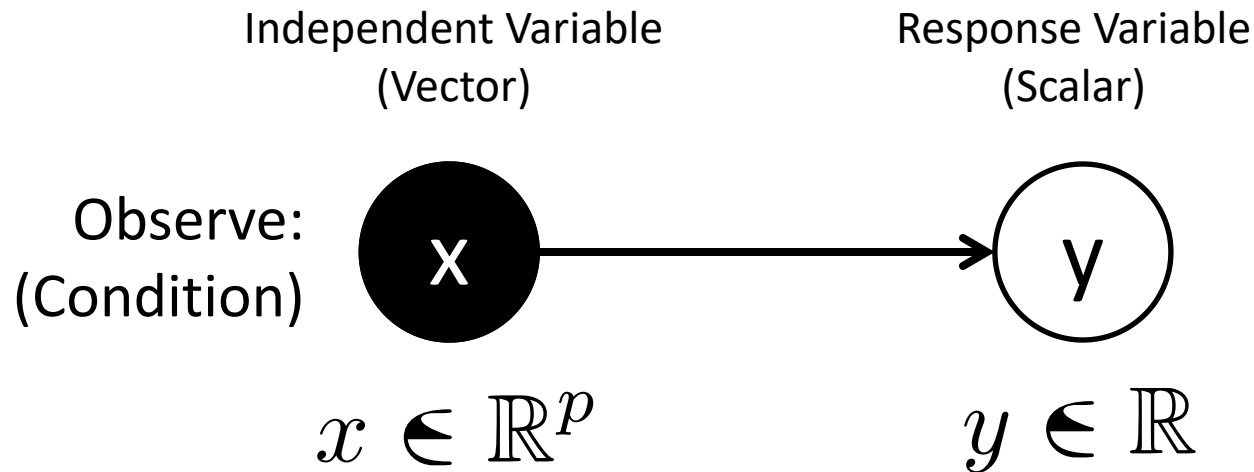
Slope

Intercept (bias)

Motivation

- One of the most widely used techniques
- Fundamental to many larger models
  - Generalized Linear Models
  - Collaborative filtering
- Easy to interpret
- Efficient to solve

Multiple Linear Regression

The Regression Model

- For a *single* data point *(x,y)*:

Independent Variable
(Vector)

Response Variable
(Scalar)

Observe:
(Condition)

$$x \in \mathbb{R}^p \qquad\qquad y \in \mathbb{R}$$

- Joint Probability:

$$p(x, y) = p(x) \boxed{p(y|x) \qquad \text{Discriminative Model}}$$

The Linear Model

Vector of
Parameters

Vector of
Covariates

Scalar
Response

$$y = \theta^T x + \epsilon$$

Real Value
Noise

$+ \; b$

Linear Combination
of Covariates

$$\sum_{i=1}^{p} \theta_i x_i$$

Noise Model:
$$\epsilon \sim N(0, \sigma^2)$$

What about bias/intercept term?

Define: $x_{p+1} = 1$

Then redefine p := p+1 for notational simplicity

Conditional Likelihood p(y|x)

- Conditioned on x:

Constant

$$y = \theta^T x + \boxed{\epsilon \sim N(0, \sigma^2)}$$

Normal Distribution

Mean   Variance

- Conditional distribution of Y:

$$Y \sim N(\theta^T x, \sigma^2)$$

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

Parameters and Random Variables

Parameters

$$y \sim N(\theta^T x, \sigma^2)$$

- Conditional distribution of y:
  - Bayesian: parameters as random variables

$$p(y|x, \theta, \sigma^2)$$

  - Frequentist: parameters as (unknown) constants

$$p_{\theta, \sigma^2}(y|x)$$

So far …

Y

X₂

X₁

*

I'm lonely
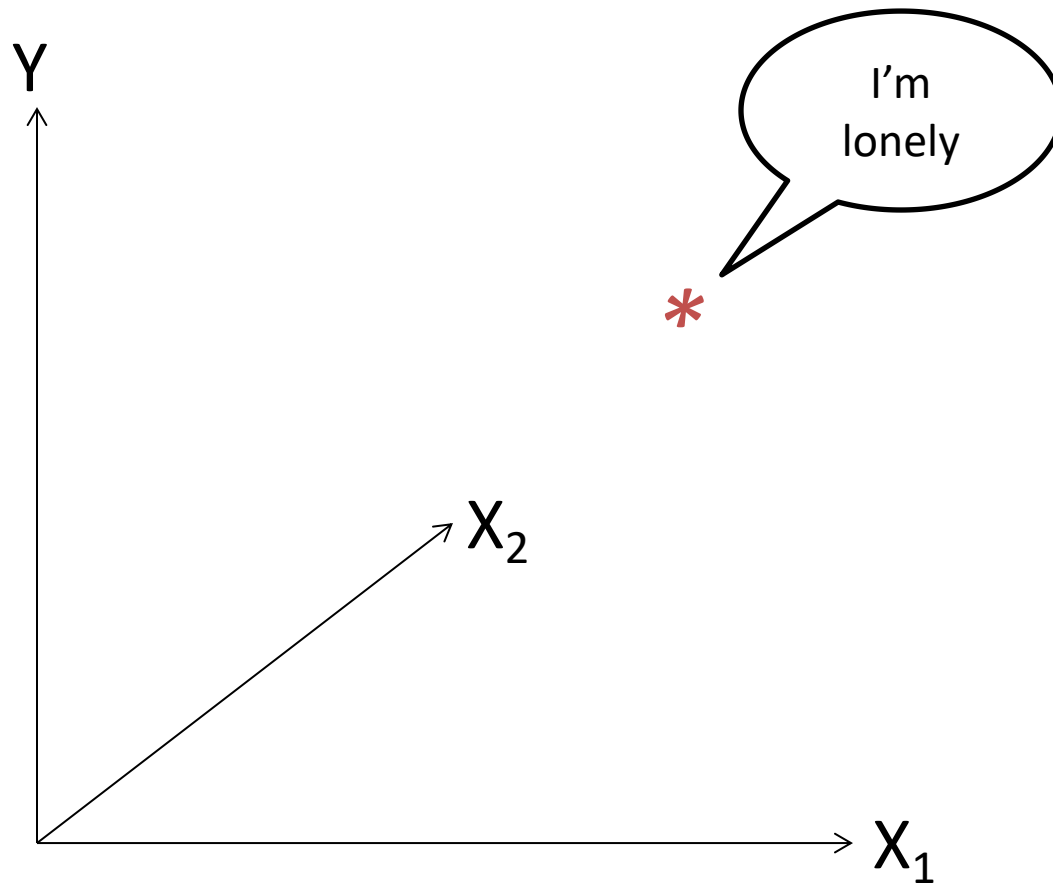
Independent and Identically Distributed (iid) Data

- For n data points:

$$\mathcal{D} = \{(x_1, y_1), \ldots, (x_n, y_n)\}$$
$$= \{(x_i, y_i)\}_{i=1}^{n}$$

Plate Diagram

Independent Variable (Vector)

Response Variable (Scalar)

$x_i \in \mathbb{R}^p$

$y_i \in \mathbb{R}$

$i \in \{1, \ldots, n\}$
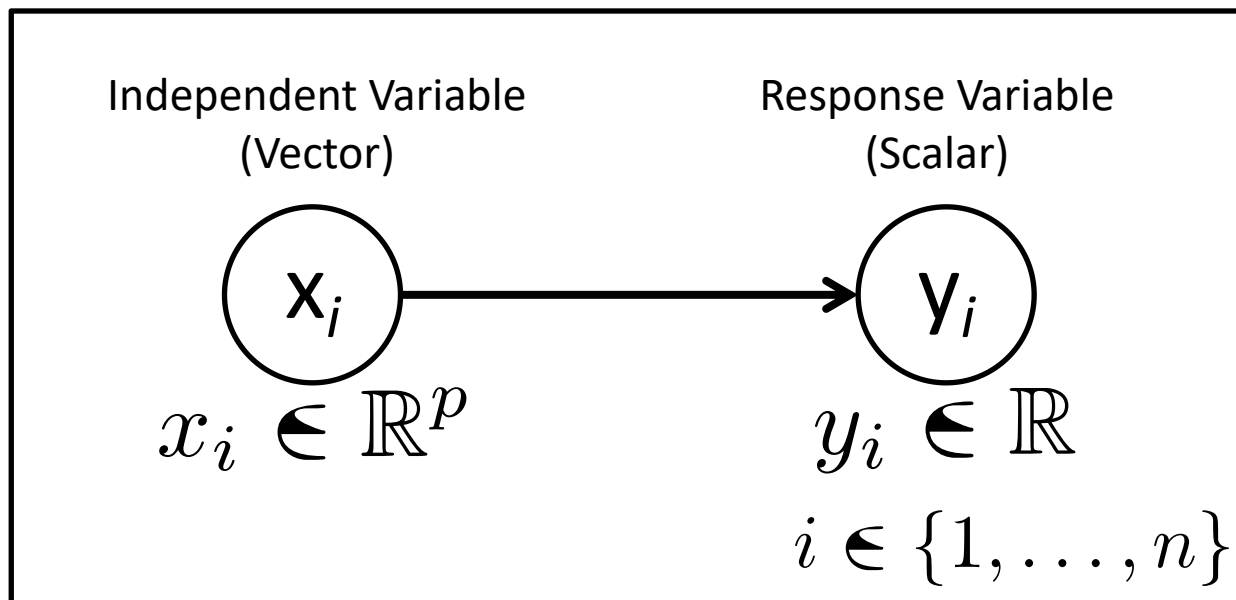
Joint Probability
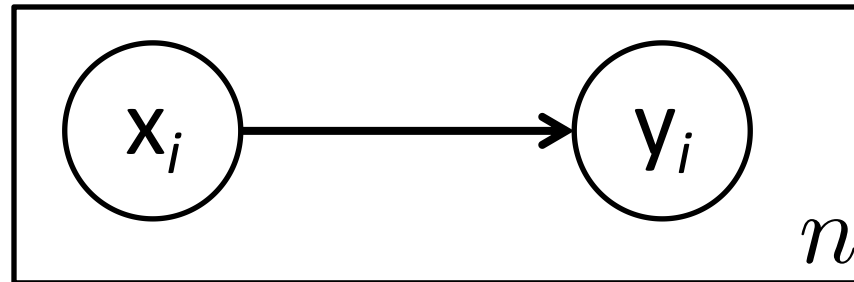
- For *n* data points **independent and identically distributed (iid)**:

$$p(\mathcal{D}) = \prod_{i=1}^{n} p(x_i, y_i)$$

$$= \prod_{i=1}^{n} p(x_i) p(y_i | x_i)$$

Rewriting with Matrix Notation

- Represent data $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{n}$ as:

Covariate (Design) Matrix

$$X = \begin{bmatrix} — x_1 — \\ — x_2 — \\ \cdots \\ — x_n — \end{bmatrix} \in \mathbb{R}^{np}$$

n

p

Assume X has rank p
(not degenerate)

Response Vector

$$Y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} \in \mathbb{R}^{n}$$

n

1

Rewriting with Matrix Notation

- Rewriting the model using matrix operations:

$$Y = X\theta + \boldsymbol{\epsilon}$$

Estimating the Model

- Given data how can we estimate θ?

$$Y = X\theta + \epsilon$$

- Construct maximum likelihood estimator (MLE):
  - Derive the log-likelihood
  - Find θ_{MLE} that maximizes log-likelihood
    - Analytically: Take derivative and set = 0
    - Iteratively: (Stochastic) gradient descent

Joint Probability
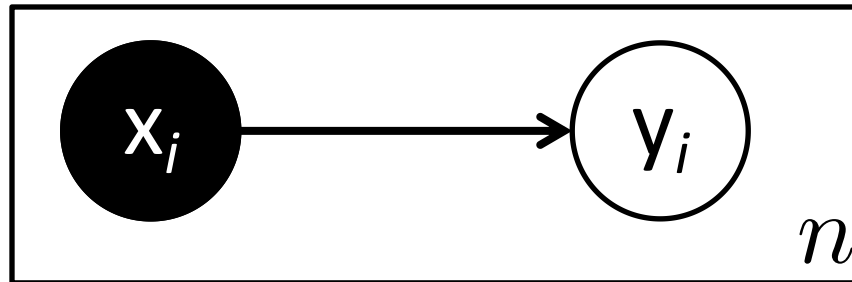
- For *n* data points:

$$p(\mathcal{D}) = \prod_{i=1}^{n} p(x_i, y_i)$$

$$= \prod_{i=1}^{n} p(x_i) p(y_i|x_i)$$

"1"

Discriminative Model

Defining the Likelihood

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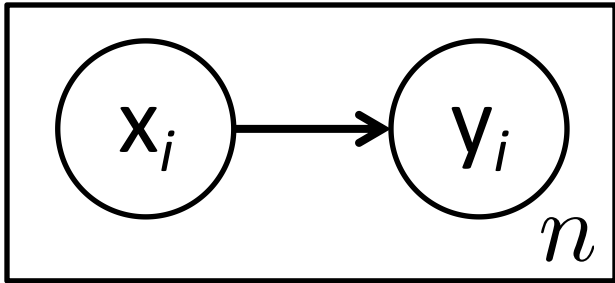
$$\frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(y - \theta^T x)^2}{2\sigma^2}\right)$$

$$\mathcal{L}(\theta|\mathcal{D}) = \prod_{i=1}^{n} p_\theta(y_i|x_i)$$

$$= \prod_{i=1}^{n} \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(y_i - \theta^T x_i)^2}{2\sigma^2}\right)$$

$$= \frac{1}{\sigma^n (2\pi)^{\frac{n}{2}}} \exp\left(-\frac{1}{2\sigma^2} \sum_{i=1}^{n} (y_i - \theta^T x_i)^2\right)$$

Maximizing the Likelihood

- Want to compute:

$$\hat{\theta}_{\mathrm{MLE}} = \arg\max_{\theta \in \mathbb{R}^p} \mathcal{L}(\theta | \mathcal{D})$$

- To simplify the calculations we take the log:

$$\hat{\theta}_{\mathrm{MLE}} = \arg\max_{\theta \in \mathbb{R}^p} \boxed{\log} \mathcal{L}(\theta | \mathcal{D})$$

which does not affect the maximization because log is a monotone function.

$$\mathcal{L}(\theta|\mathcal{D}) = \frac{1}{\sigma^n (2\pi)^{\frac{n}{2}}} \exp\left(-\frac{1}{2\sigma^2} \sum_{i=1}^{n} (y_i - \theta^T x_i)^2\right)$$

- Take the log:

$$\log \mathcal{L}(\theta|\mathcal{D}) = -\log(\sigma^n (2\pi)^{\frac{n}{2}}) - \frac{1}{2\sigma^2} \sum_{i=1}^{n} (y_i - \theta^T x_i)^2$$

- Removing constant terms with respect to θ:

$$\log \mathcal{L}(\theta) = -\sum_{i=1}^{n} (y_i - \theta^T x_i)^2$$

Monotone Function
(Easy to maximize)

$$\log \mathcal{L}(\theta) = - \sum_{i=1}^{n} (y_i - \theta^T x_i)^2$$
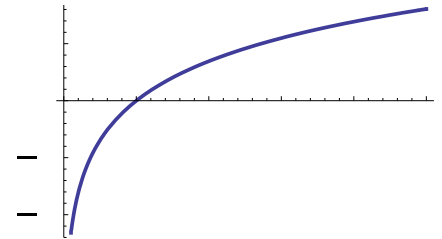
- Want to compute:

$$\hat{\theta}_{\mathrm{MLE}} = \arg \max_{\theta \in \mathbb{R}^p} \log \mathcal{L}(\theta | \mathcal{D})$$

- Plugging in log-likelihood:

$$\hat{\theta}_{\mathrm{MLE}} = \arg \max_{\theta \in \mathbb{R}^p} - \sum_{i=1}^{n} (y_i - \theta^T x_i)^2$$

$$\hat{\theta}_{\mathrm{MLE}} = \arg\max_{\theta \in \mathbb{R}^p} -\sum_{i=1}^{n}(y_i - \theta^T x_i)^2$$

- Dropping the sign and flipping from maximization to minimization:

$$\hat{\theta}_{\mathrm{MLE}} = \arg\min_{\theta \in \mathbb{R}^p} \sum_{i=1}^{n}(y_i - \theta^T x_i)^2$$

Minimize Sum (Error)^2

- Gaussian Noise Model → Squared Loss
  - Least Squares Regression

Pictorial Interpretation of Squared Error
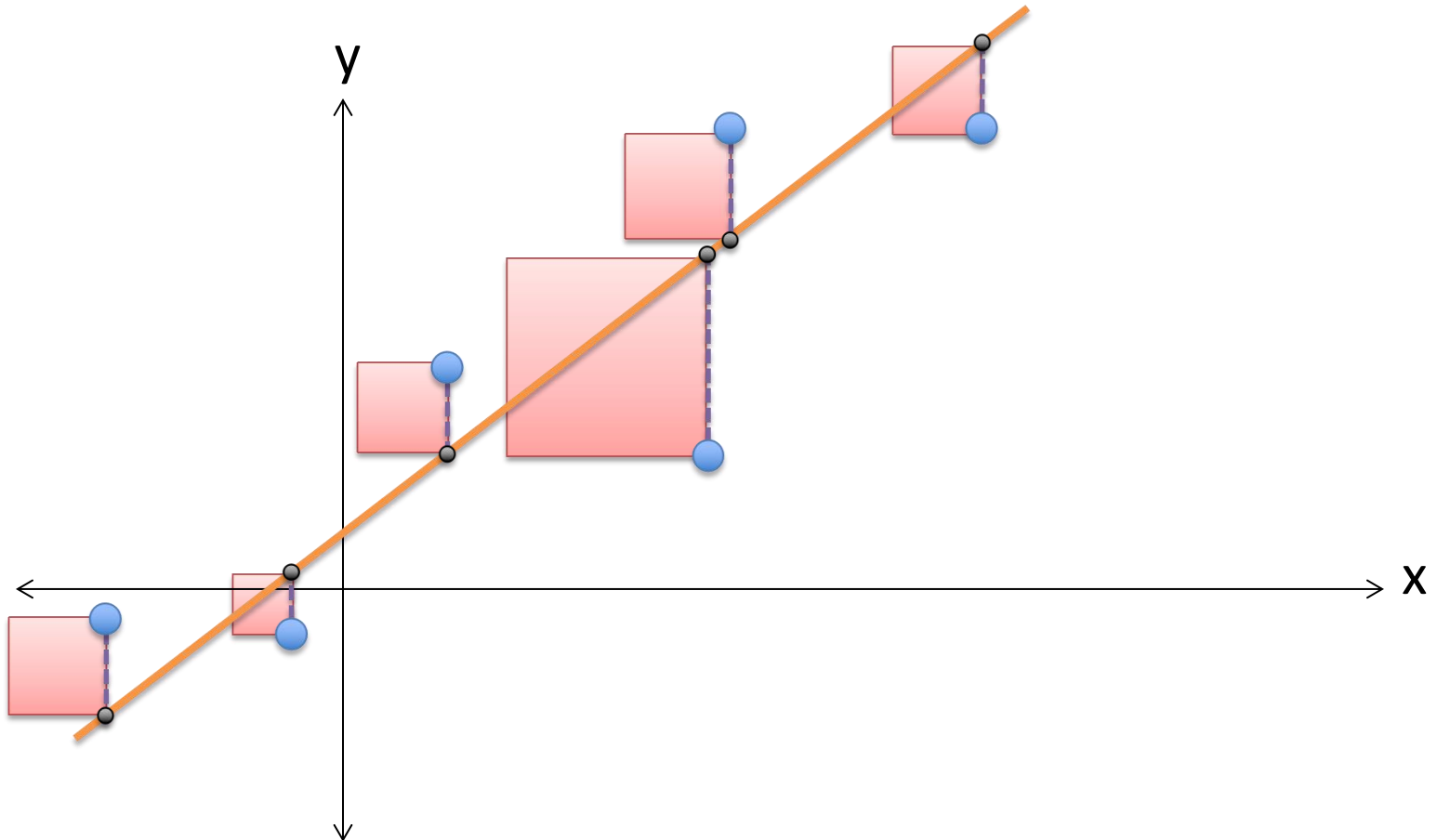
Maximizing the Likelihood (Minimizing the Squared Error)

$$\hat{\theta}_{\mathrm{MLE}} = \arg \min_{\theta \in \mathbb{R}^p} \sum_{i=1}^{n} (y_i - \theta^T x_i)^2$$

Convex Function

$-\log \mathcal{L}(\theta)$

Slope = 0

θ

$\hat{\theta}_{\mathrm{MLE}}$
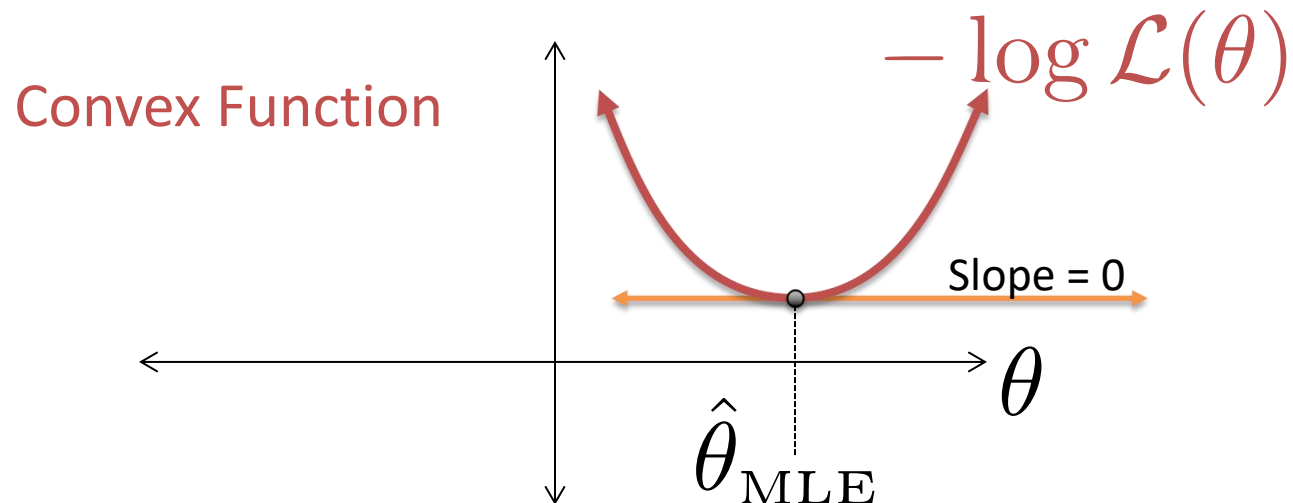
- Take the gradient and set it equal to zero

Minimizing the Squared Error

$$\hat{\theta}_{\mathrm{MLE}} = \arg \min_{\theta \in \mathbb{R}^p} \sum_{i=1}^{n} (y_i - \theta^T x_i)^2$$

- Taking the gradient

$$-\nabla_\theta \log \mathcal{L}(\theta) = \nabla_\theta \sum_{i=1}^{n} (y_i - \theta^T x_i)^2$$

Chain Rule →

$$= -2 \sum_{i=1}^{n} (y_i - \theta^T x_i) x_i$$

$$= -2 \sum_{i=1}^{n} y_i x_i + 2 \sum_{i=1}^{n} (\theta^T x_i) x_i$$

- Rewriting the gradient in matrix form:

$$-\nabla_\theta \log \mathcal{L}(\theta) = -2 \sum_{i=1}^{n} y_i x_i + 2 \sum_{i=1}^{n} (\theta^T x_i) x_i$$

$$= -2X^T Y + 2X^T X \theta$$

- To make sure the log-likelihood is convex compute the second derivative (Hessian)

$$-\nabla^2 \log \mathcal{L}(\theta) = 2X^T X$$

- If X is full rank then $X^T X$ is positive definite and therefore θ_{MLE} is the minimum
  - Address the degenerate cases with regularization

$$-\nabla_\theta \log \mathcal{L}(\theta) = -2X^T y + 2X^T X\theta = 0$$

- Setting gradient equal to 0 and solve for θ_{MLE}:

$$(X^T X)\hat{\theta}_{\text{MLE}} = X^T Y$$

$$\hat{\theta}_{\text{MLE}} = (X^T X)^{-1} X^T Y$$

Normal Equations
(Write on board)

Geometric Interpretation

- View the MLE as finding a projection on col(X)
  - Define the estimator:
  $$\hat{Y} = X\theta$$

  - Observe that Ŷ is in col(X)
    - linear combination of cols of X
  - Want to Ŷ closest to Y

- Implies (Y-Ŷ) normal to X
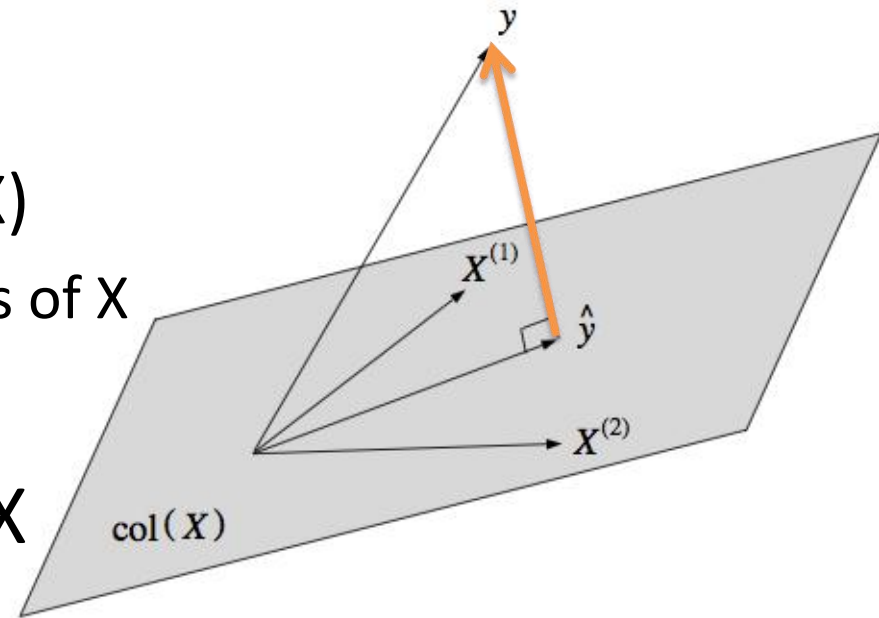
$$X^T(Y - \hat{Y}) = X^T(Y - X\theta) = 0$$
$$\Rightarrow X^T X\theta = X^T Y$$

Geometric Interpretation
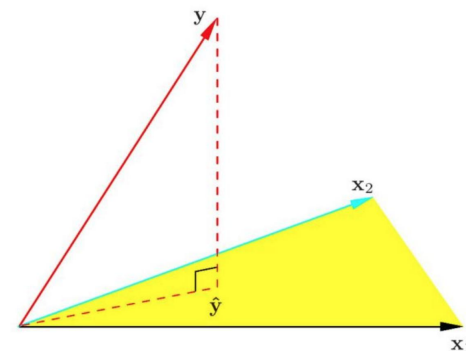
假设我们通过一个特征来估计 y ，依然有 5 条数据， $X = \begin{bmatrix} 1 & x_{11} \\ 1 & x_{21} \\ 1 & x_{31} \\ 1 & x_{41} \\ 1 & x_{51} \end{bmatrix}$ ， X 的两条列向量

$x_1 = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$ 和 $x_2 = \begin{bmatrix} x_{11} \\ x_{21} \\ x_{31} \\ x_{41} \\ x_{51} \end{bmatrix}$ 通过权重 $w = \begin{bmatrix} w_1 \\ w_2 \end{bmatrix}$ 张成了 2 维的子空间，也就是一个平面，我们

通过获得数据知道向量 y 。因为误差， y 不在 x_1 和 x_2 张成的平面上，怎么样才能找到最好的 w 呢？ 一种答案是找到 y 在 x_1 和 x_2 张成的平面上投影 $\hat{y} = X\hat{w}$ ，如上图所示， $\hat{w}$ 就是最好的 w 。

由投影的性质可知 $y - \hat{y}$ 垂直于 x_1 和 x_2 ，用人话说就是从 $\hat{y}$ 的终点到 y 的终点是一条垂直于 x_1 和 x_2 的向量，可得

Geometric Interpretation

$$(y - \hat{y})^T x_1 = (y - X\hat{w})^T x_1 = 0$$
$$(y - \hat{y})^T x_2 = (y - X\hat{w})^T x_2 = 0$$
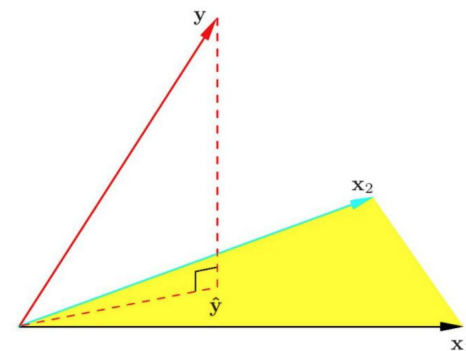
把上式合并，可得

$$(y - X\hat{w})^T X = 0$$
$$X^T (y - X\hat{w}) = 0$$
$$X^T y - X^T X\hat{w} = 0$$
$$X^T X\hat{w} = X^T y$$
$$\hat{w} = (X^T X)^{-1} X^T y$$

最小二乘法和投影矩阵是两种完全不同的思路，却得到了相同结论，究其原因，投影使 ŷ 与 y 的欧式距离最短，这与最小二乘法让误差项平方和最小的思想不谋而合，岂不妙哉？

Connection to Pseudo-Inverse

$$\hat{\theta}_{\mathrm{MLE}} = \underbrace{(X^T X)^{-1} X^T}_{} Y$$

Moore-Penrose Psuedoinverse $X^\dagger$

- Generalization of the inverse:
  - Consider the case when X is square and invertible:

$$X^\dagger = (X^T X)^{-1} X^T = X^{-1}(X^T)^{-1} X^T = X^{-1}$$

  - Which implies $\theta_{\mathrm{MLE}} = X^{-1} Y$ the solution to $X \theta = Y$ when X is square and invertible

Computing the MLE

$$\hat{\theta}_{\mathrm{MLE}} = (X^T X)^{-1} X^T Y$$

- **Not** typically solved by inverting X^TX

- Solved using direct methods:

  - Cholesky factorization:
    - Up to a factor of 2 faster
  - QR factorization:
    - More numerically stable

  or use the
  built-in solver
  in your math library.
  R: solve(Xt %*% X, Xt %*% y)

- Solved using various iterative methods:

  - Krylov subspace methods
  - (Stochastic) Gradient Descent

Cholesky Factorization

$$\text{solve } \hat{\theta}_{\mathrm{MLE}} \quad \underbrace{(X^T X)}_{C} \hat{\theta}_{\mathrm{MLE}} = \underbrace{X^T Y}_{d}$$

- Compute symm. matrix $C = X^T X$ $\qquad O(np^2)$

- Compute vector $d = X^T Y$ $\qquad O(np)$

- Cholesky Factorization $LL^T = C$ $\qquad O(p^3)$
  - L is lower triangular

- Forward subs. to solve: $Lz = d$ $\qquad O(p^2)$

- Backward subs. to solve: $L^T \hat{\theta}_{\mathrm{MLE}} = z$ $\qquad O(p^2)$

Connections to graphical model inference:
http://ssg.mit.edu/~willsky/publ_pdfs/185_pub_MLR.pdf and http://yaroslavvb.blogspot.com/2011/02/junction-trees-in-numerical-analysis.html with illustrations

Solving Triangular System

$$\begin{bmatrix} A_{11}x_1 & A_{12}x_2 & A_{13}x_3 & A_{14}x_4 \\ & A_{22} & A_{23} & A_{24} \\ & & A_{33} & A_{34} \\ & & & A_{44} \end{bmatrix} * \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \end{bmatrix}$$

Bonus Content

Solving Triangular System

$A_{11}x_1$	$A_{12}x_2$	$A_{13}x_3$	$A_{14}x_4$
	$A_{22}x_2$	$A_{23}x_3$	$A_{24}x_4$
		$A_{33}x_3$	$A_{34}x_4$
			$A_{44}x_4$

b_1

$$x_1 = \frac{b_1 - A_{12}x_2 - A_{13}x_3 - A_{14}x_4}{A_{11}}$$

b_2

$$x_2 = \frac{b_2 - A_{23}x_3 - A_{24}x_4}{A_{22}}$$

b_3

$$x_3 = \frac{(b_3 - A_{34}x_4)}{A_{33}}$$

b_4

$$x_4 = b_4 / A_{44}$$
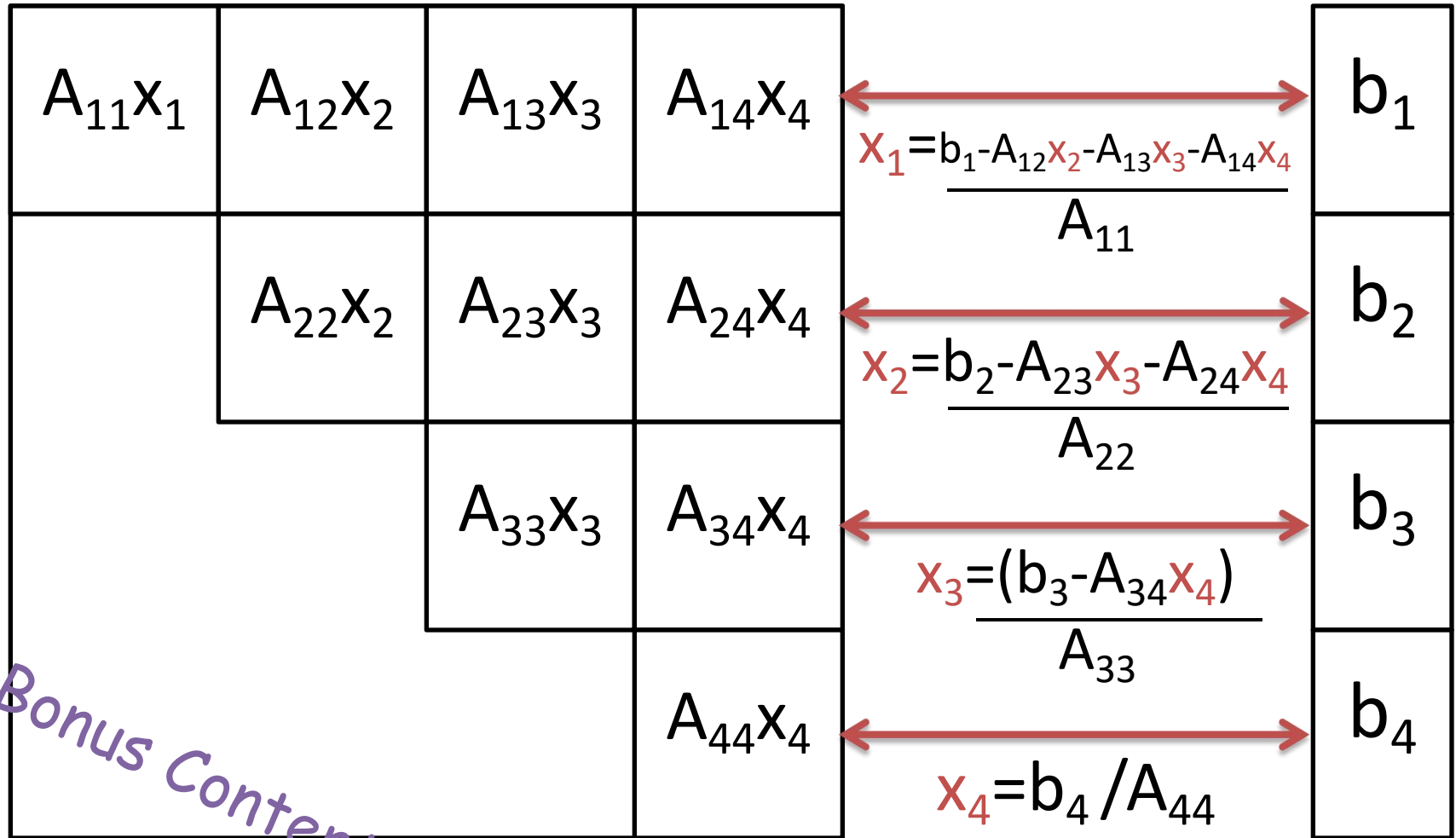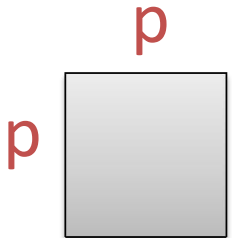
Bonus Content

Distributed Direct Solution (Map-Reduce)

$$\hat{\theta}_{\mathrm{MLE}} = (X^T X)^{-1} X^T Y$$

- Distribution computations of sums:

$$C = X^T X = \sum_{i=1}^{n} x_i x_i^T \qquad O(np^2)$$

$$d = X^T y = \sum_{i=1}^{n} x_i y_i \qquad O(np)$$

- Solve system $C\,\theta_{\mathrm{MLE}} = d$ on master. $\qquad O(p^3)$

Gradient Descent:
What if p is large? (e.g., n/2)

- The cost of O(np^2) = O(n^3) could by prohibitive

- Solution: Iterative Methods
  - Gradient Descent:

For τ from 0 until *convergence*

$$\theta^{(\tau+1)} = \theta^{(\tau)} - \rho(\tau)\nabla \log \mathcal{L}(\theta^{(\tau)}|D)$$

Learning rate

Gradient Descent Illustrated:
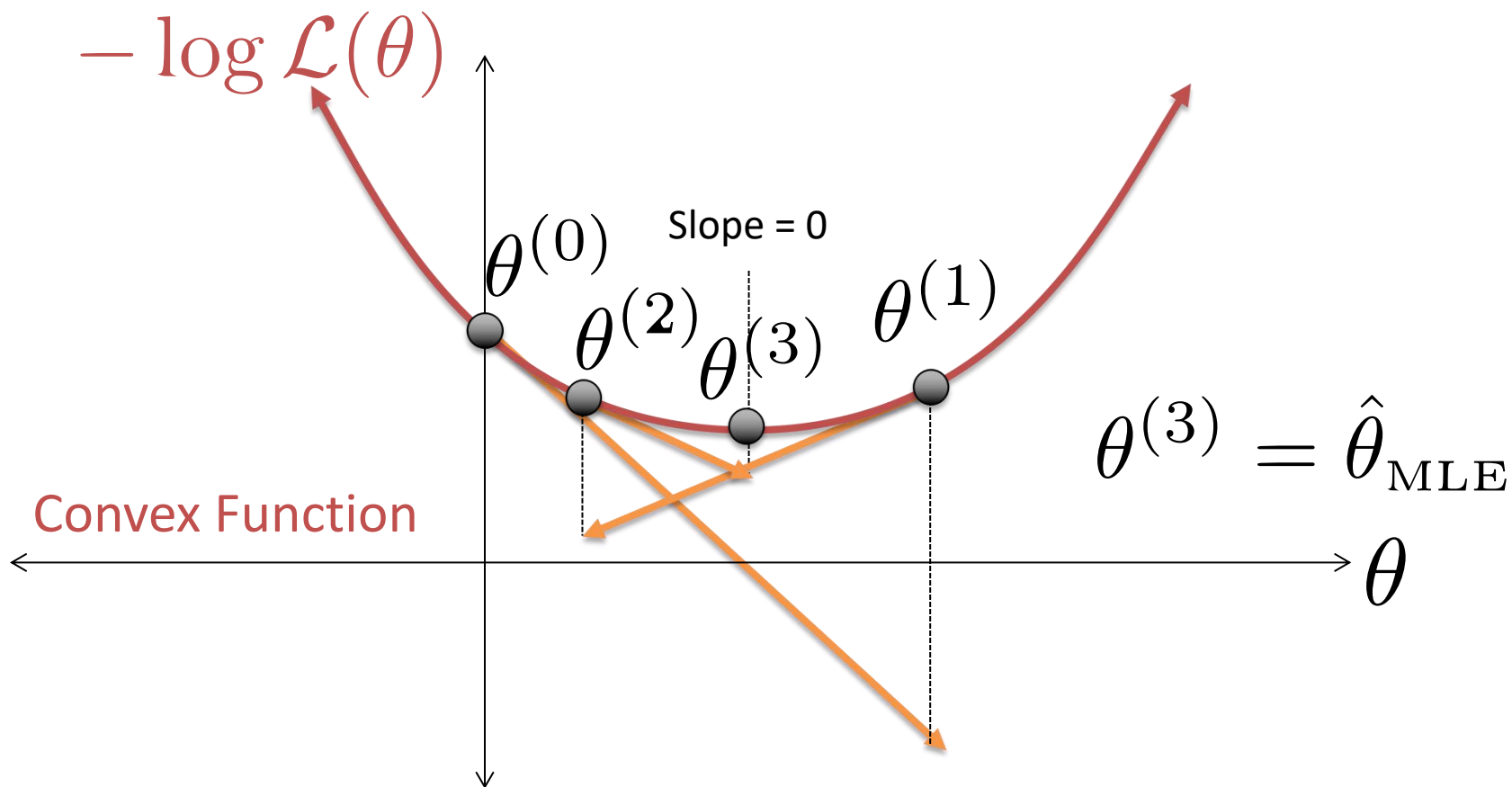
$-\log \mathcal{L}(\theta)$

$\theta^{(0)}$

Slope = 0

$\theta^{(2)}$ $\theta^{(3)}$ $\theta^{(1)}$

Convex Function

$\theta^{(3)} = \hat{\theta}_{\mathrm{MLE}}$

θ

Gradient Descent:
What if p is large? (e.g., n/2)

- The cost of O(np^2) = O(n^3) could by prohibitive

- Solution: Iterative Methods
  - Gradient Descent:

For τ from 0 until *convergence*

$$\theta^{(\tau+1)} = \theta^{(\tau)} - \rho(\tau)\nabla \log \mathcal{L}(\theta^{(\tau)}|D)$$

$$= \theta^{(\tau)} + \rho(\tau)\frac{1}{n}\sum_{i=1}^{n}(y_i - \theta^{(\tau)T}x_i)x_i \quad O(np)$$

Estimate of the Gradient

- Can we do better?

Stochastic Gradient Descent

- Construct noisy estimate of the gradient:

For τ from 0 until *convergence*
  1) pick a random i
  2) $\theta^{(\tau+1)} = \theta^{(\tau)} + \rho(\tau)(y_i - \theta^{(\tau)T}x_i)x_i$ $O(p)$

- Sensitive to choice of ρ(τ) typically (ρ(τ)=1/τ)

- Also known as Least-Mean-Squares (LMS)

- Applies to streaming data *O(p)* storage

Fitting Non-linear Data

- What if Y has a non-linear response?
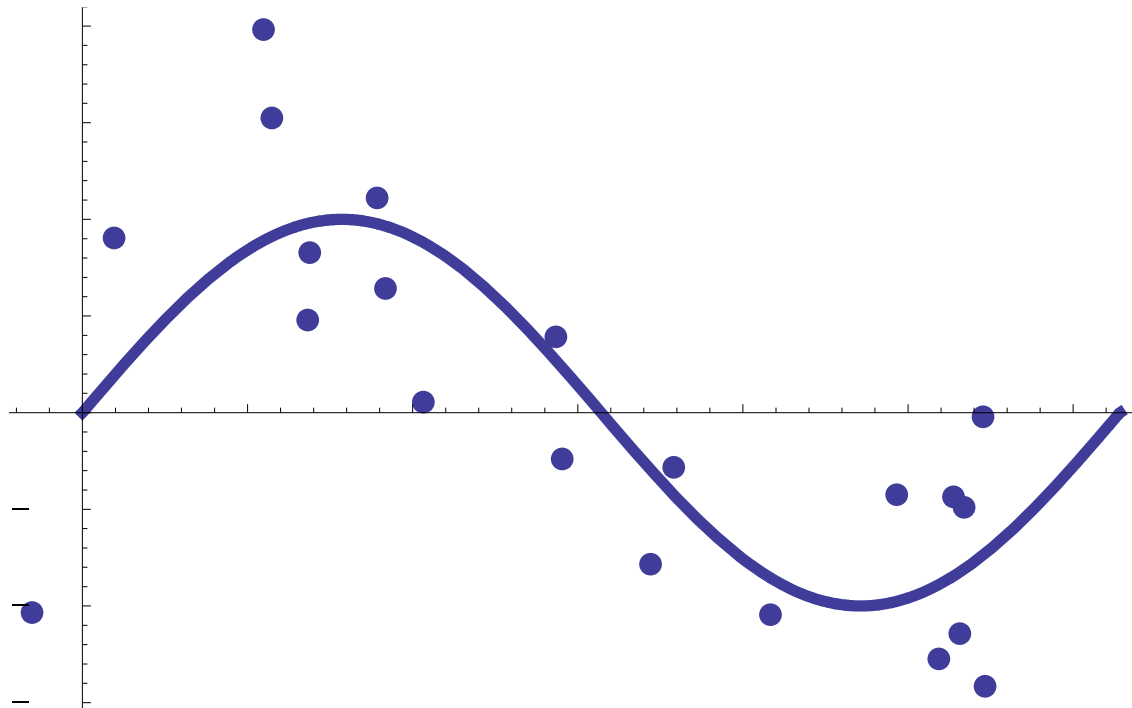
- Can we still use a linear model?

Transforming the Feature Space

- Transform features x_i

$$x_i = (X_{i,1}, X_{i,2}, \ldots, X_{i,p})$$

- By applying non-linear transformation ϕ:

$$\phi : \mathbb{R}^p \rightarrow \mathbb{R}^k$$

- Example:

$$\phi(x) = \{1, x, x^2, \ldots, x^k\}$$

  – others: splines, radial basis functions, …

  – Expert engineered features (modeling)

Under-fitting

{   }

{   }

{   }

{   }
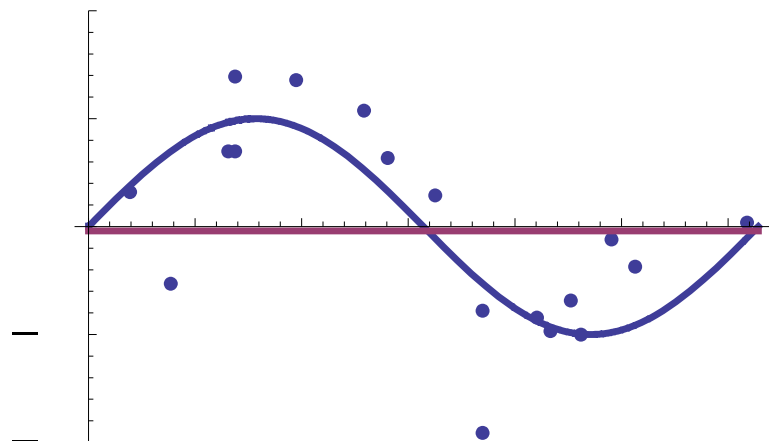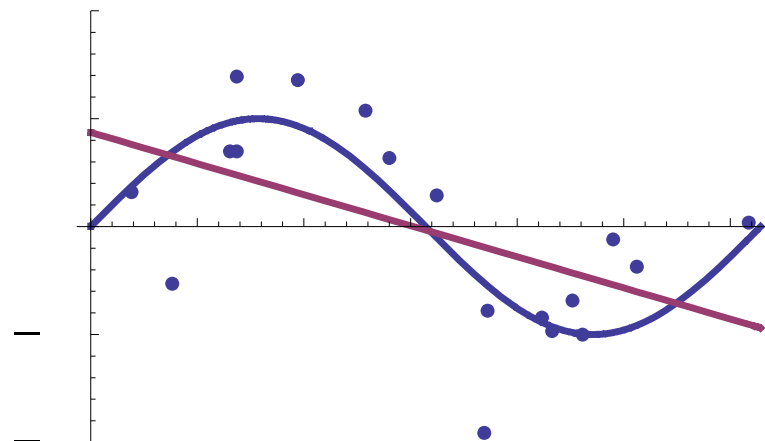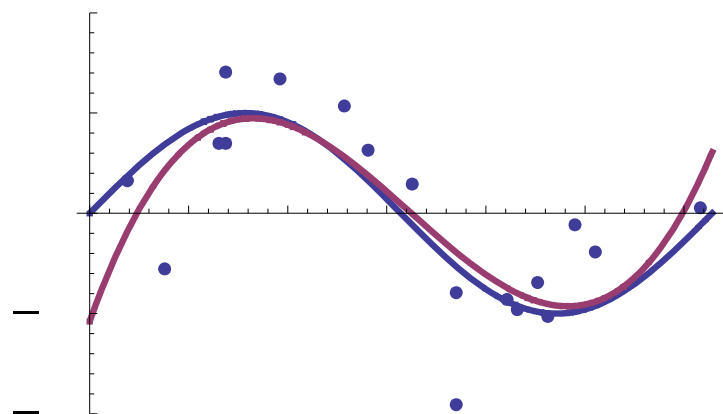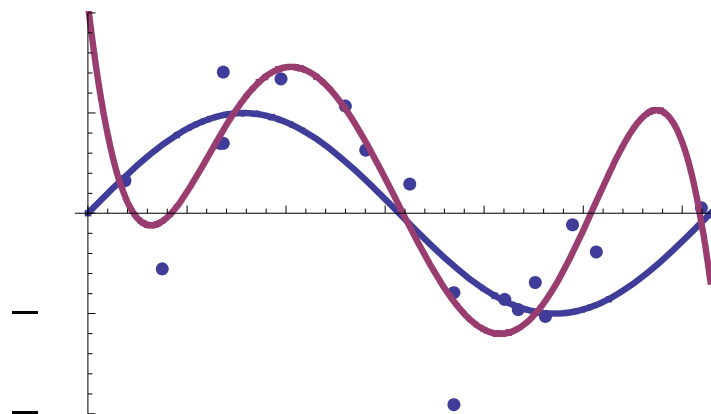
Over-fitting

Really Over-fitting!
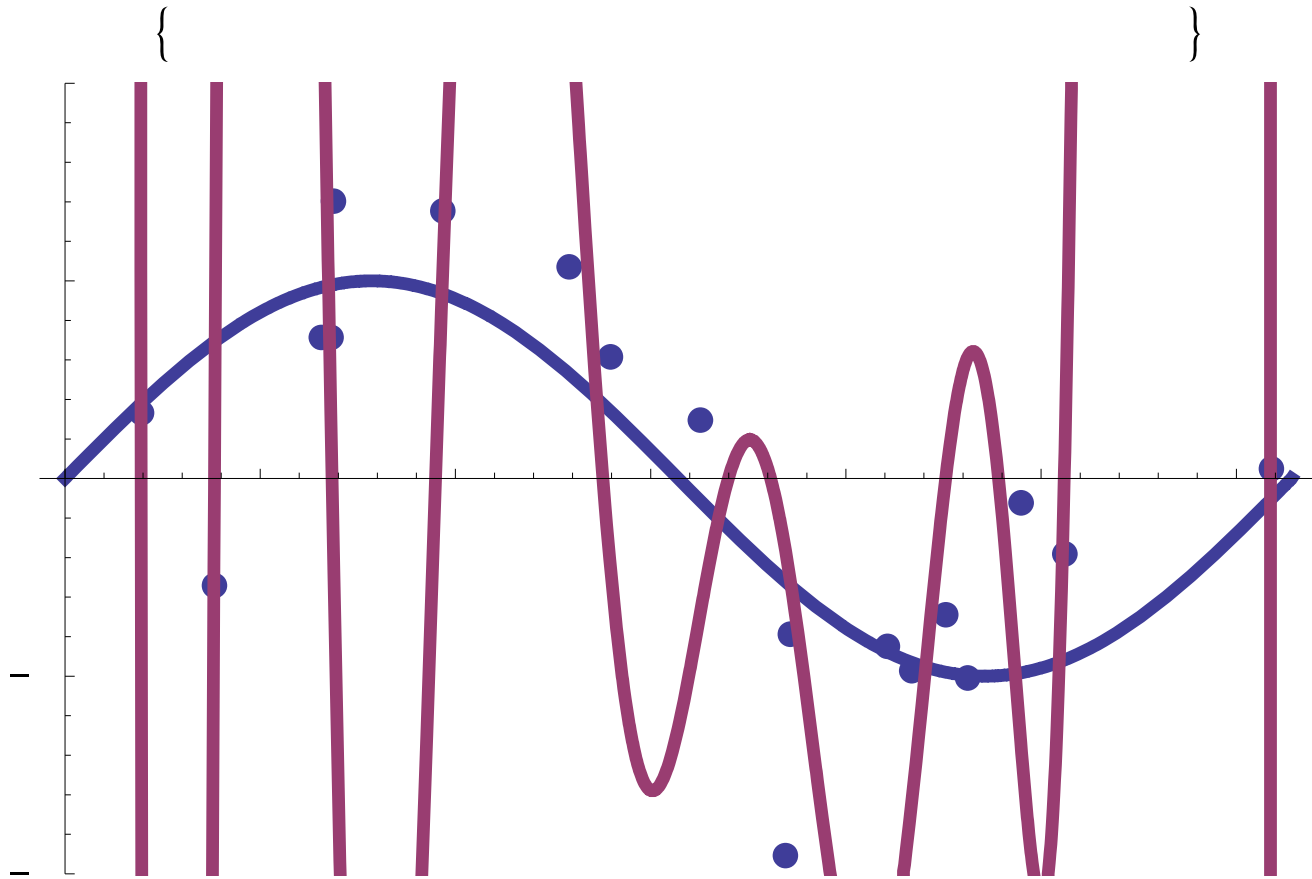
- Errors on training data are small
- But errors on new points are likely to be large

What if I train on different data?

Low Variability:
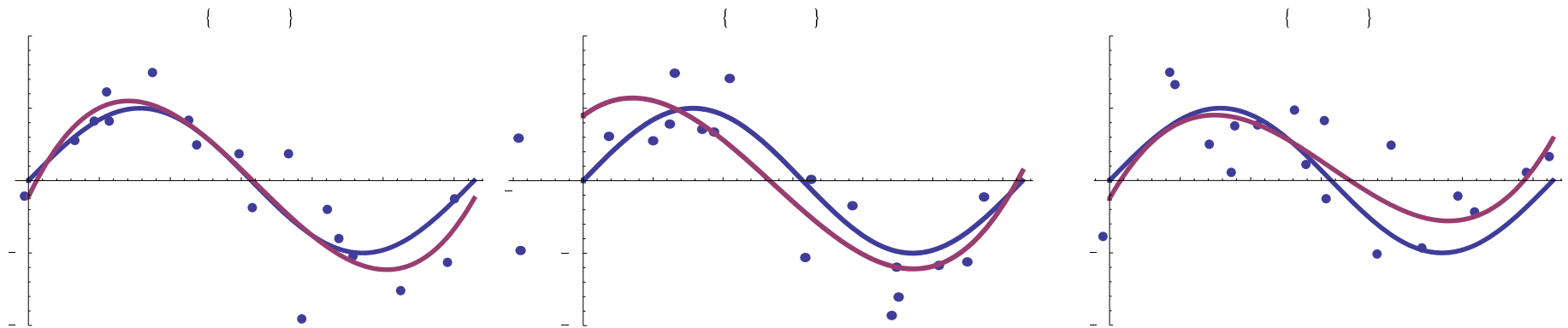
High Variability
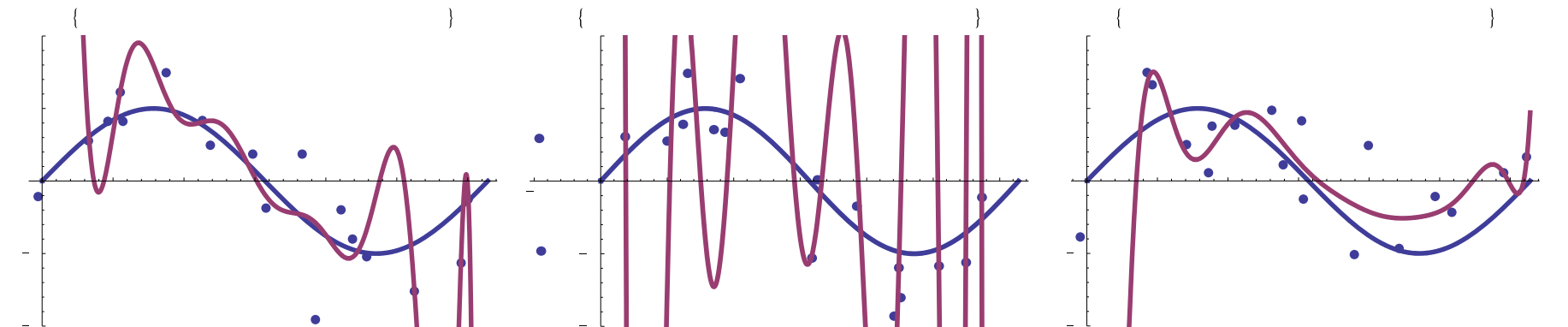
Bias-Variance Tradeoff

- So far we have minimized the error (loss) with respect to **training data**

  – Low training error does not imply good expected performance: **over-fitting**

- We would like to reason about the **expected loss (Prediction Risk)** over:

  – Training Data: $\{(y_1, x_1), ..., (y_n, x_n)\}$

  – Test point: (y_*, x_*)

- We will decompose the expected loss into:

$$\mathbf{E}_{D,(y_*,x_*)} \left[(y_* - f(x_*|D))^2 \right] = \text{Noise} + \text{Bias}^2 + \text{Variance}$$

Bias-Variance Tradeoff

- Bias (偏差)、Error (误差)、Variance (方差)
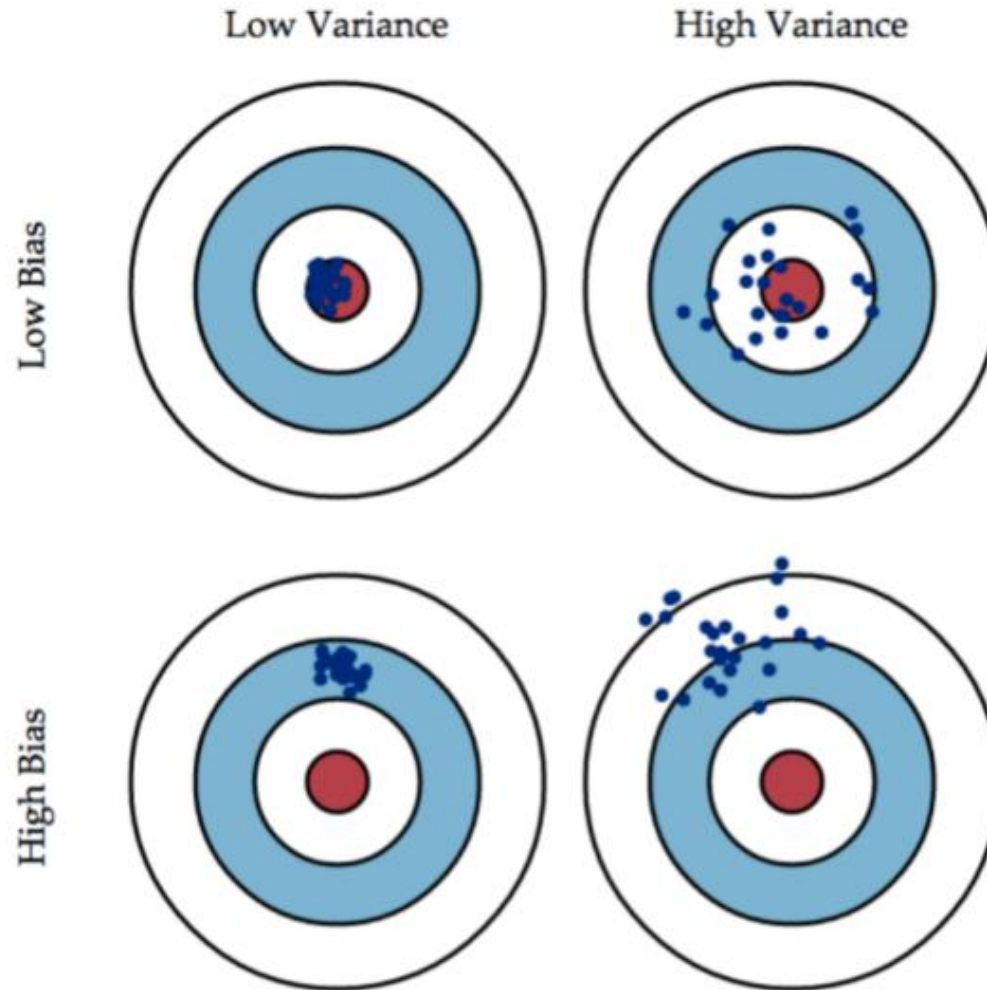
- Define (unobserved) the true model (*h*):

$$y_* = h(x_*) + \epsilon_*$$

Assume 0 mean noise
[bias goes in *h(x*)*]

- Completed the squares with: $h(x_*) = h_*$

$$\mathbf{E}_{D,(y_*,x_*)} \left[(y_* - f(x_*|D))^2 \right] \text{ Expected Loss}$$

$$= \mathbf{E}_{D,(y_*,x_*)} \left[(y_* \underbrace{- h(x_*)}_{a} + \underbrace{h(x_*) - f(x_*|D)}_{b})^2 \right]$$

$$(a+b)^2 = a^2 + b^2 + 2ab$$

$$= \mathbf{E}_{\epsilon_*} \left[(y_* - h(x_*))^2 \right] + \mathbf{E}_D \left[(h(x_*) - f(x_*|D))^2 \right]$$
$$+ 2\mathbf{E}_{D,(y_*,x_*)} \left[y_* h_* - y_* f_* - h_* h_* + h_* f_* \right]$$

- Define (unobserved) the true model (*h*):

$$y_* = h(x_*) + \epsilon_*$$

- Completed the squares with: $h(x_*) = h_*$

$$\mathbf{E}_{D,(y_*,x_*)}\left[(y_* - f(x_*|D))^2\right] \quad \text{Expected Loss}$$

$$= \mathbf{E}_{D,(y_*,x_*)}\left[(y_* - h(x_*) + h(x_*) - f(x_*|D))^2\right]$$

$$= \mathbf{E}_{\epsilon_*}\left[(y_* - h(x_*))^2\right] + \mathbf{E}_D\left[(h(x_*) - f(x_*|D))^2\right]$$

$$+ 2\mathbf{E}_{D,(y_*,x_*)}\left[y_* h_* - y_* f_* - h_* h_* + h_* f_*\right]$$

Substitute defn. $y_* = h_* + e_*$

$$\mathbf{E}\left[(h_* + \epsilon_*)h_* - (h_* + \epsilon_*)f_* - h_* h_* + h_* f_*\right] =$$

$$h_* h_* + \mathbf{E}[\epsilon_*]h_* - h_*\mathbf{E}[f_*] - \mathbf{E}[\epsilon_*]f_* - h_* h_* + h_*\mathbf{E}[f_*]$$

- Define (unobserved) the true model (h):

$$y_* = h(x_*) + \epsilon_*$$

- Completed the squares with: $h(x_*) = h_*$

$$\mathbf{E}_{D,(y_*,x_*)}\left[(y_* - f(x_*|D))^{\mathbf{2}}\right] \text{ Expected Loss}$$

$$= \mathbf{E}_{D,(y_*,x_*)}\left[(y_* - h(x_*) + h(x_*) - f(x_*|D))^{\mathbf{2}}\right]$$

$$= \mathbf{E}_{\epsilon_*}\left[(y_* - h(x_*))^{\mathbf{2}}\right] + \mathbf{E}_D\left[(h(x_*) - f(x_*|D))^{\mathbf{2}}\right]$$

Noise Term
(out of our control)
☹

Model Estimation Error
(we want to minimize this)

Expand

- Minimum error is governed by the noise.

- Expanding on the model estimation error:

$$\mathbf{E}_D\left[(h(x_*) - f(x_*|D))^2\right]$$

- Completing the squares with $\mathbf{E}\left[f(x_*|D)\right] = \bar{f}_*$

$$\mathbf{E}_D\left[(h(x_*) - f(x_*|D))^2\right]$$

$$= \mathbf{E}\left[(h(x_*) - \mathbf{E}\left[f(x_*|D)\right] + \mathbf{E}\left[f(x_*|D)\right] - f(x_*|D))^2\right]$$

$$= \mathbf{E}\left[(h(x_*) - \mathbf{E}\left[f(x_*|D)\right])^2\right] + \mathbf{E}\left[(f(x_*|D) - \mathbf{E}\left[f(x_*|D)\right])^2\right]$$

$$+ 2\mathbf{E}\left[h_* \bar{f}_* - h_* f_* - \bar{f}_* f_* + \bar{f}_*^2\right]$$

$$= h_* \bar{f}_* - h_* \mathbf{E}\left[f_*\right] - \bar{f}_* \mathbf{E}\left[f_*\right] + \bar{f}_*^2 =$$

$$h_* \bar{f}_* - h_* \bar{f}_* - \bar{f}_* \bar{f}_* + \bar{f}_*^2 = 0$$

- Expanding on the model estimation error:

$$\mathbf{E}_D \left[(h(x_*) - f(x_*|D))^2 \right]$$

- Completing the squares with $\mathbf{E}\left[f(x_*|D)\right] = \bar{f}_*$

$$\mathbf{E}_D \left[(h(x_*) - f(x_*|D))^2 \right]$$
$$= \mathbf{E}\left[(h(x_*) - \mathbf{E}\left[f(x_*|D)\right])^2 \right] + \mathbf{E}\left[(f(x_*|D) - \mathbf{E}\left[f(x_*|D)\right])^2 \right]$$

$$(h(x_*) - \mathbf{E}\left[f(x_*|D)\right])^2$$

- Expanding on the model estimation error:
$$\mathbf{E}_D\left[(h(x_*) - f(x_*|D))^2\right]$$

- Completing the squares with $\mathbf{E}\left[f(x_*|D)\right] = \bar{f}_*$

$$\mathbf{E}_D\left[(h(x_*) - f(x_*|D))^2\right]$$
$$= \underbrace{(h(x_*) - \mathbf{E}\left[f(x_*|D)\right])^2}_{\text{(Bias)}^2} + \underbrace{\mathbf{E}\left[(f(x_*|D) - \mathbf{E}\left[f(x_*|D)\right])^2\right]}_{\text{Variance}}$$

- Tradeoff between bias and variance:
  - **Simple Models:** High Bias, Low Variance
  - **Complex Models:** Low Bias, High Variance

Summary of Bias Variance Tradeoff

$$\mathbf{E}_{D,(y_*,x_*)}\left[(y_* - f(x_*|D))^2\right] = \qquad \text{Expected Loss}$$

$$\mathbf{E}_{\epsilon_*}\left[(y_* - h(x_*))^2\right] \qquad\qquad \text{Noise}$$

$$+ (h(x_*) - \mathbf{E}_D\left[f(x_*|D)\right])^2 \qquad \text{(Bias)}^2$$

$$+ \mathbf{E}_D\left[(f(x_*|D) - \mathbf{E}_D\left[f(x_*|D)\right])^2\right] \quad \text{Variance}$$

- Choice of models balances bias and variance.
  - Over-fitting ➜ Variance is too High
  - Under-fitting ➜ Bias is too High

Bias Variance Plot

Image from http://scott.fortmann-roe.com/docs/BiasVariance.html
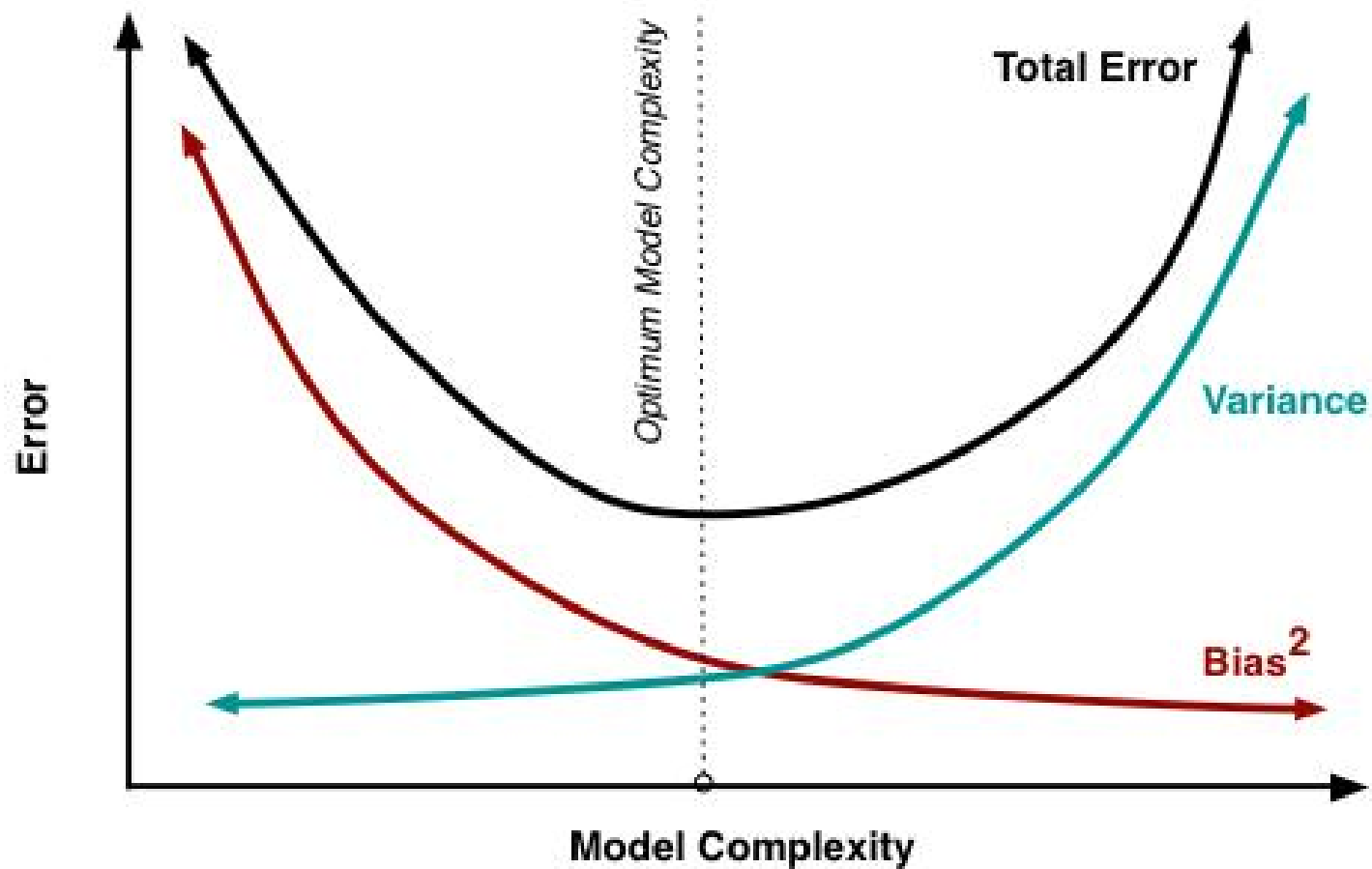
Analyze bias of $f(x_*|D) = x_*^T \hat{\theta}_{\mathrm{MLE}}$

- Assume a true model is linear: $h(x_*) = x_*^T \theta$

$$\mathrm{bias} = h(x_*) - \mathbf{E}_D\left[f(x_*|D)\right]$$

$$= x_*^T \theta - \mathbf{E}_D\left[x_*^T \hat{\theta}_{\mathrm{MLE}}\right]$$

$$= x_*^T \theta - \mathbf{E}_D\left[x_*^T (X^T X)^{-1} X^T Y\right]$$

$$= x_*^T \theta - \mathbf{E}_D\left[x_*^T (X^T X)^{-1} X^T (X\theta + \epsilon)\right]$$

$$= x_*^T \theta - \mathbf{E}_D\left[x_*^T (X^T X)^{-1} X^T X\theta + x_*^T (X^T X)^{-1} X^T \epsilon\right]$$

$$= x_*^T \theta - \mathbf{E}_D\left[x_*^T \theta + x_*^T (X^T X)^{-1} X^T \epsilon\right]$$

$$= x_*^T \theta - x_*^T \theta + x_*^T (X^T X)^{-1} X^T \mathbf{E}_D\left[\epsilon\right]$$

$$= x_*^T \theta - x_*^T \theta = 0$$

Substitute MLE

Plug in definition of Y

Expand and cancel

Assumption:
$$\mathbf{E}_D\left[\epsilon\right] = 0$$

$\hat{\theta}_{\mathrm{MLE}}$ is unbiased!

Analyze Variance of $f(x_*|D) = x_*^T \hat{\theta}_{\mathrm{MLE}}$

- Assume a true model is linear: $h(x_*) = x_*^T \theta$

$$\mathrm{Var.} = \mathbf{E}\left[(f(x_*|D) - \mathbf{E}_D[f(x_*|D)])^2\right]$$

$$= \mathbf{E}\left[(x_*^T \hat{\theta}_{\mathrm{MLE}} - x_*^T \theta)^2\right]$$ Substitute MLE + unbiased result

$$= \mathbf{E}\left[(x_*^T (X^T X)^{-1} X^T Y - x_*^T \theta)^2\right]$$ Plug in definition of Y

$$= \mathbf{E}\left[(x_*^T (X^T X)^{-1} X^T (X\theta + \epsilon) - x_*^T \theta)^2\right]$$

$$= \mathbf{E}\left[(x_*^T \theta + x_*^T (X^T X)^{-1} X^T \epsilon - x_*^T \theta)^2\right]$$

$$= \mathbf{E}\left[(x_*^T (X^T X)^{-1} X^T \epsilon)^2\right]$$ Expand and cancel

- Use property of scalar: a^2 = a a^T

Analyze Variance of $f(x_*|D) = x_*^T \hat{\theta}_{\mathrm{MLE}}$

- Use property of scalar: a² = a aᵀ

$$\mathrm{Var.} = \mathbf{E}\left[(f(x_*|D) - \mathbf{E}_D\left[f(x_*|D)\right])^{\mathbf{2}}\right]$$

$$= \mathbf{E}\left[(x_*^T(X^TX)^{-1}X^T\boldsymbol{\epsilon})^{\mathbf{2}}\right]$$

$$= \mathbf{E}\left[(x_*^T(X^TX)^{-1}X^T\boldsymbol{\epsilon})(x_*^T(X^TX)^{-1}X^T\boldsymbol{\epsilon})^T\right]$$

$$= \mathbf{E}\left[x_*^T(X^TX)^{-1}X^T\boldsymbol{\epsilon}\boldsymbol{\epsilon}^T(x_*^T(X^TX)^{-1}X^T)^T\right]$$

$$= x_*^T(X^TX)^{-1}X^T\mathbf{E}\left[\boldsymbol{\epsilon}\boldsymbol{\epsilon}^T\right](x_*^T(X^TX)^{-1}X^T)^T$$

$$= x_*^T(X^TX)^{-1}X^T\sigma_\epsilon^2 I(x_*^T(X^TX)^{-1}X^T)^T$$

$$= \sigma_\epsilon^2 x_*^T(X^TX)^{-1}X^TX(x_*^T(X^TX)^{-1})^T$$

$$= \sigma_\epsilon^2 x_*^T(x_*^T(X^TX)^{-1})^T$$

$$= \sigma_\epsilon^2 x_*^T(X^TX)^{-1}x_*$$

Consequence of Variance Calculation

$$\text{Var.} = \mathbf{E}\left[(f(x_*|D) - \mathbf{E}_D\left[f(x_*|D)\right])^{\mathbf{2}}\right]$$
$$= \sigma_{\boldsymbol{\epsilon}}^{\mathbf{2}} x_*^T (X^T X)^{-1} x_*$$

Higher Variance   Lower Variance
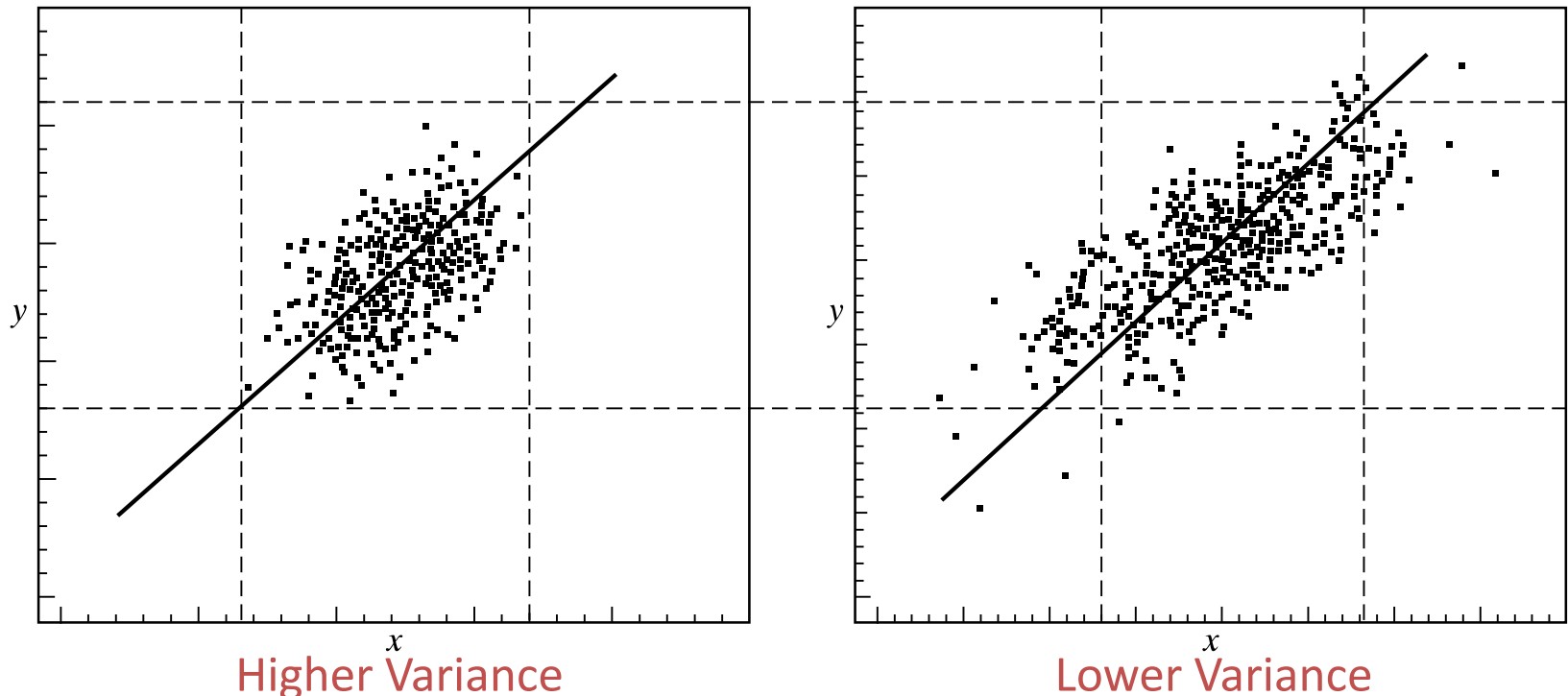
Analyze Variance of $f(x_*|D) = x_*^T \hat{\theta}_{\mathrm{MLE}}$

- Assume a true model is linear: $h(x_*) = x_*^T \theta$

$$\mathrm{Var.} = \mathbf{Var}\left[f(x_*|D) - \mathbf{E}_D\left[f(x_*|D)\right]\right]$$

$$= \mathbf{Var}\left[x_*^T \hat{\theta}_{\mathrm{MLE}} - x_*^T \theta\right]$$

Substitute MLE + unbiased result

$$= \mathbf{Var}\left[x_*^T (X^T X)^{-1} X^T Y - x_*^T \theta\right]$$

Plug in definition of Y

$$= \mathbf{Var}\left[x_*^T (X^T X)^{-1} X^T (X\theta + \epsilon) - x_*^T \theta\right]$$

$$= \mathbf{Var}\left[x_*^T \theta + x_*^T (X^T X)^{-1} X^T \epsilon - x_*^T \theta\right]$$

$$= \mathbf{Var}\left[x_*^T (X^T X)^{-1} X^T \epsilon\right]$$

Expand and cancel

- Next: use matrix variance identity

Analyze Variance of $f(x_* | D) = x_*^T \hat{\theta}_{\mathrm{MLE}}$

- Define: $A = x_*^T (X^T X)^{-1} X^T$

$$\mathrm{Var.} = \mathbf{Var}\left[x_*^T (X^T X)^{-1} X^T \boldsymbol{\epsilon} \right] = \mathbf{Var}\left[A\boldsymbol{\epsilon} \right]$$

- Use matrix variance identity: $\mathbf{Var}\left[A\boldsymbol{\epsilon} \right] = A\Sigma_\epsilon A^T$

$$\mathrm{Var.} = A\Sigma_\epsilon A^T = \sigma_\epsilon^2 A A^T$$

$$= \sigma_\epsilon^2 x_*^T (X^T X)^{-1} X^T (x_*^T (X^T X)^{-1} X^T)^T$$

$$= \sigma_\epsilon^2 x_*^T (X^T X)^{-1} X^T X (x_*^T (X^T X)^{-1})^T$$

$$= \sigma_\epsilon^2 x_*^T (x_*^T (X^T X)^{-1})^T$$

$$= \sigma_\epsilon^2 x_*^T (X^T X)^{-1} x_*$$

- If we assume x is iid N(0, 1): $\mathbf{E}_{X, x_*}\left[\mathrm{Var.} \right] = \sigma_\epsilon^2 \dfrac{p}{n}$

Deriving the final identity

- Assume x_i and x_* are N(0,1)

$$
\begin{aligned}
\mathbf{E}_{X,x_*}\left[\text{Var.}\right] &= \sigma_{\boldsymbol{\epsilon}}^{\mathbf{2}} \mathbf{E}_{X,x_*}\left[x_*^T(X^TX)^{-1}x_*\right] \\
&= \sigma_{\boldsymbol{\epsilon}}^{\mathbf{2}} \mathbf{E}_{X,x_*}\left[tr(x_*x_*^T(X^TX)^{-1})\right] \\
&= \sigma_{\boldsymbol{\epsilon}}^{\mathbf{2}} tr(\mathbf{E}_{X,x_*}\left[x_*x_*^T(X^TX)^{-1}\right]) \\
&= \sigma_{\boldsymbol{\epsilon}}^{\mathbf{2}} tr(\mathbf{E}_{x_*}\left[x_*x_*^T\right]\mathbf{E}_X\left[(X^TX)^{-1}\right]) \\
&= \frac{\sigma_{\boldsymbol{\epsilon}}^{\mathbf{2}}}{n} tr(\mathbf{E}_{x_*}\left[x_*x_*^T\right]) \\
&= \frac{\sigma_{\boldsymbol{\epsilon}}^{\mathbf{2}}}{n} p
\end{aligned}
$$

Summary

- Least-Square Regression is Unbiased:

$$\mathbf{E}_D \left[x_*^T \hat{\theta}_{\mathrm{MLE}} \right] = x_*^T \theta$$

- Variance depends on:

$$\mathbf{E} \left[(f(x_*|D) - \mathbf{E}\left[f(x_*|D)\right])^2 \right] = \sigma_\epsilon^2 x_*^T (X^T X)^{-1} x_*$$

$$\approx \sigma_\epsilon^2 \frac{p}{n}$$

  – Number of data-points *n*
  – Dimensionality *p*
  – Not on observations *Y*

Gauss-Markov Theorem

- The linear model:

$$f(x_*) = x_*^T \hat{\theta}_{\mathrm{MLE}} = x_*^T (X^T X)^{-1} X^T Y$$

  has the **minimum variance** among all **unbiased** linear estimators
  - Note that this is linear in Y

- **BLUE: B**est **L**inear **U**nbiased **E**stimator

Summary

- Introduced the Least-Square regression model
  - Maximum Likelihood: Gaussian Noise
  - Loss Function: Squared Error
  - Geometric Interpretation: Minimizing Projection
- Derived the normal equations:
  - Walked through process of constructing MLE
  - Discussed efficient computation of the MLE
- Introduced basis functions for non-linearity
  - Demonstrated issues with over-fitting
- Derived the classic bias-variance tradeoff
  - Applied to least-squares model

Logistic Regression

Classification

- **Learn**: h:**X**->Y
  - **X** – features
  - Y – target classes

- Suppose you know P(Y|**X**) exactly, how should you classify?
  - Bayes classifier:
  $$y^* = h_{bayes}(x) = \arg\max_y P(Y = y \mid X = x)$$

- Why?

Generative vs. Discriminative Classifiers - Intuition

- Generative classifier, e.g., Naïve Bayes:
  - Assume some functional form for **P(X|Y), P(Y)**
  - Estimate parameters of P(X|Y), P(Y) directly from training data
  - Use Bayes rule to calculate P(Y|X=x)
  - This is 'generative' model
    - Indirect computation of P(Y|X) through Bayes rule
    - But, can generate a sample of the data, $P(X) = \sum_{y} P(y)P(X \mid y)$

- Discriminative classifier, e.g., Logistic Regression:
  - Assume some functional form for **P(Y|X)**
  - Estimate parameters of P(Y|X) directly from training data
  - This is the 'discriminative' model
    - Directly learn P(Y|X)
    - But cannot sample data, because P(X) is not available

The Naïve Bayes Classifier

- Given:
  - Prior P(Y)
  - n conditionally independent features X given the class Y
  - For each Xi, we have likelihood $P(X_i|Y)$
- Decision rule:

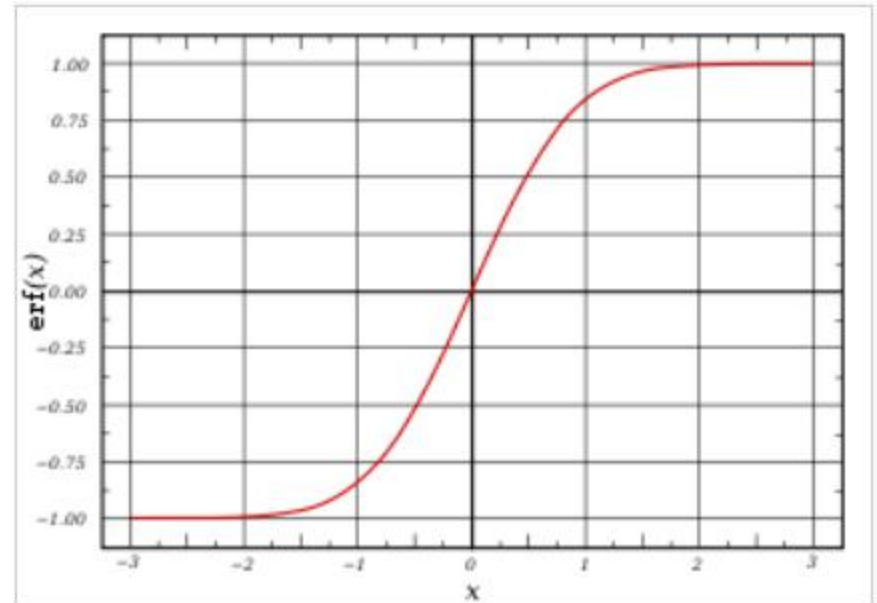$$y^* = h_{NB}(x) = \arg\max_y P(y)P(x_1,...,x_n \mid y)$$

$$= \arg\max_y P(y)\prod_i P(x_i \mid y)$$

- If assumption holds, NB is optimal classifier!

Logistic Regression

- Let X be the data instance, and Y be the class label: Learn P(Y|X) directly
  - Let W = (W1, W2, … Wn), X=(X1, X2, …, Xn), **WX** is the dot product
  - Sigmoid function:

$$P(Y = 1 \mid \mathbf{X}) = \frac{1}{1 + e^{-\mathbf{wx}}}$$

Logistic Regression

- In logistic regression, we learn the conditional distribution $P(y|x)$
- Let $p_y(x;w)$ be our estimate of $P(y|x)$, where w is a vector of adjustable parameters.
- Assume there are two classes, $y = 0$ and $y = 1$ and

$$p_1(\mathbf{x};\mathbf{w}) = \frac{1}{1+e^{-\mathbf{wx}}} \qquad p_0(\mathbf{x};\mathbf{w}) = 1 - \frac{1}{1+e^{-\mathbf{wx}}}$$

- This is equivalent to

$$\log \frac{p_1(\mathbf{x};\mathbf{w})}{p_0(\mathbf{x};\mathbf{w})} = \mathbf{wx}$$

- That is, the log odds of class 1 is a linear function of x
- Q: How to find **W**?

Constructing a Learning Algorithm

- The conditional data likelihood is the probability of the observed Y values in the training data, conditioned on their corresponding X values. We choose parameters w that satisfy

$$\mathbf{w} = \arg\max_{\mathbf{w}} \prod_{l} P(y^l \mid \mathbf{x}^l, \mathbf{w})$$

- where $\mathbf{w} = <w_0, w_1, ..., w_n>$ is the vector of parameters to be estimated, y^l denotes the observed value of Y in the l th training example, and $\mathbf{x}^l$ denotes the observed value of $\mathbf{X}$ in the l th training example

Constructing a Learning Algorithm

- Equivalently, we can work with the log of the conditional likelihood:

$$\mathbf{w} = \arg\max_{\mathbf{w}} \sum_{l} \ln P(y^l \mid \mathbf{x}^l, \mathbf{w})$$

- This conditional data log likelihood, which we will denote l(W) can be written as

$$l(\mathbf{w}) = \sum_{l} y^l \ln P(y^l = 1 \mid \mathbf{x}^l, \mathbf{w}) + (1 - y^l) \ln P(y^l = 0 \mid \mathbf{x}^l, \mathbf{w})$$

- Note here we are utilizing the fact that Y can take only values 0 or 1, so only one of the two terms in the expression will be non-zero for any given y^l

Computing the Likelihood

- We can re-express the log of the conditional likelihood as:

$$l(\mathbf{w}) = \sum_l y^l \ln P(y^l = 1 \mid \mathbf{x}^l, \mathbf{w}) + (1 - y^l) \ln P(y^l = 0 \mid \mathbf{x}^l, \mathbf{w})$$

$$= \sum_l y^l \ln \frac{P(y^l = 1 \mid \mathbf{x}^l, \mathbf{w})}{P(y^l = 0 \mid \mathbf{x}^l, \mathbf{w})} + \ln P(y^l = 0 \mid \mathbf{x}^l, \mathbf{w})$$

$$= \sum_l y^l (w_0 + \sum_{i=1}^{n} w_i x_i^l) - \ln(1 + \exp(w_0 + \sum_{i=1}^{n} w_i x_i^l))$$

Fitting LR by Gradient Ascent

- Unfortunately, there is no closed form solution to maximizing l(**w**) with respect to **w**. Therefore, one common approach is to use gradient ascent

- The i th component of the vector gradient has the form

Logistic Regression prediction

$$\frac{\partial}{\partial w_i} l(\mathbf{w}) = \sum_l x_i^l (y^l - \hat{P}(y^l = 1 \mid \mathbf{x}^l, \mathbf{w}))$$

^ gu

Fitting LR by Gradient Ascent

- Given this formula for the derivative of each wi, we can use standard gradient ascent to optimize the weights **w**. Beginning with initial weights of zero, we repeatedly update the weights in the direction of the gradient, changing the i th weight according to

$$w_i \leftarrow w_i + \eta \sum_l x_i^l (y^l - \hat{P}(y^l = 1 \mid \mathbf{x}^l, \mathbf{w}))$$

Regularization in Logistic Regression

- Overfitting the training data is a problem that can arise in Logistic Regression, especially when data has very high dimensions and is sparse.

- One approach to reducing overfitting is regularization, in which we create a modified "penalized log likelihood function," which penalizes large values of **w**.

$$\mathbf{w} = \arg\max_{\mathbf{w}} \sum_{l} \ln P(y^l \mid \mathbf{x}^l, \mathbf{w}) - \frac{\lambda}{2} \| \mathbf{w} \|^2$$

Regularization in Logistic Regression

- The derivative of this penalized log likelihood function is similar to our earlier derivative, with one additional penalty term

$$\frac{\partial}{\partial w_i} l(\mathbf{w}) = \sum_l x_i^l (y^l - \hat{P}(y^l = 1 \mid \mathbf{x}^l, \mathbf{w})) - \lambda w_i$$

- which gives us the modified gradient descent rule

$$w_i \leftarrow w_i + \eta \sum_l x_i^l (y^l - \hat{P}(y^l = 1 \mid \mathbf{x}^l, \mathbf{w})) - \eta \lambda w_i$$

Summary of Logistic Regression

- Learns the Conditional Probability Distribution P(y|x)
- Local Search.
  - Begins with initial weight vector.
  - Modifies it iteratively to maximize an objective function.
  - The objective function is the conditional log likelihood of the data – so the algorithm seeks the probability distribution P(y|x) that is most likely given the data.

What you should know LR

- In general, NB and LR make different assumptions
  - NB: Features independent given class -> assumption on $P(X|Y)$
  - LR: Functional form of $P(Y|X)$, no assumption on $P(X|Y)$
- LR is a linear classifier
  - decision rule is a hyperplane
- LR optimized by conditional likelihood
  - no closed-form solution
  - concave -> global optimum with gradient ascent