

UVA CS 4501: Machine Learning

Lecture 3: Linear Regression

Dr. Yanjun Qi

University of Virginia
Department of Computer Science

SUPERVISED LEARNING

$$f : X \rightarrow Y$$

- Find function to map **input** space X to **output** space Y
- **Generalisation**: learn function / hypothesis from **past data** in order to “explain”, “predict”, “model” or “control” **new** data examples

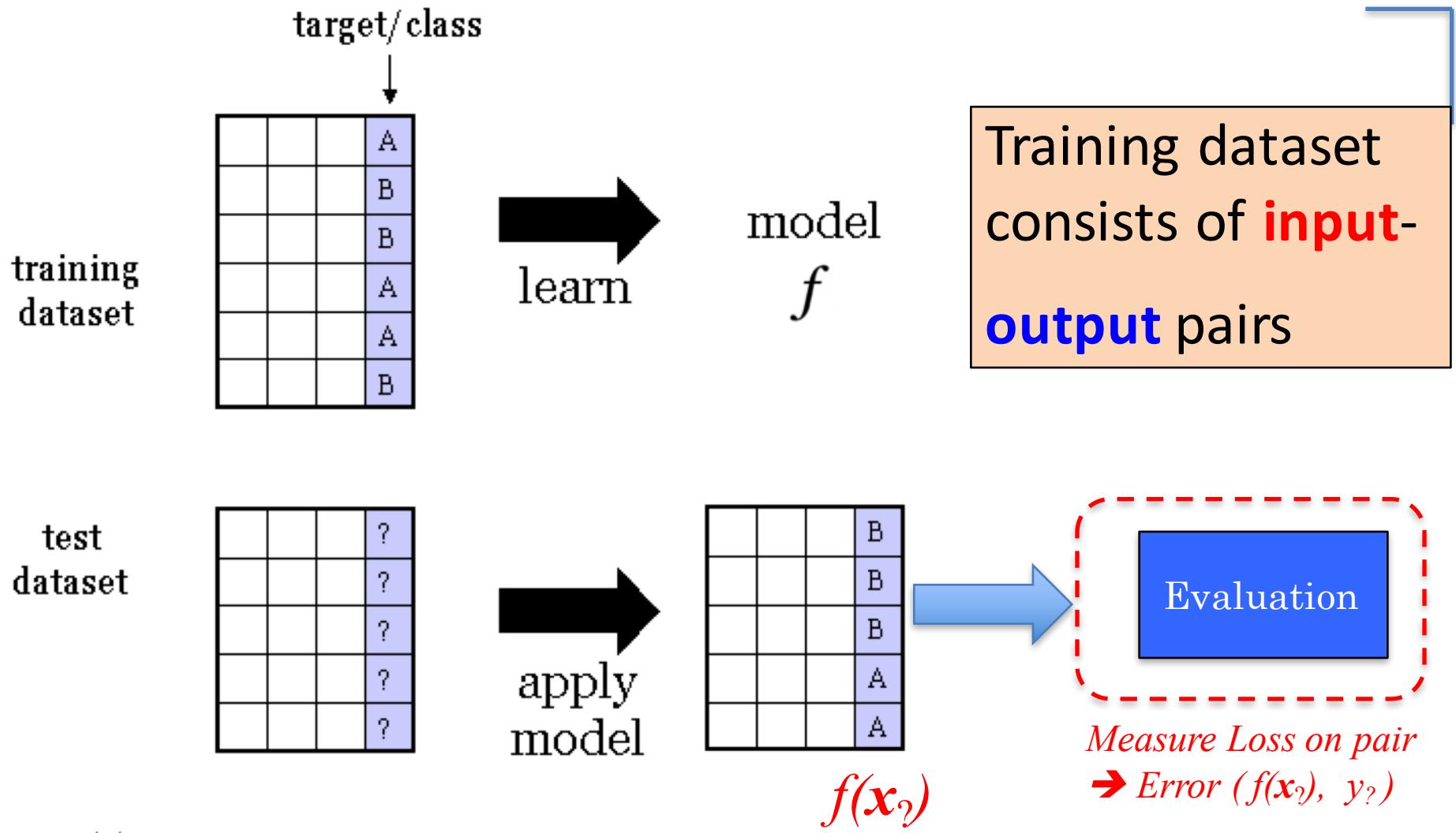
	X ₁	X ₂	X ₃	Y
S ₁				
S ₂				
S ₃				
S ₄				
S ₅				
S ₆				

A Dataset

$$f : \boxed{X} \longrightarrow \boxed{Y}$$

- **Data/points/instances/examples/samples/records:** [rows]
- **Features/attributes/dimensions/independent variables/covariates/predictors/regressors:** [columns, except the last]
- **Target/outcome/response/label/dependent variable:** special column to be predicted [last column]

SUPERVISED LEARNING



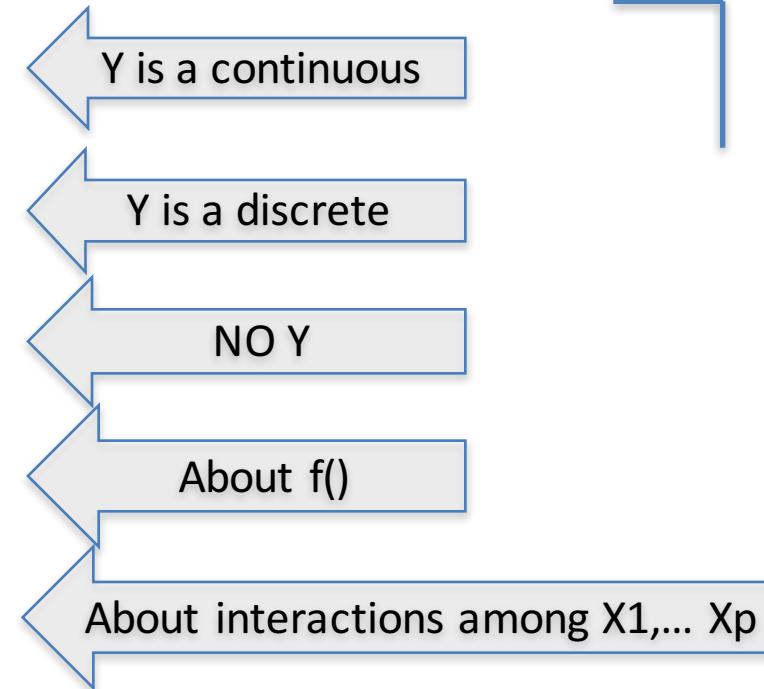
Notation

- Inputs
 - X, X_j (jth element of vector X) : random variables written in capital letter
 - p #inputs, N #observations
 - X : matrix written in bold capital
 - Vectors are assumed to be column vectors
 - Discrete inputs often described by characteristic vector (dummy variables)
- Outputs
 - quantitative Y
 - qualitative C (for categorical)
- Observed variables written in lower case
 - The i-th observed value of X is x_i and can be a scalar or a vector

Where are we?

Five major sections of this course

- Regression (supervised)
- Classification (supervised)
- Unsupervised models
- Learning theory
- Graphical models



X ₁	X ₂	X ₃	Y

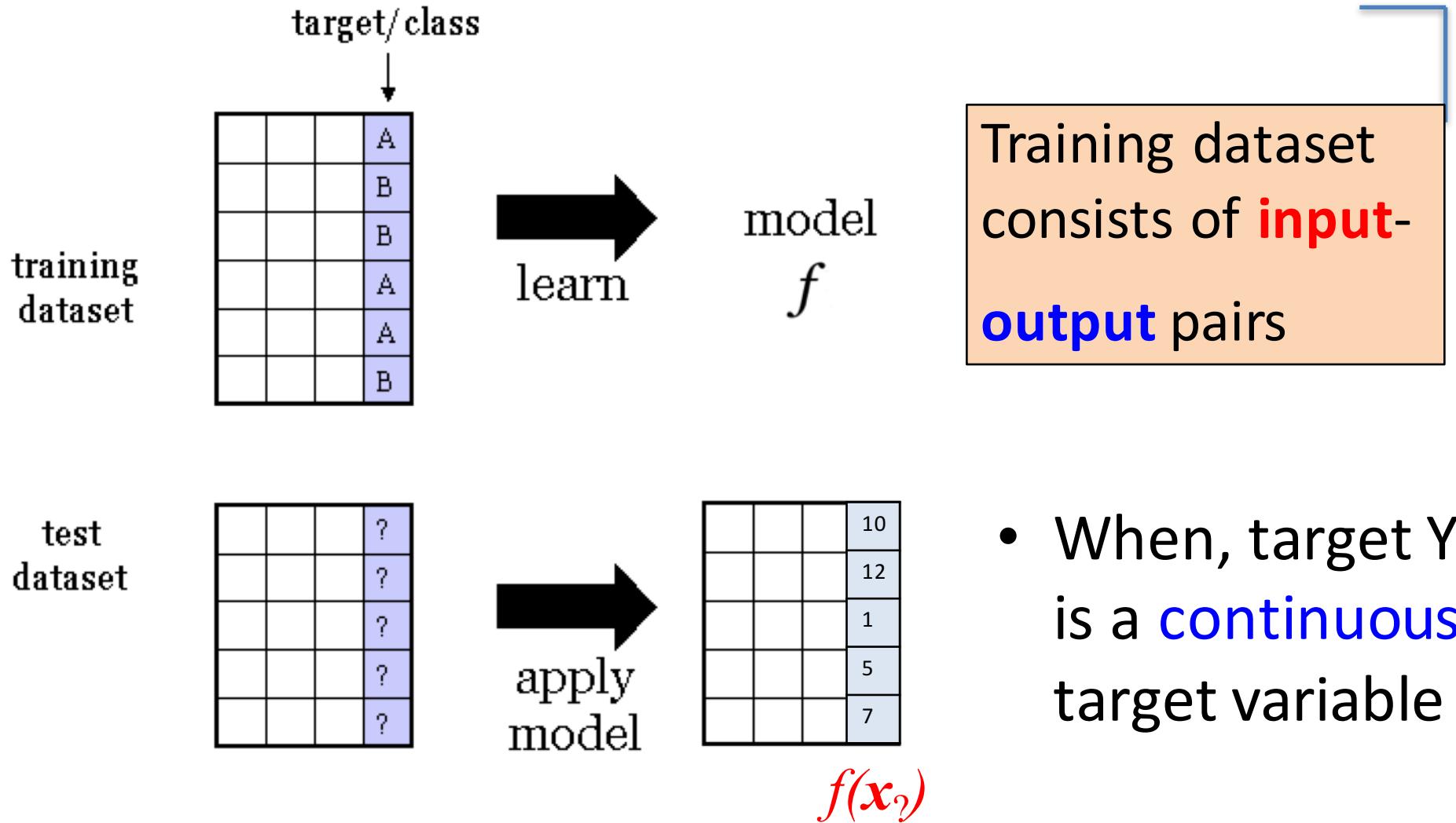
A Dataset for regression

$$f : \boxed{X} \longrightarrow \boxed{Y}$$

continuous
valued
variable

- **Data/points/instances/examples/samples/records:** [rows]
- **Features/attributes/dimensions/independent variables/covariates/predictors/regressors:** [columns, except the last]
- **Target/outcome/response/label/dependent variable:** special column to be predicted [last column]

SUPERVISED Regression



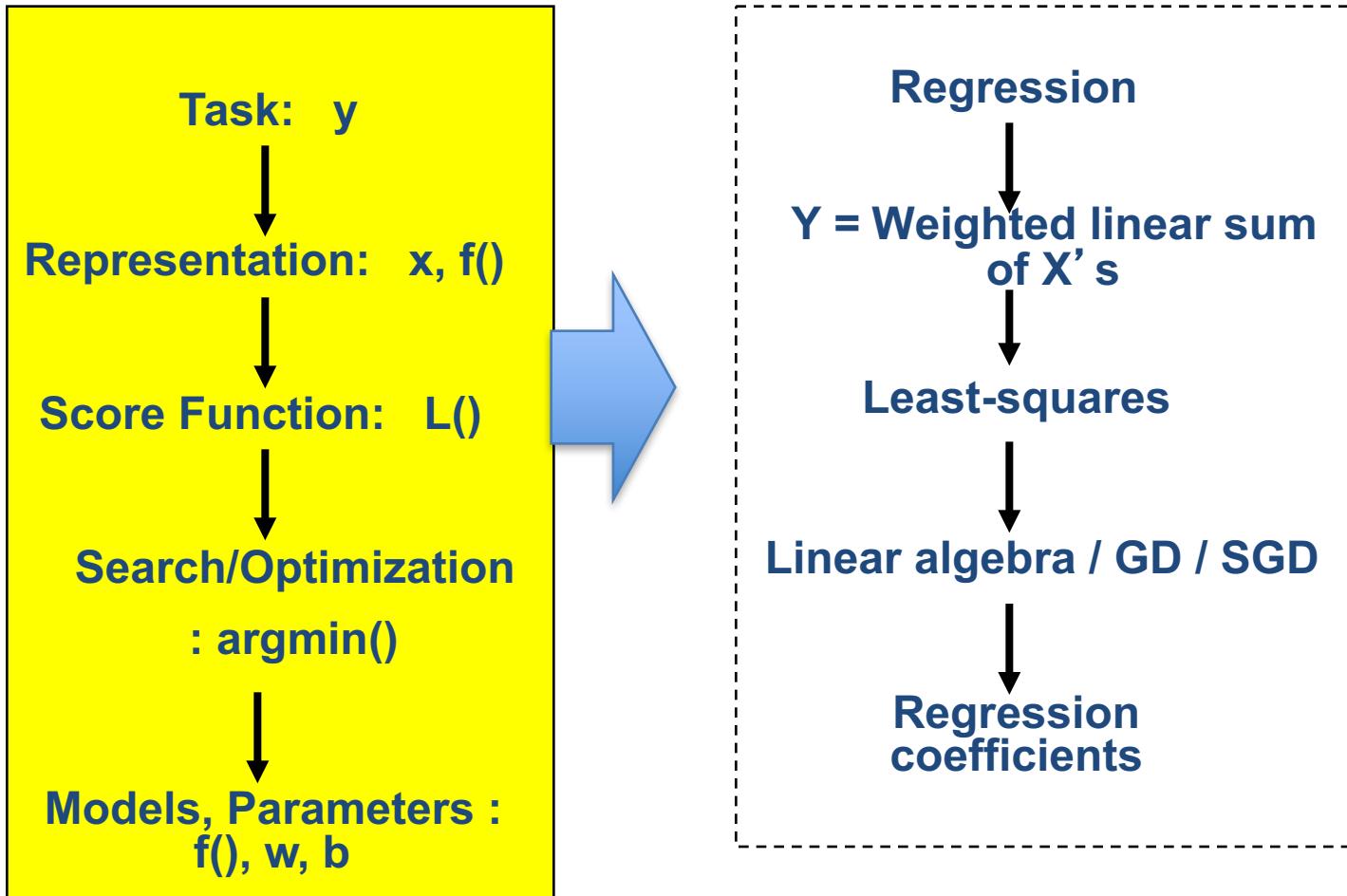
training dataset

$$\mathbf{X}_{train} = \begin{bmatrix} \cdots & \mathbf{x}_1^T & \cdots \\ \cdots & \mathbf{x}_2^T & \cdots \\ \vdots & \vdots & \vdots \\ \cdots & \mathbf{x}_n^T & \cdots \end{bmatrix}$$
$$\bar{\mathbf{y}}_{train} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$$

test dataset

$$\mathbf{X}_{test} = \begin{bmatrix} \cdots & \mathbf{x}_{n+1}^T & \cdots \\ \cdots & \mathbf{x}_{n+2}^T & \cdots \\ \vdots & \vdots & \vdots \\ \cdots & \mathbf{x}_{n+m}^T & \cdots \end{bmatrix}$$
$$\bar{\mathbf{y}}_{test} = \begin{bmatrix} y_{n+1} \\ y_{n+2} \\ \vdots \\ y_{n+m} \end{bmatrix}$$

Multivariate Linear Regression Variations in a Nutshell



$$\hat{y} = f(x) = \theta^T x$$

Regression (supervised)

- Four ways to train / perform optimization for linear regression models
 - Normal Equation
 - Gradient Descent (GD)
 - Stochastic GD
 - Newton's method
- Supervised regression models
 - Linear regression (LR)
 - LR with non-linear basis functions
 - Locally weighted LR
 - LR with Regularizations

Today

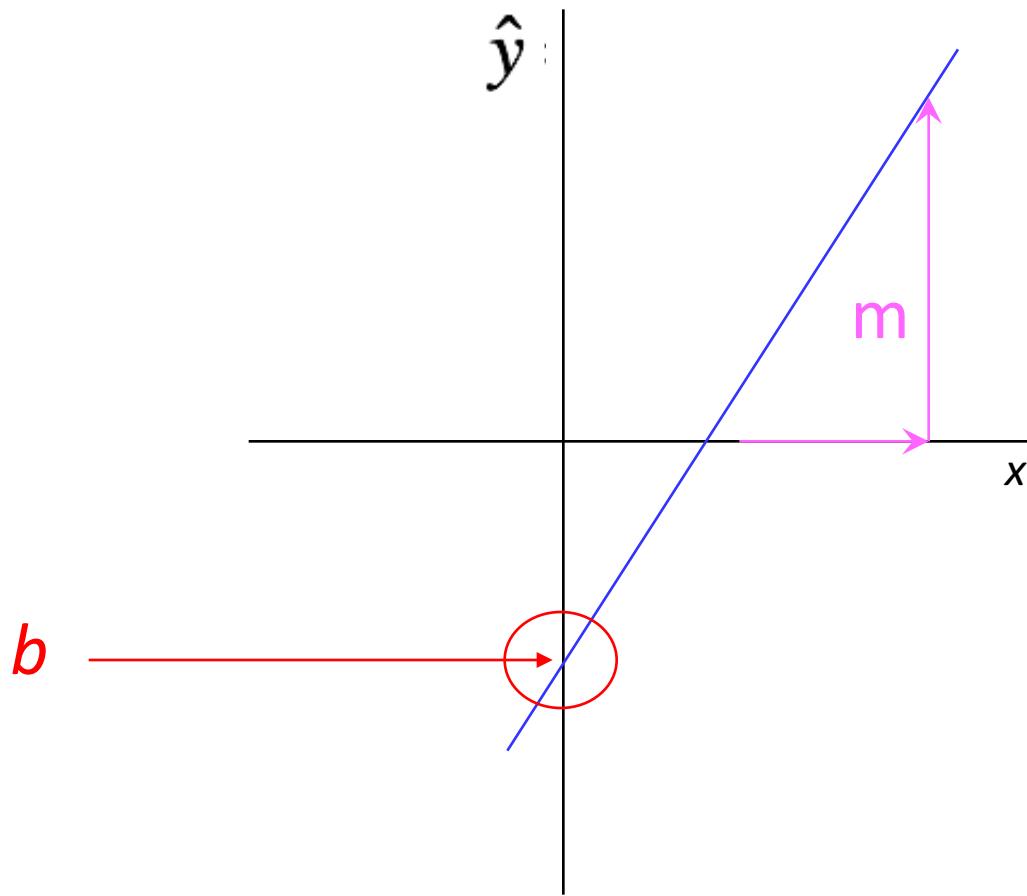
- ❑ Linear regression (aka **least squares**)
- ❑ Learn to derive the least squares estimate by normal equation
- ❑ Evaluation with Cross-validation

A few concrete examples

Review: $f(x)$ is Linear when X is 1D

- $f(x)=mx+b?$

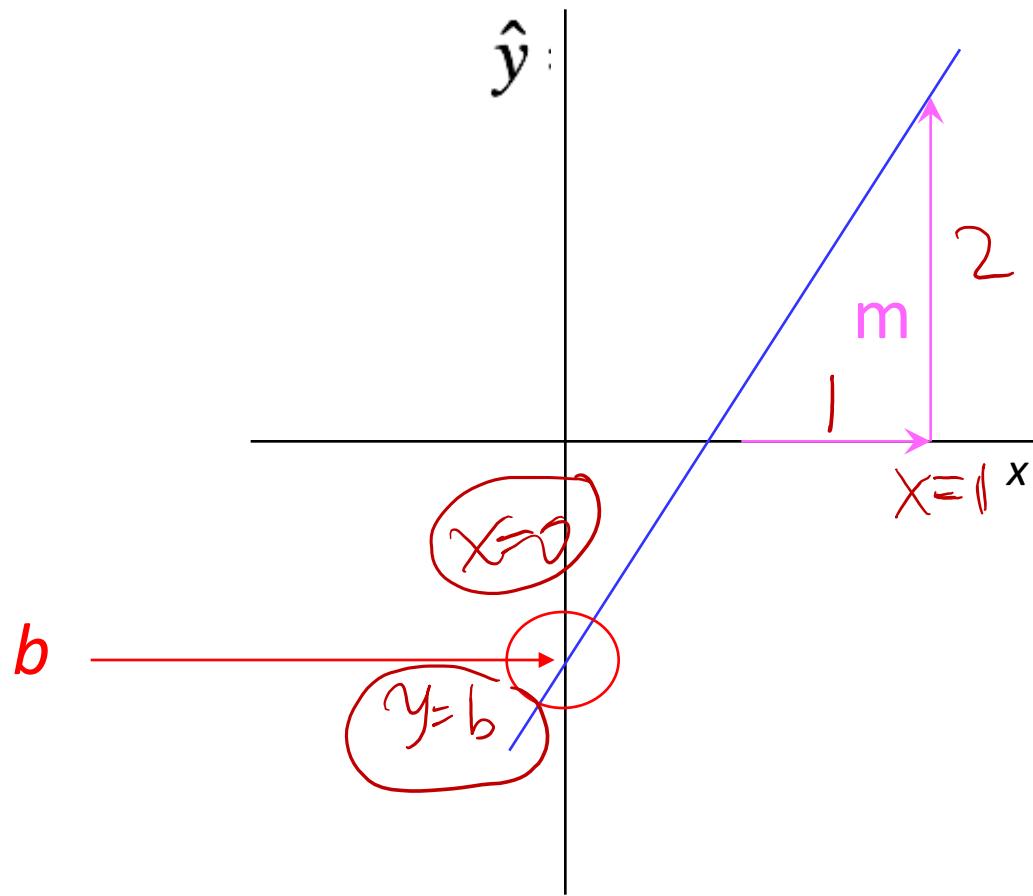
A slope of 2 (i.e. $m=2$) means that every 1-unit change in X yields a 2-unit change in Y.

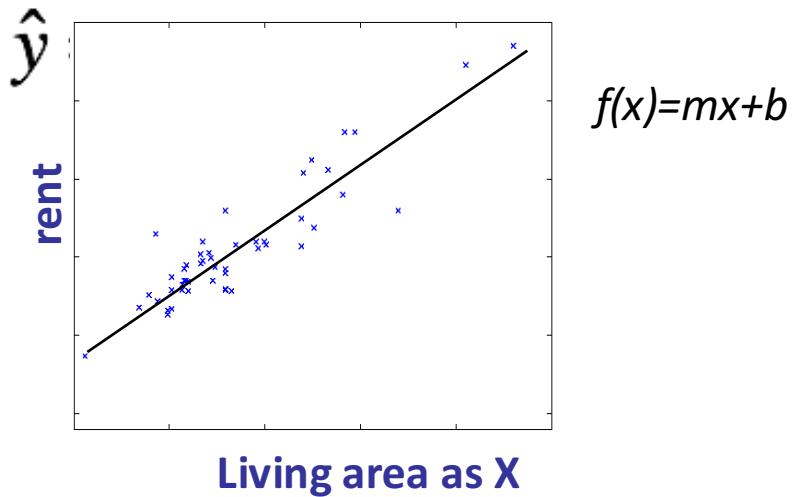


Review: $f(x)$ is Linear when X is 1D

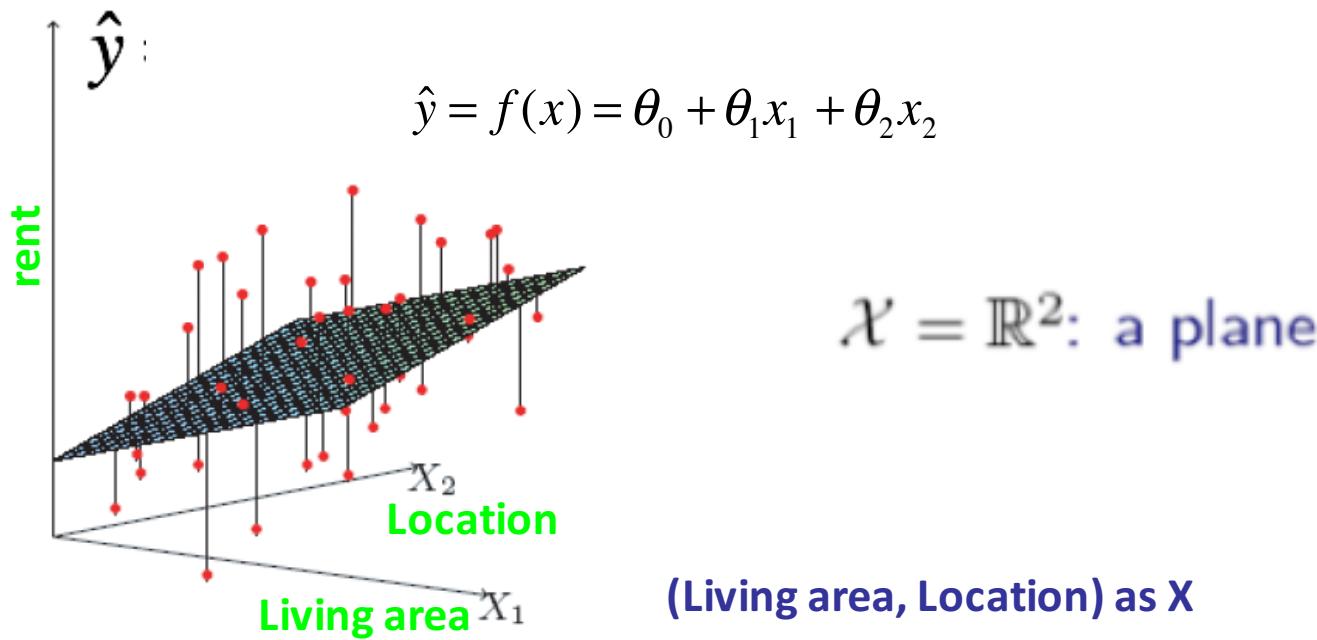
- $f(x)=mx+b?$

A slope of 2 (i.e. $m=2$) means that every 1-unit change in X yields a 2-unit change in Y .





1D case ($\mathcal{X} = \mathbb{R}$): a line



$\mathcal{X} = \mathbb{R}^2$: a plane

(Living area, Location) as X

Linear SUPERVISED Regression

$$[f]: X \longrightarrow Y$$

e.g. Linear Regression Models

$$\hat{y} = f(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2$$

=> Features **x**:

e.g., Living area, distance to campus, # bedroom ...

=> Target **y**:

e.g., Rent → Continuous

Review: Special Uses for Matrix Multiplication

- Dot (or Inner) Product of two Vectors $\langle x, y \rangle$

which is the sum of products of elements in similar positions for the two vectors

$$\langle x, y \rangle = \langle y, x \rangle$$

Where $\langle x, y \rangle = x^T y \in \mathbb{R} = [x_1 \ x_2 \ \dots \ x_n] \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} = \sum_{i=1}^n x_i y_i.$

$$\mathbf{a}^\top \mathbf{b} = \mathbf{b}^\top \mathbf{a}$$

A new representation (for each single data sample)

- Assume that **each sample \mathbf{x}** is a column vector,

- Here we assume a pseudo "feature" $x_0=1$ (this is the **intercept term**), and **RE-define** the feature vector to be:

$$\mathbf{x}^T = [(x_0=1), x_1, x_2, \dots, x_p]$$

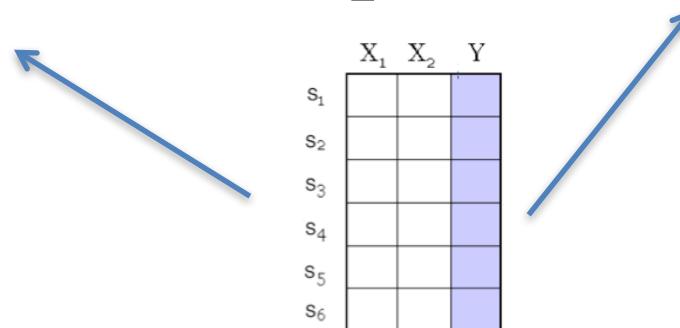
- the parameter vector θ is also a column vector

$$\theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \vdots \\ \theta_p \end{bmatrix} \quad \rightarrow \quad \hat{y} = f(\mathbf{x}) = \mathbf{x}^T \theta = \theta^T \mathbf{x}$$

Training Set to Matrix Form

- Now represent the whole Training set (with n samples) as matrix form :

$$\mathbf{X} = \begin{bmatrix} \cdots & \mathbf{x}_1^T & \cdots \\ \cdots & \mathbf{x}_2^T & \cdots \\ \vdots & \vdots & \vdots \\ \cdots & \mathbf{x}_n^T & \cdots \end{bmatrix} = \begin{bmatrix} x_{1,0} & x_{1,1} & \cdots & x_{1,p} \\ x_{2,0}^0 & x_{2,1} & \cdots & x_{2,p} \\ \vdots & \vdots & \vdots & \vdots \\ x_{n,0} & x_{n,1} & \cdots & x_{n,p} \end{bmatrix} \quad \mathbf{Y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$$



Regression Formulation: from a single example to multiple examples in train set

- Matrix-Vector Products (I)

Regression Formulation: from a single example to multiple examples in train set

- Represent as matrix form:
 - Predicted output

$$\hat{Y} = \mathbf{X}\theta = \begin{bmatrix} f(\mathbf{x}_1) \\ f(\mathbf{x}_2) \\ \vdots \\ f(\mathbf{x}_n) \end{bmatrix} = \begin{bmatrix} \mathbf{x}_1^T \theta \\ \mathbf{x}_2^T \theta \\ \vdots \\ \mathbf{x}_n^T \theta \end{bmatrix}$$

n × p p × 1 h × 1

- Labels (given output value)

$$\mathbf{Y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$$

n × 1

Now How to Learn the Regression Model?

- Using matrix form, we get the following general representation of the linear regression function:

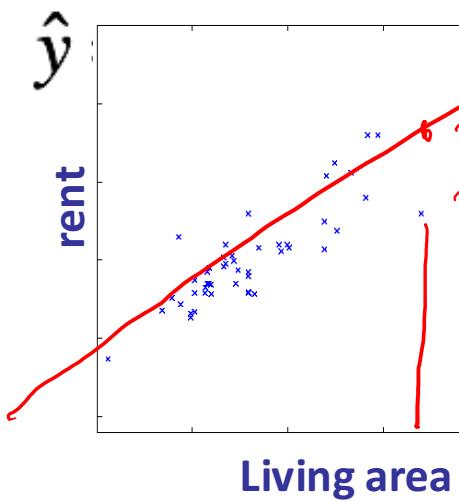
$$\hat{Y} = \mathbf{X}\theta$$

Y } $\Rightarrow \|Y - \hat{Y}\|_2^2$

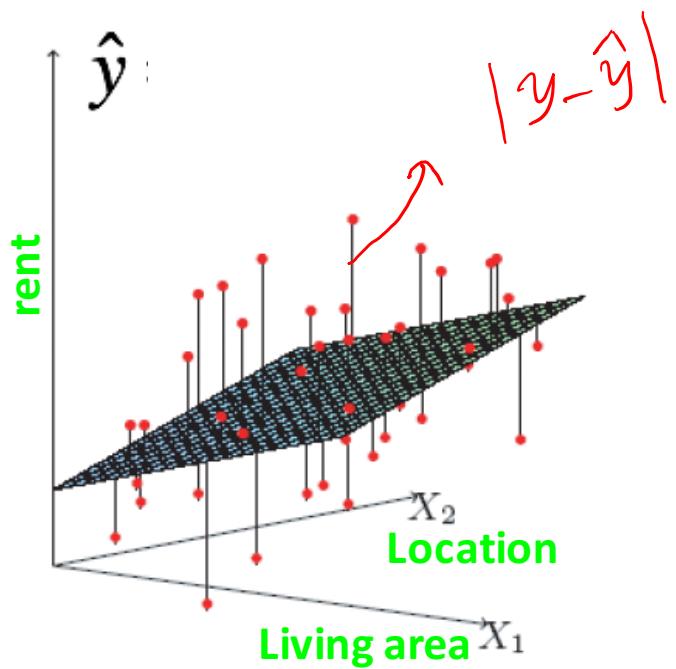
- Our goal is to pick the optimal θ that minimize the following lost function:

$$J(\theta) = \frac{1}{2} \sum_{i=1}^n (f(\mathbf{x}_i) - y_i)^2$$

SSE: Sum of squared error



1D case ($\mathcal{X} = \mathbb{R}$): a line



$\mathcal{X} = \mathbb{R}^2$: a plane

Today

- ❑ Linear regression (aka **least squares**)
- ❑ Learn to derive the least squares estimate by Normal Equation
- ❑ Evaluation with Cross-validation

Method I: normal equations

- Write the cost function in matrix form:

$$\begin{aligned}
 J(\theta) &= \frac{1}{2} \sum_{i=1}^n (\mathbf{x}_i^T \theta - y_i)^2 \\
 &= \frac{1}{2} (\mathbf{X}\theta - \bar{\mathbf{y}})^T (\mathbf{X}\theta - \bar{\mathbf{y}}) \\
 &= \frac{1}{2} (\theta^T \mathbf{X}^T \mathbf{X}\theta - \theta^T \mathbf{X}^T \bar{\mathbf{y}} - \bar{\mathbf{y}}^T \mathbf{X}\theta + \bar{\mathbf{y}}^T \bar{\mathbf{y}})
 \end{aligned}$$

$$\mathbf{X} = \begin{bmatrix} \cdots & \mathbf{x}_1^T & \cdots \\ \cdots & \mathbf{x}_2^T & \cdots \\ \vdots & \vdots & \vdots \\ \cdots & \mathbf{x}_n^T & \cdots \end{bmatrix} \quad \bar{\mathbf{y}} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$$

To minimize $J(\theta)$, take derivative and set to zero:

$$\Rightarrow \boxed{X^T X\theta = X^T \bar{y}}$$

The normal equations

WHY ??

$$\theta^* = \downarrow \left(X^T X \right)^{-1} X^T \bar{y}$$

Review: Special Uses for Matrix Multiplication

- Sum the Squared Elements of a Vector

$$\mathbf{a} = \begin{bmatrix} 5 \\ 2 \\ 8 \end{bmatrix}$$

$$J(\theta) = \frac{1}{2} \sum_{i=1}^n (\mathbf{x}_i^T \theta - y_i)^2$$

$$\mathbf{a}^T = \begin{bmatrix} 5 & 2 & 8 \end{bmatrix}$$

$$\mathbf{a}^T \mathbf{a} = \begin{bmatrix} 5 & 2 & 8 \end{bmatrix} \begin{bmatrix} 5 \\ 2 \\ 8 \end{bmatrix} = 5^2 + 2^2 + 8^2 = 93$$

Next : see White Board

$$a = \begin{bmatrix} \mathbf{x}_1^T \theta - y_1 \\ \mathbf{x}_2^T \theta - y_2 \\ \vdots \\ \mathbf{x}_n^T \theta - y_n \end{bmatrix} = X\theta - \vec{y}$$

$$\mathbf{a}^\top \mathbf{a} = 2J(\theta) = \sum_{i=1}^n (\mathbf{x}_i^T \theta - y_i)^2$$

$$J(\theta) = \frac{1}{2} \sum_{i=1}^n (\mathbf{x}_i^\top \theta - y_i)^2$$

$$J(\theta) = \frac{1}{2} \sum_{i=1}^n (\mathbf{x}_i^\top \theta - y_i)^2$$

$$= \frac{1}{2} (\mathbf{X}\theta - \mathbf{y})^\top (\mathbf{X}\theta - \mathbf{y})$$

$\mathbf{N} \times \mathbf{P}$ $\mathbf{P} \times \mathbf{1}$ $\mathbf{N} \times \mathbf{1}$

$$= \frac{1}{2} (\theta^\top \mathbf{X}^\top - \mathbf{y}^\top) (\mathbf{X}\theta - \mathbf{y})$$

$$= \frac{1}{2} (\theta^\top \mathbf{X}^\top \mathbf{X}\theta + \mathbf{y}^\top \mathbf{y} - \theta^\top \mathbf{X}^\top \mathbf{y} - \mathbf{y}^\top \mathbf{X}\theta)$$

$$\vec{a}^T \vec{b} = \vec{b}^T \vec{a}$$

$$\Rightarrow \vec{\theta}^T \vec{x}^T \vec{y} = \vec{y}^T \vec{x} \vec{\theta}$$

$$\Rightarrow J(\vec{\theta}) = \frac{1}{2} (\vec{\theta}^T \vec{x}^T \vec{x} \vec{\theta} - 2 \vec{\theta}^T \vec{x}^T \vec{y} + \vec{y}^T \vec{y})$$

A convex function is minimized
@ point whose

- ✓ derivative (slope) is zero
- ✓ gradient is zero vector
(multivariate case)

Review

$$\frac{\partial \mathbf{x}^T \mathbf{a}}{\partial \mathbf{x}} = \frac{\partial \mathbf{a}^T \mathbf{x}}{\partial \mathbf{x}} = \mathbf{a}$$

$$\frac{\partial \mathbf{x}^T \mathbf{Bx}}{\partial \mathbf{x}} = (\mathbf{B} + \mathbf{B}^T)\mathbf{x}$$



$$\Rightarrow \frac{\partial J(\theta)}{\partial \theta} = \frac{1}{2} (\mathbf{X}^T \mathbf{X} \theta - \mathbf{X}^T \mathbf{y}) \stackrel{\text{Set to } 0.}{=} 0.$$

Review (I): a simple example

$$f(w) = w^T a = [w_1, w_2, w_3] \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix} = w_1 + 2w_2 + 3w_3$$

→ Denominator layout

$$\begin{aligned} \frac{\partial f}{\partial w_1} &= 1 \\ \frac{\partial f}{\partial w_2} &= 2 \\ \frac{\partial f}{\partial w_3} &= 3 \end{aligned}$$

$$\frac{\partial f}{\partial w} = \frac{\partial w^T a}{\partial w} = a = \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix}$$

$$\frac{\partial(\theta^T X^T y)}{\partial \theta} = X^T y$$

Review (II): Gradient of Quadratic Func

- See L2-note Page 17, Page 23-24
- See white board

$$\frac{\partial(\theta^T X^T X \theta)}{\partial \theta} = \frac{\partial(\theta^T G \theta)}{\partial \theta} = 2G\theta = 2X^T X \theta$$

Taking Gradient of the $J(\theta)$

$$\Rightarrow \frac{\partial J(\theta)}{\partial \theta} = \frac{1}{2} (X^T X \theta - X^T y) \stackrel{\text{Set to } 0}{=} 0.$$

$$\Rightarrow X^T X \theta = X^T \bar{y}$$

The normal equations

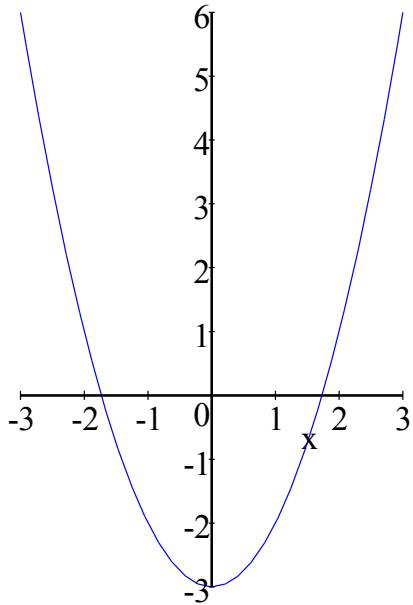
$$\theta^* = (X^T X)^{-1} X^T \bar{y}$$

$$\Rightarrow \theta = \underbrace{(X^T X)}_{\substack{p \times n \\ p \times p}}^{-1} \underbrace{X^T y}_{\substack{p \times 1 \\ p \times 1}} \Rightarrow p \times 1$$

Extra: Convex function

- Intuitively, a convex function **(1D case)** has a **single point at which the derivative goes to zero**, and this point is a minimum.
- Intuitively, a function **f (1D case)** is convex on the range $[a,b]$ if a function's **second derivative is positive** every-where in that range.
- Intuitively, if a multivariate function's **Hessians is psd (positive semi-definite!)**, this (multivariate) function is Convex
 - Intuitively, we can think “Positive definite” matrices as analogy to positive numbers in matrix case

Review: Derivative of a Quadratic Function



$$y = x^2 - 3$$

$$y' = 2x$$

$$y'' = 2$$

This convex function is minimized @ the unique point whose derivative (slope) is zero.
→ If finding zeros of the derivative of this function, we can also find minima (or maxima) of that function.

Extra: Loss $J(\theta)$ is Convex

$$\Rightarrow J(\theta) = \frac{1}{2} (\theta^T X^T X \theta - 2\theta^T X^T y + y^T y)$$

$$\Rightarrow \text{Hessian } (J(\theta)) = X^T X$$

Gram matrix
PSD

$J(\theta)$ is convex

If $\nabla J(\theta^*) = 0$, $J(\theta)$ is minimized @ θ^*

Extra: positive semi-definite!

$A \in \mathbb{R}^{n \times n}, \forall x \in \mathbb{R}^n$

L2-Note: Page 17

If $x^T A x \geq 0$

($x \in \mathbb{R}^n$)

$\Rightarrow A$ is positive semi-definite (PSD)

If $x^T A x > 0$

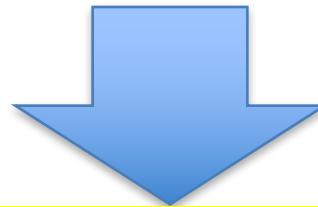
$\Rightarrow A$ is PD \Rightarrow full rank / invertible

See proof on L2-Note: Page 18

Extra: Gram Matrix $G = X^T X$
is always **positive semi-definite!**

Because for any vector a

$$a^T X^T X a = \|Xa\|_2^2 \geq 0$$



Besides, when X is full rank, G is invertible

Extra: Hessian Matrix

Derivatives and Second Derivatives

Cost function

$$J(\boldsymbol{\theta})$$

Gradient

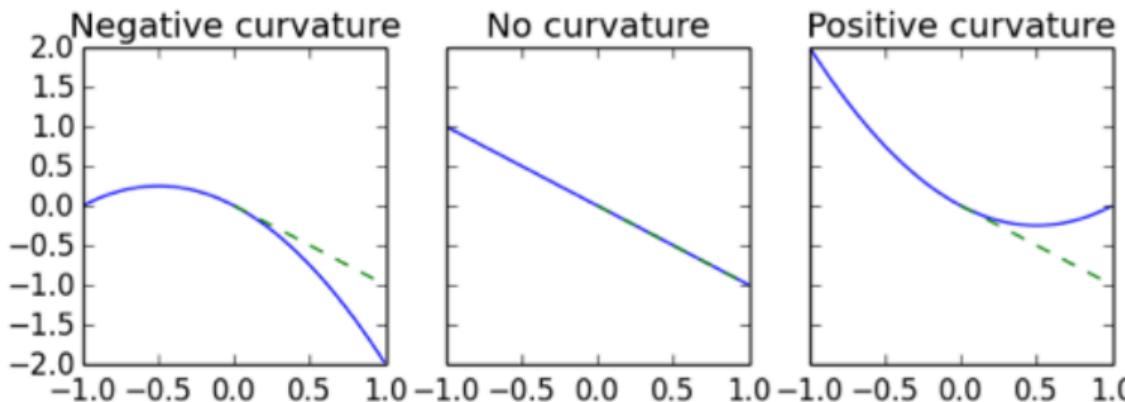
$$\mathbf{g} = \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$$

Hessian

$$\mathbf{H}$$

$$g_i = \frac{\partial}{\partial \theta_i} J(\boldsymbol{\theta})$$

$$H_{i,j} = \frac{\partial}{\partial \theta_j} g_i$$



$\mathbf{H} \text{ PD}$
for positive
curvature



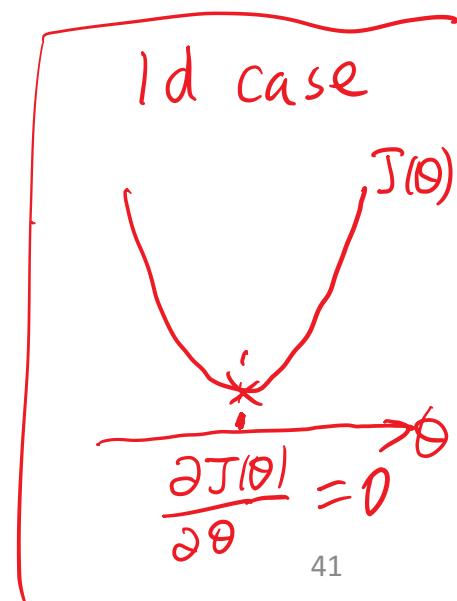
$$\begin{aligned}
 J(\theta) &= (\underline{x}\theta - y)^T (\underline{x}\theta - y) \frac{1}{2} \\
 &= ((x\theta)^T - y^T)(\underline{x}\theta - y) \frac{1}{2} \\
 &= (\theta^T x^T - y^T)(\underline{x}\theta - y) \frac{1}{2} \\
 &= (\theta^T \underline{x}^T \underline{x}\theta - \underbrace{\theta^T x^T y}_\cdot - \underbrace{y^T \underline{x}\theta}_\cdot + y^T y) \frac{1}{2}.
 \end{aligned}$$

Since $\theta^T x^T y = y^T \underline{x}\theta$

$$\langle x\theta, y \rangle \quad \langle y, \underline{x}\theta \rangle$$

$$= (\underbrace{\theta^T \underline{x}^T \underline{x}\theta}_\cdot - 2\underbrace{\theta^T x^T y}_\cdot + y^T y) \frac{1}{2}$$

$\Rightarrow J(\theta)$ quadratic func of θ ;



See handout 4.1 + 4.3 \Rightarrow matrix calculus, partial deri \Rightarrow Gradient

$$\nabla_{\theta} (\theta^T X^T X \theta) = 2X^T X \theta \quad (\text{P24})$$

$$\nabla_{\theta} (-2\theta^T X^T Y) = -2X^T Y \quad (\text{P24})$$

$$\nabla_{\theta} (Y^T Y) = 0$$

$$\Rightarrow \nabla_{\theta} J(\theta) = \boxed{X^T X \theta - X^T Y}$$

gram matrix is PSD

if X full rank, $X^T X$ PD \Rightarrow invert

$$\Rightarrow \theta = \underbrace{(X^T X)^{-1}}_{P \times P} \underbrace{X^T Y}_{P \times 1} \Rightarrow P \times 1$$

Our loss function's Hessian is Positive Semi-definite

Comments on the normal equation

- In most situations of practical interest, the number of data points n is larger than the dimensionality p of the input space and the matrix \mathbf{X} is of full column rank. If this condition holds, then it is easy to verify that $\mathbf{X}^T \mathbf{X}$ is necessarily invertible.
 $n \gg p$
- The assumption that $\mathbf{X}^T \mathbf{X}$ is invertible implies that it is positive definite, thus the critical point we have found is a minimum.
- What if \mathbf{X} has less than full column rank? → regularization (later).

Extra: Scalability to big data?

- Traditional CS view: Polynomial time algorithm,
Wow!
- Large-scale learning: Sometimes even $O(n)$ is bad!
 \Rightarrow Many state-of-the-art solutions (e.g., low rank, sparse, hardware, sampling, randomized...)

Simple example: Matrix multiplication

$$\begin{matrix} n & \begin{matrix} \text{[Blue square]} \end{matrix} & \times & \begin{matrix} \text{[Blue square]} \end{matrix} & = & \begin{matrix} \text{[Blue square]} \end{matrix} & O(n^3)! \\ & \text{n} & & & & & \end{matrix}$$

The following complexity figures assume that arithmetic with individual elements has complexity $O(1)$, as is the case with fixed-precision operations on a finite field.

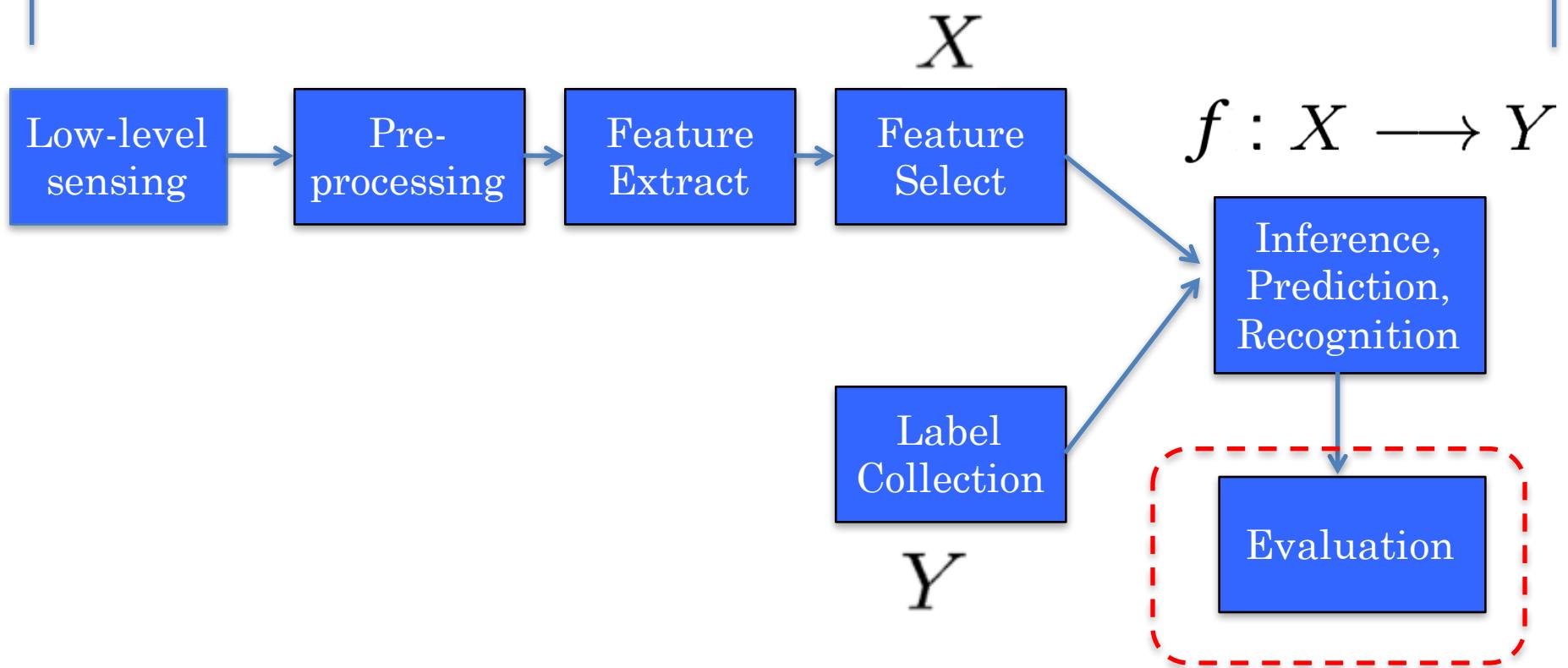
From Wiki

Operation	Input	Output	Algorithm	Complexity
Matrix multiplication	Two $n \times n$ matrices	One $n \times n$ matrix	Schoolbook matrix multiplication	$O(n^3)$
			Strassen algorithm	$O(n^{2.807})$
			Coppersmith–Winograd algorithm	$O(n^{2.376})$
			Optimized CW-like algorithms ^{[14][15][16]}	$O(n^{2.373})$
Matrix multiplication	One $n \times m$ matrix & one $m \times p$ matrix	One $n \times p$ matrix	Schoolbook matrix multiplication	$O(nmp)$
Matrix inversion*	One $n \times n$ matrix	One $n \times n$ matrix	Gauss–Jordan elimination	$O(n^3)$
			Strassen algorithm	$O(n^{2.807})$
			Coppersmith–Winograd algorithm	$O(n^{2.376})$
			Optimized CW-like algorithms	$O(n^{2.373})$
Singular value decomposition	One $m \times n$ matrix	One $m \times m$ matrix, one $m \times n$ matrix, & one $n \times n$ matrix		$O(mn^2)$ ($m \leq n$)
		One $m \times r$ matrix, one $r \times r$ matrix, & one $n \times r$ matrix		
Determinant	One $n \times n$ matrix	One number	Laplace expansion	$O(n!)$
			Division-free algorithm ^[17]	$O(n^4)$
			LU decomposition	$O(n^3)$
			Bareiss algorithm	$O(n^3)$
			Fast matrix multiplication ^[18]	$O(n^{2.373})$
Back substitution	Triangular matrix	n solutions	Back substitution ^[19]	$O(n^2)$

Today

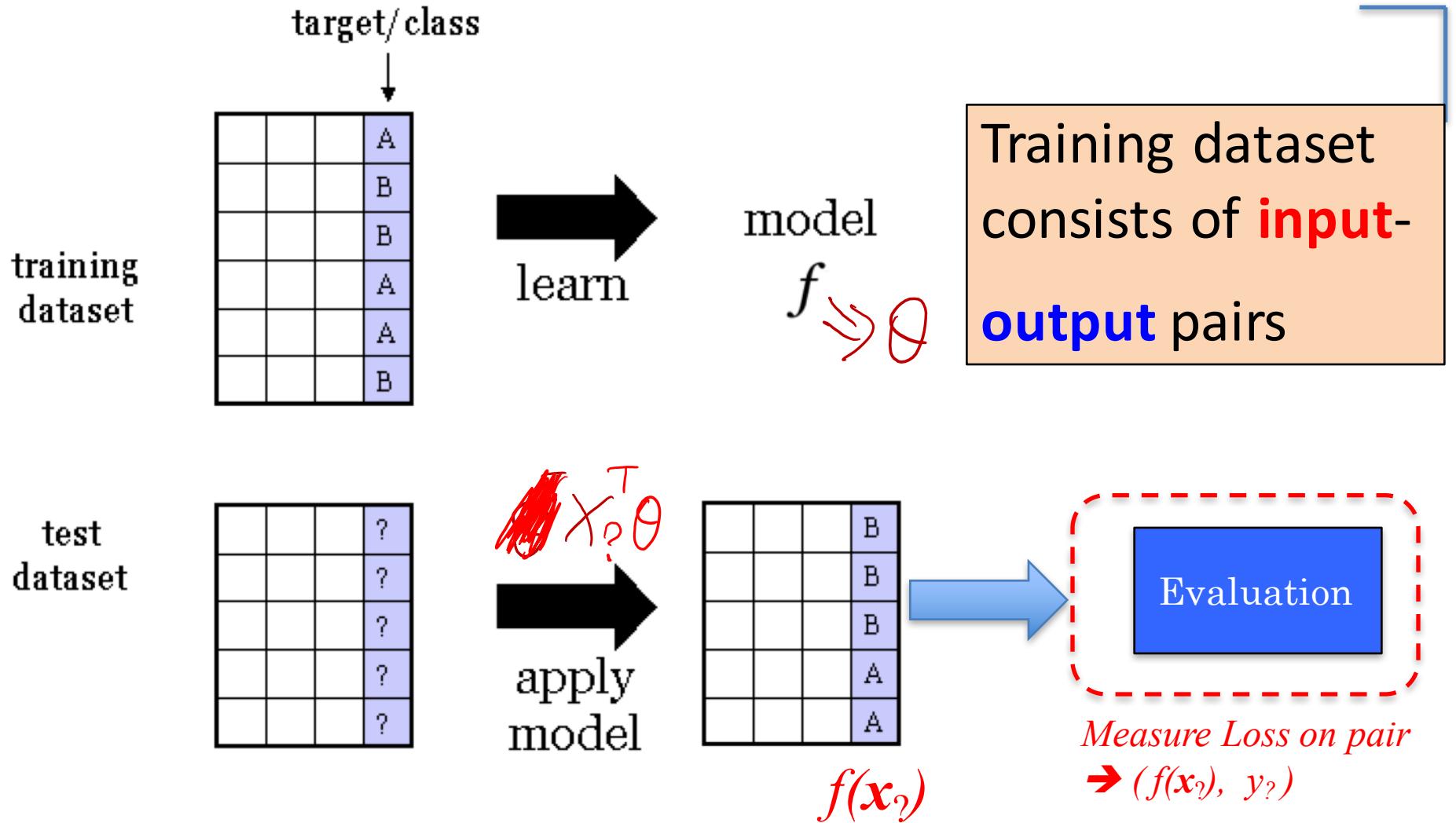
- ❑ Linear regression (aka **least squares**)
- ❑ Learn to derive the least squares estimate by optimization
- ❑ Evaluation with Train/Test OR k-folds Cross-validation

TYPICAL MACHINE LEARNING SYSTEM



Evaluation Choice-I:

Train and Test



Evaluation Choice-I:

e.g. for linear regression models

training dataset

$$\mathbf{X}_{train} = \begin{bmatrix} \cdots & \mathbf{x}_1^T & \cdots \\ \cdots & \mathbf{x}_2^T & \cdots \\ \vdots & \vdots & \vdots \\ \cdots & \mathbf{x}_n^T & \cdots \end{bmatrix} \quad \bar{\mathbf{y}}_{train} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$$

test dataset

$$\mathbf{X}_{test} = \begin{bmatrix} \cdots & \mathbf{x}_{n+1}^T & \cdots \\ \cdots & \mathbf{x}_{n+2}^T & \cdots \\ \vdots & \vdots & \vdots \\ \cdots & \mathbf{x}_{n+m}^T & \cdots \end{bmatrix} \quad \bar{\mathbf{y}}_{test} = \begin{bmatrix} y_{n+1} \\ y_{n+2} \\ \vdots \\ y_{n+m} \end{bmatrix}$$

Evaluation Choice-I:

e.g. for linear regression models

- Training SSE (sum of squared error):

$$J_{train}(\theta) = \frac{1}{2} \sum_{i=1}^n (\mathbf{x}_i^T \theta - y_i)^2$$

- Minimize $J_{train}(\theta)$ \rightarrow Normal Equation to get

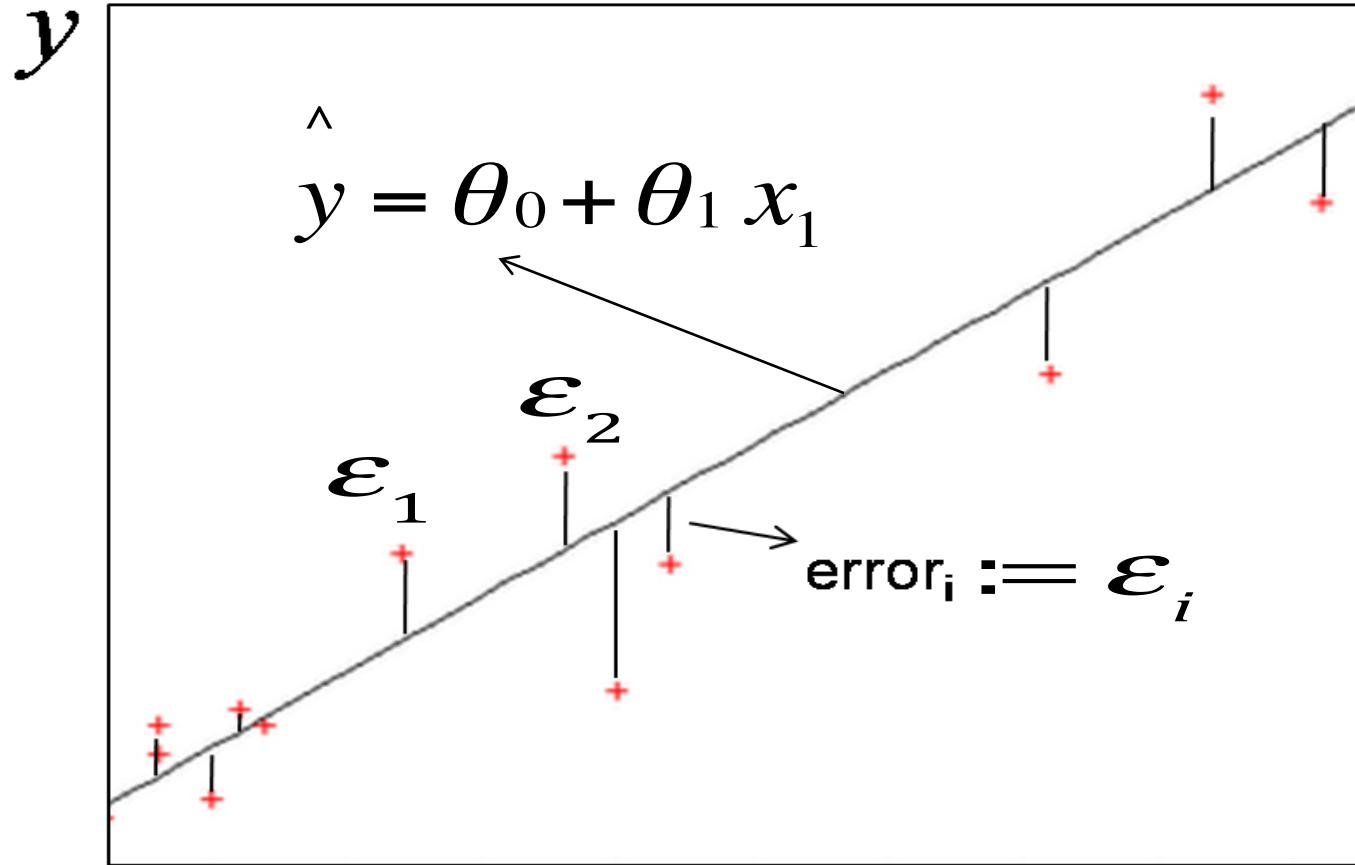
$$\theta^* = \operatorname{argmin} J_{train}(\theta) = \left(X_{train}^T X_{train} \right)^{-1} X_{train}^T \vec{y}_{train}$$

Evaluation Choice-I: e.g. for Regression Models

- Testing MSE Error to report:

$$J_{test} = \frac{1}{m} \sum_{i=n+1}^{n+m} (\mathbf{x}_i^T \boldsymbol{\theta}^* - y_i)^2 = \frac{1}{m} \sum_{i=n+1}^{n+m} \varepsilon_i^2$$

Linear regression (1D example)



$$\theta^* = (X^T X)^{-1} X^T \bar{y}$$

Evaluation Choice-II: Cross Validation

- Problem: don't have enough data to set aside a test set
- Solution: Each data point is used both as train and test
- Common types:
 - K-fold cross-validation (e.g. K=5, K=10)
 - 2-fold cross-validation
 - Leave-one-out cross-validation
(LOOCV, i.e., k=n_reference)

K-fold Cross Validation

- Basic idea:
 - Split the whole data to N pieces;
 - N-1 pieces for fit/train model; 1 for test;
 - Cycle through all N folds;
 - K=10 “folds” a common rule of thumb.
- The advantage:
 - all pieces are used for both training and validation;
 - each observation is used for validation exactly once.

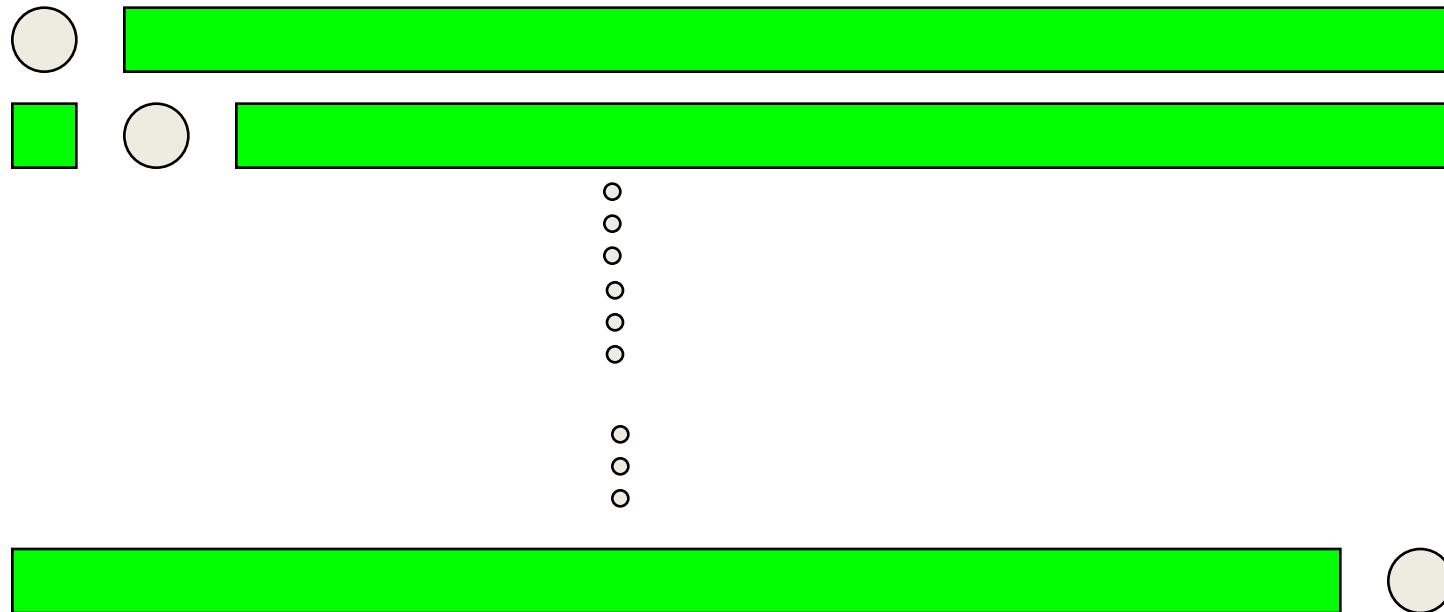
e.g. k=10 fold Cross Validation

- Divide data into 10 equal pieces
- 9 pieces as training set, the rest 1 as test set
- Collect the scores from the diagonal
- We normally use the mean of the scores

model	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10
1	train	test								
2	train	test	train							
3	train	test	train	train						
4	train	train	train	train	train	train	test	train	train	train
5	train	train	train	train	train	test	train	train	train	train
6	train	train	train	train	test	train	train	train	train	train
7	train	train	train	test	train	train	train	train	train	train
8	train	train	test	train						
9	train	test	train							
10	test	train								

e.g. (k=n-1) Leave-one-out / LOOCV (n-fold cross validation)

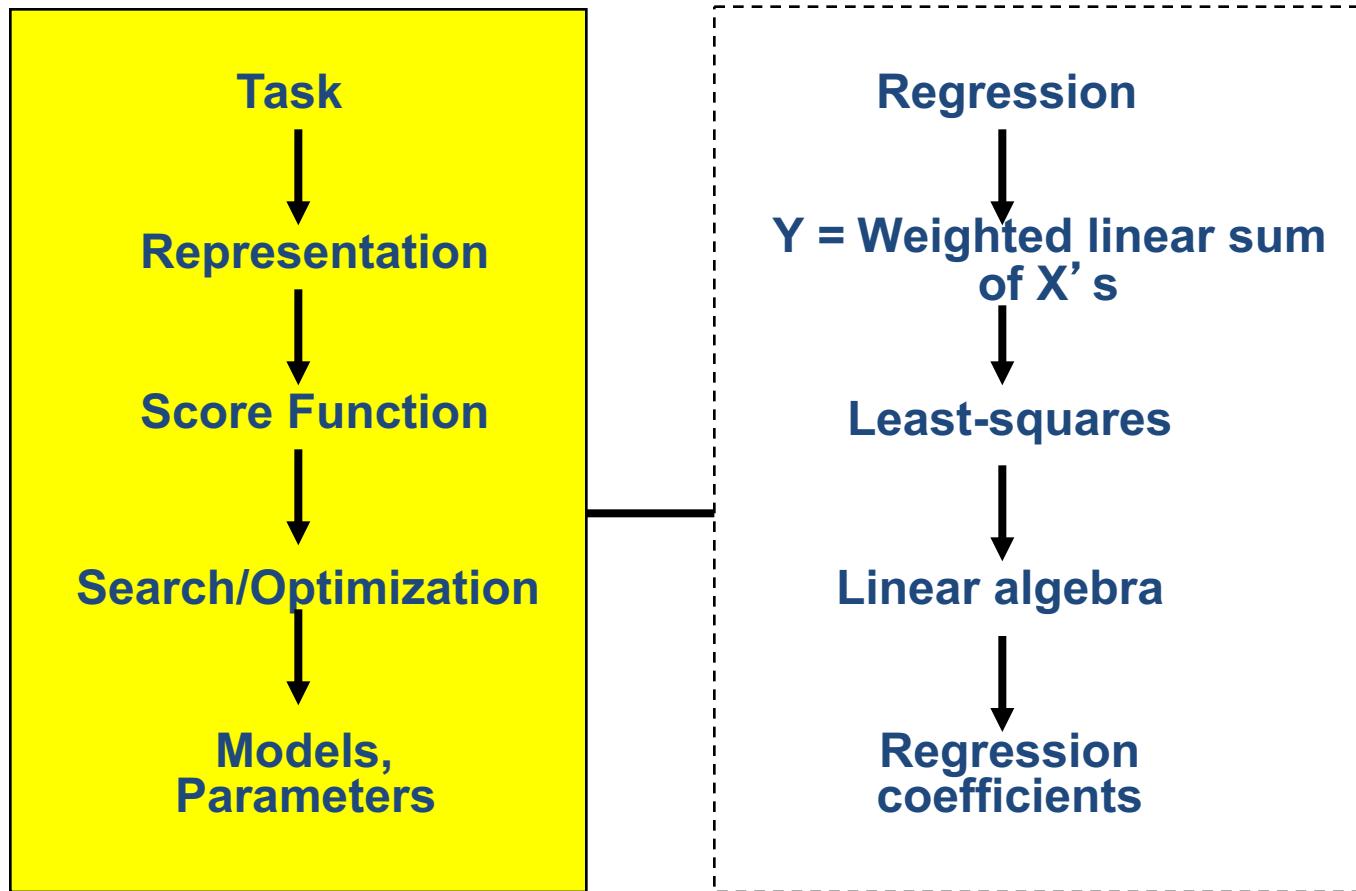
k is num. of data samples



Today Recap

- ❑ Linear regression (aka **least squares**)
- ❑ Learn to derive the least squares estimate by normal equation
- ❑ Evaluation with Train/Test OR k-folds Cross-validation

(1) Multivariate Linear Regression



$$\hat{y} = f(x) = \theta^T x$$

References

- Big thanks to Prof. Eric Xing @ CMU for allowing me to reuse some of his slides
 - ❑ <http://www.cs.cmu.edu/~zkolter/course/15-884/linalg-review.pdf> (please read)
 - ❑ Prof. Alexander Gray's slides