UVA CS 6316/4501 – Fall 2016 Machine Learning

Lecture 3: Linear Regression

Dr. Yanjun Qi

University of Virginia

Department of
Computer Science

HW1 OUT / DUE NEXT SAT

Where are we ? → Five major sections of this course

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

Today → Regression (supervised)

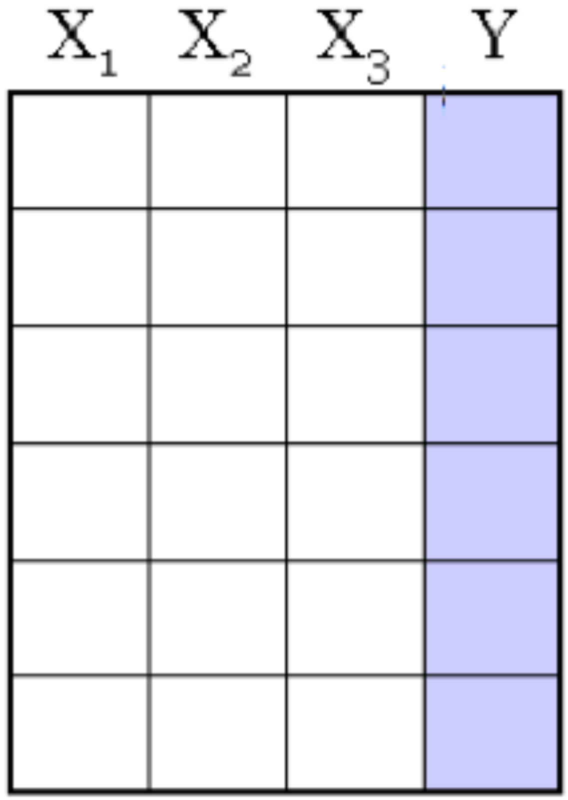
- Four ways to train / perform optimization for linear regression models
  - Normal Equation
  - Gradient Descent (GD)
  - Stochastic GD
  - Newton's method

- 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 Dataset for regression



$$f : X \longrightarrow Y$$

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]

For Example,

Machine learning for apartment hunting

- Now you've moved to Charlottesville !!

  And you want to find the **most reasonably priced** apartment satisfying your **needs:**

  square-ft., # of bedroom, distance to campus …

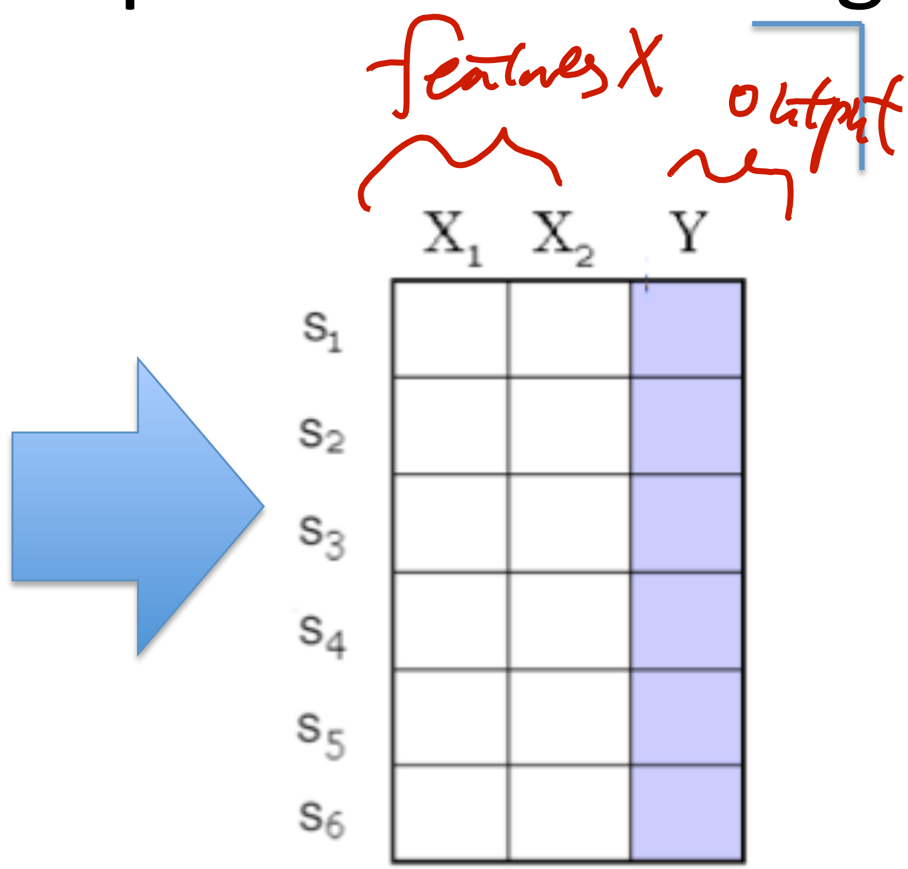
Living area (ft^2)	# bedroom	Rent ($)
230	1	600
506	2	1000
433	2	1100
109	1	500
…		
150	1	?
270	1.5	?

For Example,

Machine learning for apartment hunting

features — output

Living area (ft^2)	# bedroom	Rent ($)
230	1	600
506	2	1000
433	2	1100
109	1	500
…		
150	1	?
270	1.5	?



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**:

Living area, distance to campus, # bedroom …

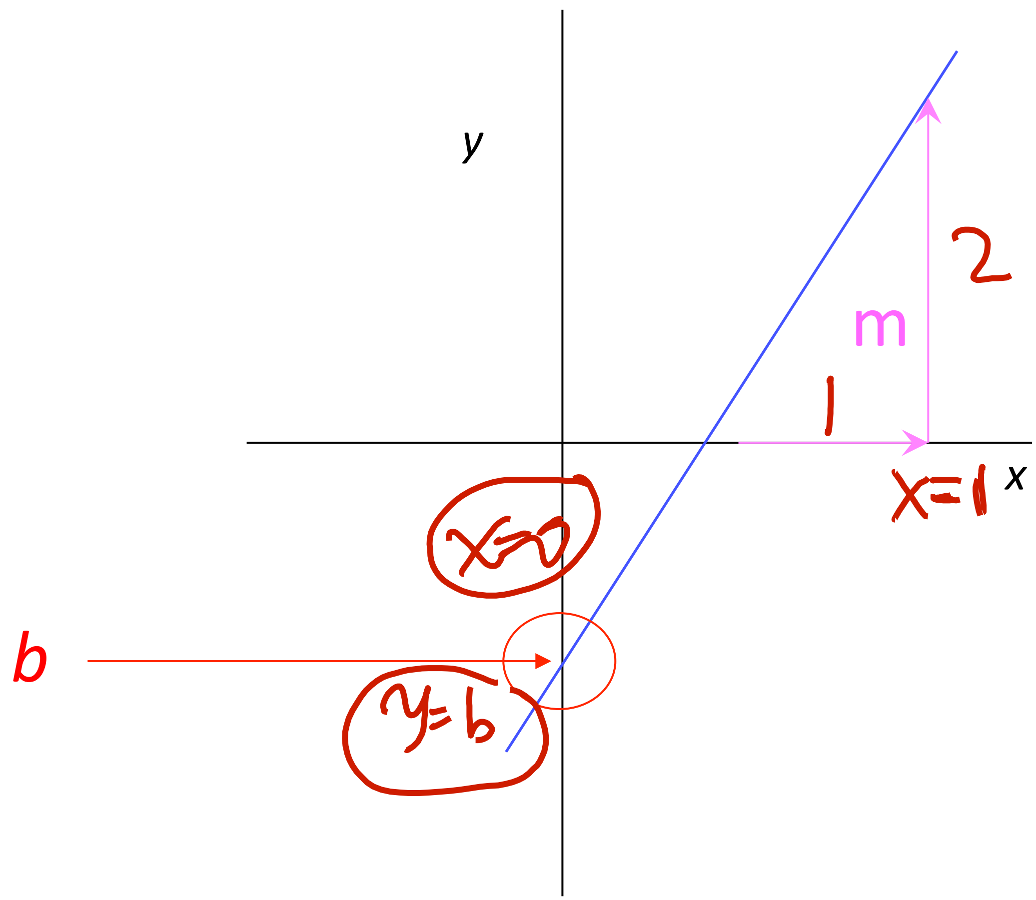
=> Target y:

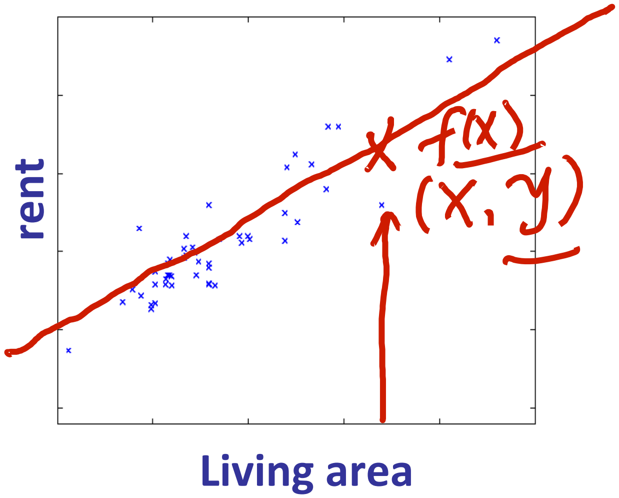
Rent ➔ Continuous

Remember this: "Linear"? (1D case)

- *y=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.





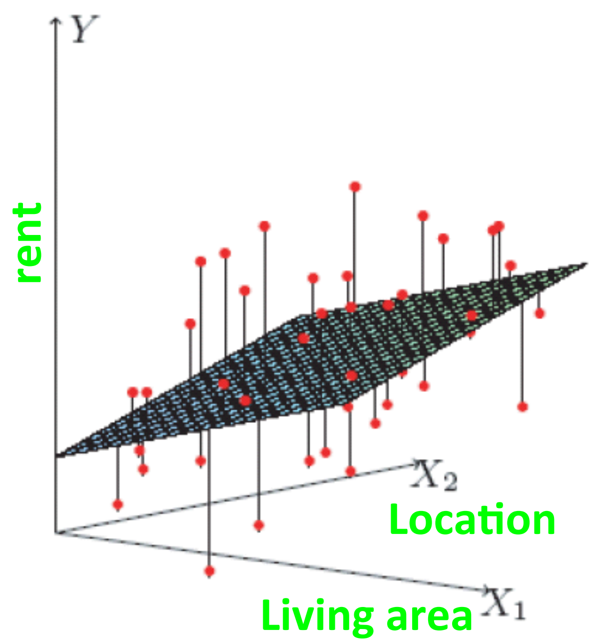
$$\sum_i (y_i - f(x_i))^2$$

$\hat{y}_i$

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

$$f(x) = m x + b$$

1×1 $\quad$ 1×1



$$\hat{y}_i = f(x) = \theta^0 + \theta^1 x_i^1 + \theta^2 x_i^2$$

$\Downarrow$

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

$$= \theta^T x_i$$

$$= x_i^T \theta$$

Review: Special Uses for Matrix Multiplication

- Dot (or Inner) Product of two Vectors <x, y>

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

<x, y> = <y, x> $\mathbf{a}^T\mathbf{b} = \mathbf{b}^T\mathbf{a}$

Where <x, y> = $x^T y \in \mathbb{R} = \begin{bmatrix} x_1 & x_2 & \cdots & x_n \end{bmatrix} \begin{bmatrix} y_1 \\ x_2 \\ \vdots \\ y_n \end{bmatrix} = \sum_{i=1}^{n} x_i y_i.$

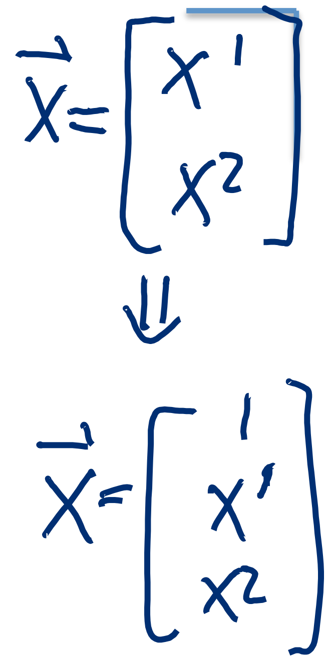
A new representation (for single sample)

- Assume that each sample **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:

$$\mathbf{x}^T=[x^0, x^1, x^2, \ldots x^{p-1}]$$

  - the parameter vector θ is also a column vector

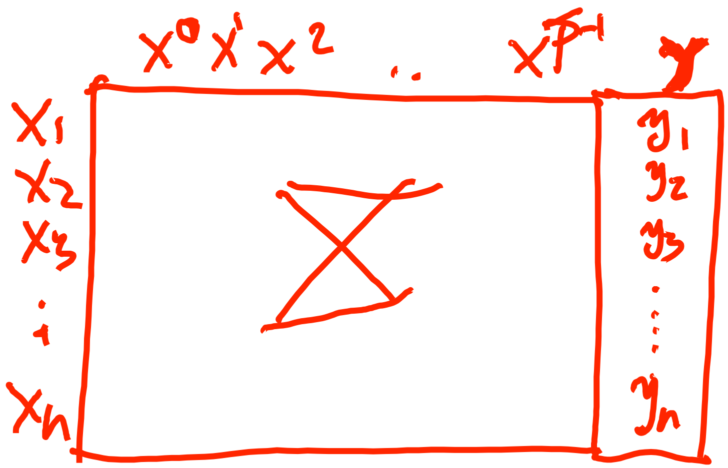
$$\theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \vdots \\ \theta_{p-1} \end{bmatrix}$$



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

$$\hat{y} = f(\mathbf{x}) = \theta_0 + \theta_1 x^{1} + \theta_2 x^{2} + \ldots + \theta_{p-1} x^{p-1}$$

$$x_1, x_2, x_3, \ldots, x_n \Rightarrow \begin{bmatrix} x_1^T\theta \\ x_2^T\theta \\ \vdots \\ x_n^T\theta \end{bmatrix} = X\theta$$



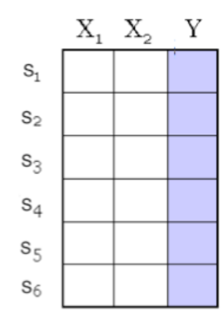
Training / learning problem

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

features

data index

$$
\mathbf{X} = \begin{bmatrix} -- & \mathbf{x}_1^T & -- \\ -- & \mathbf{x}_2^T & -- \\ \vdots & \vdots & \vdots \\ -- & \mathbf{x}_n^T & -- \end{bmatrix} = \begin{bmatrix} x_1^0 & x_1^1 & \dots & x_1^{p-1} \\ x_2^0 & x_2^1 & \dots & x_2^{p-1} \\ \vdots & \vdots & \vdots & \vdots \\ x_n^0 & x_n^1 & \dots & x_n^{p-1} \end{bmatrix}
\qquad
\mathbf{Y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}
$$



REVIEW: Special Uses for Matrix Multiplication

- Matrix-Vector Products (I)

Given a matrix $A \in \mathbb{R}^{m \times n}$ and a vector $x \in \mathbb{R}^n$, their product is a vector $y = Ax \in \mathbb{R}^m$.

If we write A by rows, then we can express Ax as,

$$y = Ax = \begin{bmatrix} — & a_1^T & — \\ — & a_2^T & — \\ & \vdots & \\ — & a_m^T & — \end{bmatrix} x = \begin{bmatrix} a_1^T x \\ a_2^T x \\ \vdots \\ a_m^T x \end{bmatrix}.$$

$$\begin{bmatrix} — & x_1^T & — \\ — & x_2^T & — \\ & \vdots & \\ — & x_m^T & — \end{bmatrix} \theta = X\theta$$

Training / learning problem

- 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 n×1

  - Labels (given output value)

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

n×1

Training / learning goal

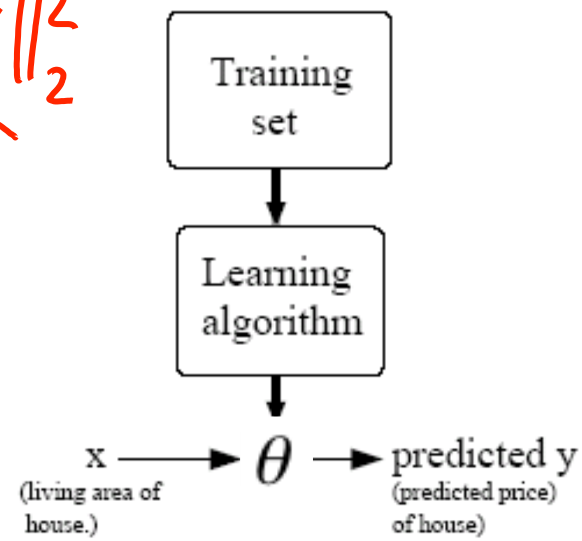
- Using matrix form, we get the following general representation of the linear regression function:

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

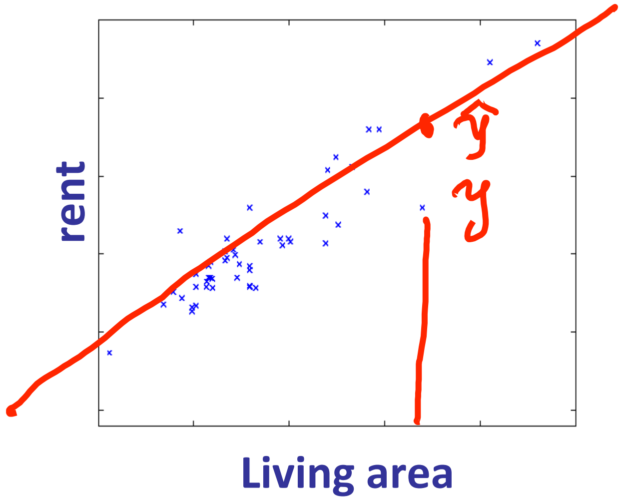
- Our goal is to pick the optimal θ that minimize the following cost function:

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

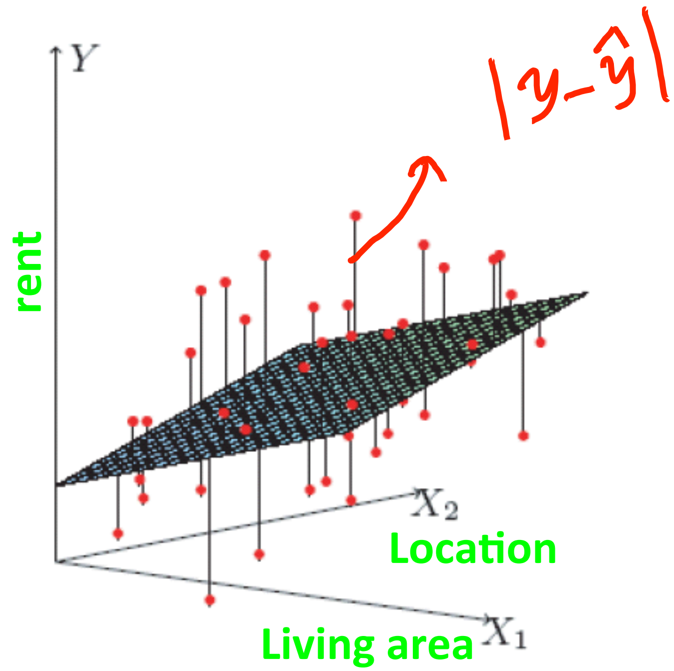
Our goal:



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:

$$J(\theta) = \frac{1}{2}\sum_{i=1}^{n}(\mathbf{x}_i{}^T\theta - y_i)^2$$
$$= \frac{1}{2}(X\theta - \vec{y})^T(X\theta - \vec{y})$$
$$= \frac{1}{2}\left(\theta^T X^T X\theta - \theta^T X^T \vec{y} - \vec{y}^T X\theta + \vec{y}^T \vec{y}\right)$$

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

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

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

The normal equations

$$\Downarrow$$

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

WHY ??

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

Review: Special Uses for Matrix Multiplication

- Sum the Squared Elements of a Vector → L2 norm 2
  - Premultiply a column vector **a** by its transpose – If

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

then premultiplication by a row vector $\mathbf{a}^T$

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

will yield the sum of the squared values of elements for **a**, i.e.

$$|a|_2^2 = a^T\mathbf{a}=\begin{bmatrix}5 & 2 & 8\end{bmatrix}\begin{bmatrix}5\\2\\8\end{bmatrix}=5^2+2^2+8^2=93$$

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

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

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

$n \times p \quad p \times 1 \quad n \times 1$

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

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

$a^T b = b^T a$

$\Rightarrow \theta^T X^T y = y^T X \theta$

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

$\Rightarrow \text{Hessian}(J(\theta)) = X^T X$ → Gram matrix ⇓ [PSD]

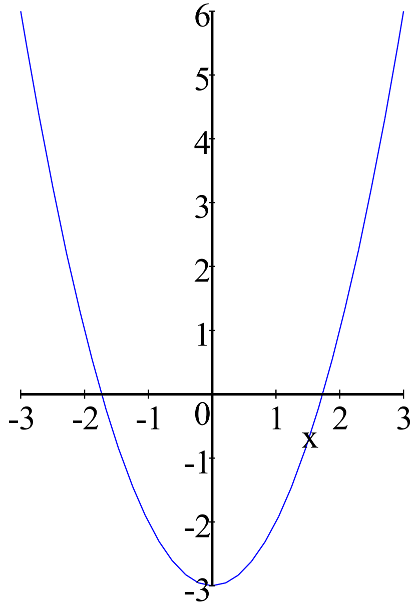
⇓

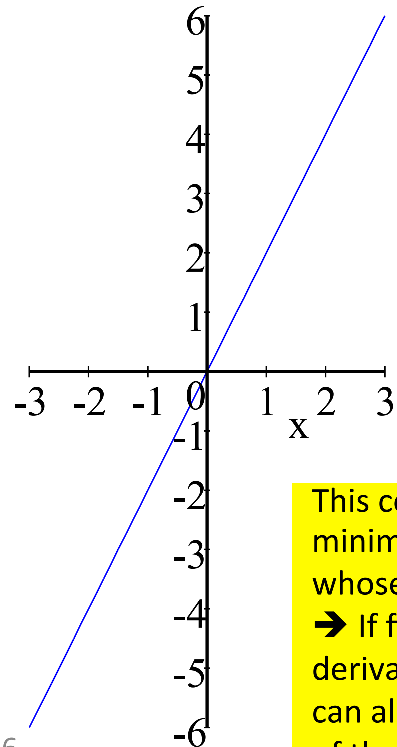
$J(\theta)$ is convex

⇓

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

Review: Derivative of a Quadratic Function





$$y = x^2 - 3$$

$$y' = \lim_{h \to 0} \frac{(x+h)^2 - 3 - (x^2 - 3)}{h}$$

$$y' = \lim_{h \to 0} \frac{x^2 + 2xh + h^2 - x^2}{h}$$

$$y' = \lim_{h \to 0} 2x + h$$

0

$$y' = 2x$$

$$y'' = 2$$

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.

Review: 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 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: positive semi-definite!

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

If $x^T A x \geq 0$
($1 \times n$, $n \times n$, $n \times 1$)

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

If $x^T A x > 0$

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

Gram is always PSD

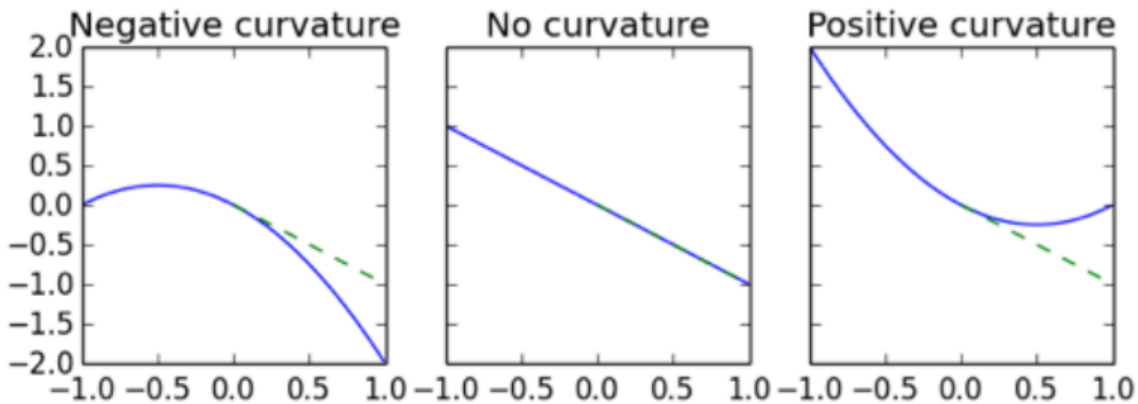
$G = X^T X$ ($p \times n$, $n \times p$)

if X full rank matrix, then G is PD

Extra: Hessian

Derivatives and Second Derivatives

Cost function	Gradient	Hessian
$J(\boldsymbol{\theta})$	$\boldsymbol{g} = \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$	$\boldsymbol{H}$
	$g_i = \frac{\partial}{\partial \theta_i} J(\boldsymbol{\theta})$	$H_{i,j} = \frac{\partial}{\partial \theta_j} g_i$



H PD for positive curvature

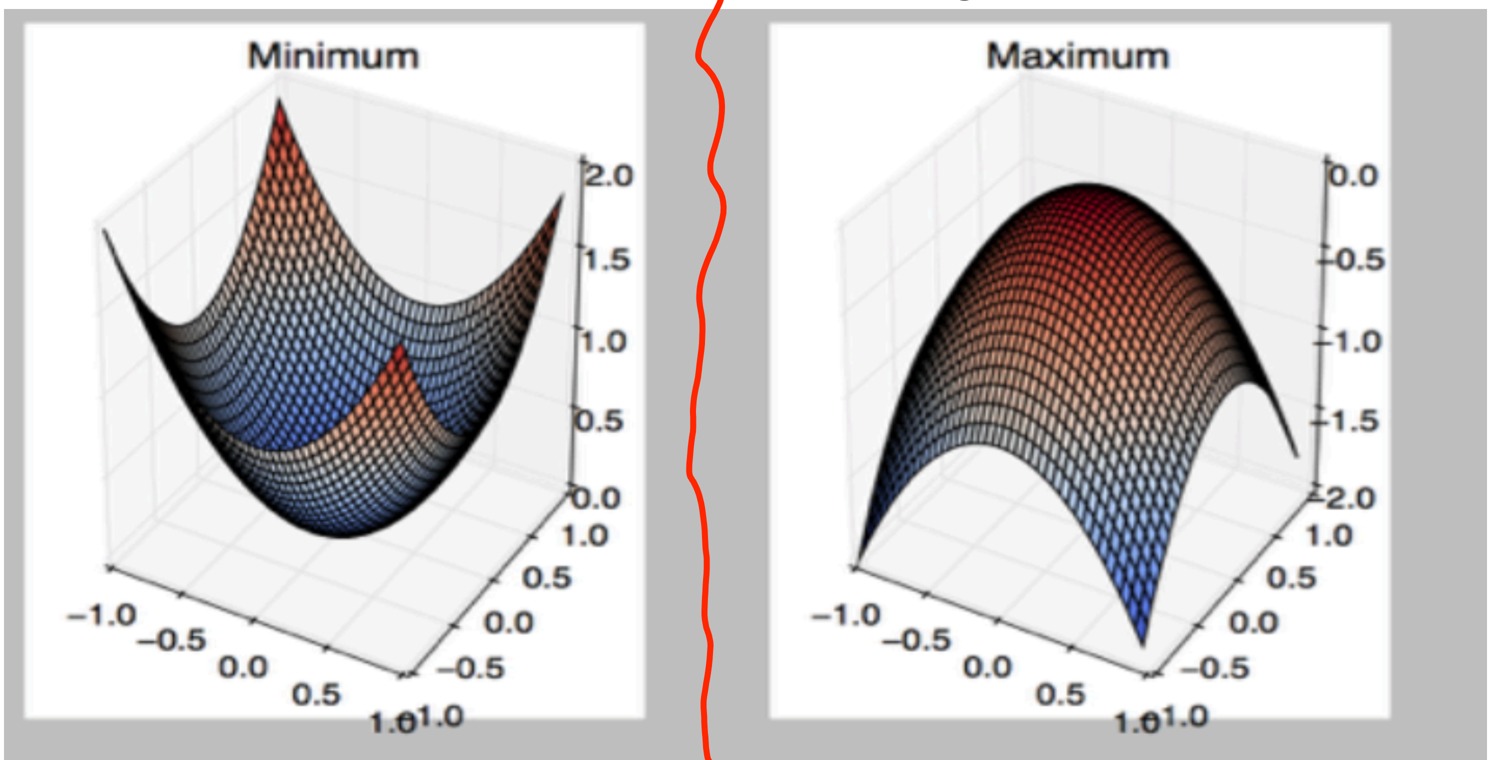
Review: Matrix Calculus: Types of Matrix Derivatives

	Scalar	Vector	Matrix
Scalar	$\frac{\mathrm{d}y}{\mathrm{d}x}$	$\frac{\mathrm{d}\mathbf{y}}{\mathrm{d}x} = \left[\frac{\partial y_i}{\partial x}\right]$	$\frac{\mathrm{d}\mathbf{Y}}{\mathrm{d}x} = \left[\frac{\partial y_{ij}}{\partial x}\right]$
Vector	$\frac{\mathrm{d}y}{\mathrm{d}\mathbf{x}} = \left[\frac{\partial y}{\partial x_j}\right]$	$\frac{\mathrm{d}\mathbf{y}}{\mathrm{d}\mathbf{x}} = \left[\frac{\partial y_i}{\partial x_j}\right]$	
Matrix	$\frac{\mathrm{d}y}{\mathrm{d}\mathbf{X}} = \left[\frac{\partial y}{\partial x_{ji}}\right]$		

By Thomas Minka. Old and New Matrix Algebra Useful for Statistics

Extra: Eigenvalues of Hessian → Curvature

Positive Definite Hessian | Negative Definite Hessian



All positive eigenvalues | All negative eigenvalues

Review: Some important rules for taking derivatives

- Scalar multiplication: $\partial_x[af(x)] = a[\partial_x f(x)]$
- Polynomials: $\partial_x[x^k] = kx^{k-1}$
- Function addition: $\partial_x[f(x) + g(x)] = [\partial_x f(x)] + [\partial_x g(x)]$
- Function multiplication: $\partial_x[f(x)g(x)] = f(x)[\partial_x g(x)] + [\partial_x f(x)]g(x)$
- Function division: $\partial_x \left[\frac{f(x)}{g(x)}\right] = \frac{[\partial_x f(x)]g(x) - f(x)[\partial_x g(x)]}{[g(x)]^2}$
- Function composition: $\partial_x[f(g(x))] = [\partial_x g(x)][\partial_x f](g(x))$
- Exponentiation: $\partial_x[e^x] = e^x$ and $\partial_x[a^x] = \log(a)e^x$
- Logarithms: $\partial_x[\log x] = \frac{1}{x}$

Review: Some important rules for taking gradient and hessian

- $$\frac{\partial \mathbf{x}^T \mathbf{a}}{\partial \mathbf{x}} = \frac{\partial \mathbf{a}^T \mathbf{x}}{\partial \mathbf{x}} = \mathbf{a}$$
- $\nabla_x x^T A x = 2Ax$ (if A symmetric)
- $\nabla_x^2 x^T A x = 2A$ (if A symmetric)

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

$$\Rightarrow \frac{\partial J(\theta)}{\partial \theta} = \frac{1}{2}\left(2X^T X\theta - 2X^T y\right) \overset{\text{set}}{\underset{\text{to}}{=}} 0$$

$$\Rightarrow \frac{\partial^2 J(\theta)}{\partial \theta^2} = X^T X \quad \text{[Hessian]}$$

$$\underbrace{X^T X}\theta = X^T y$$

gram matrix is PSD

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

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

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 $X^T X$ is necessarily invertible.

  $n >> p$

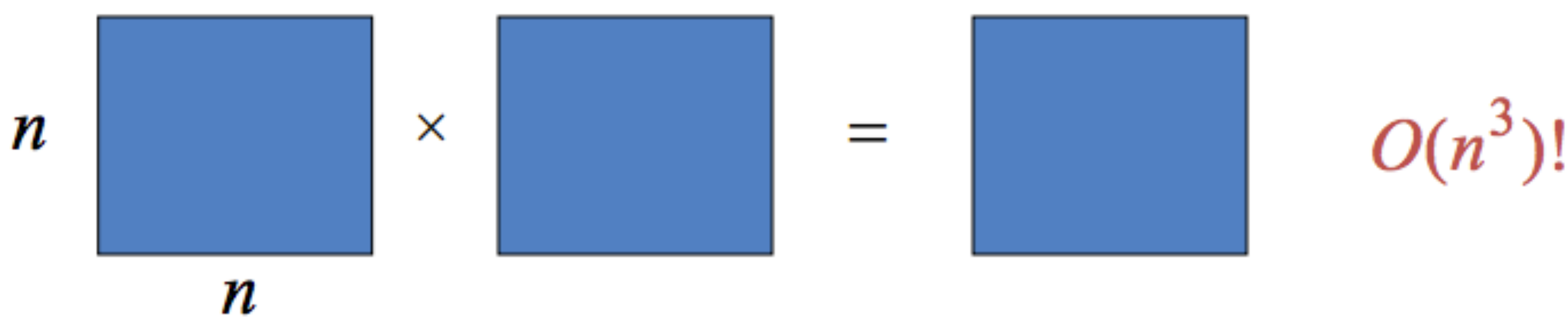
- The assumption that $X^T X$ is invertible implies that it is positive definite, thus the critical point we have found is a minimum.

- What if $\mathbf{X}$ has less than full column rank? → regularization (later).

Scalability to big ?

- Traditional CS view: Polynomial time algorithm, Wow!
- Large-scale learning: Sometimes even O(n) is bad!

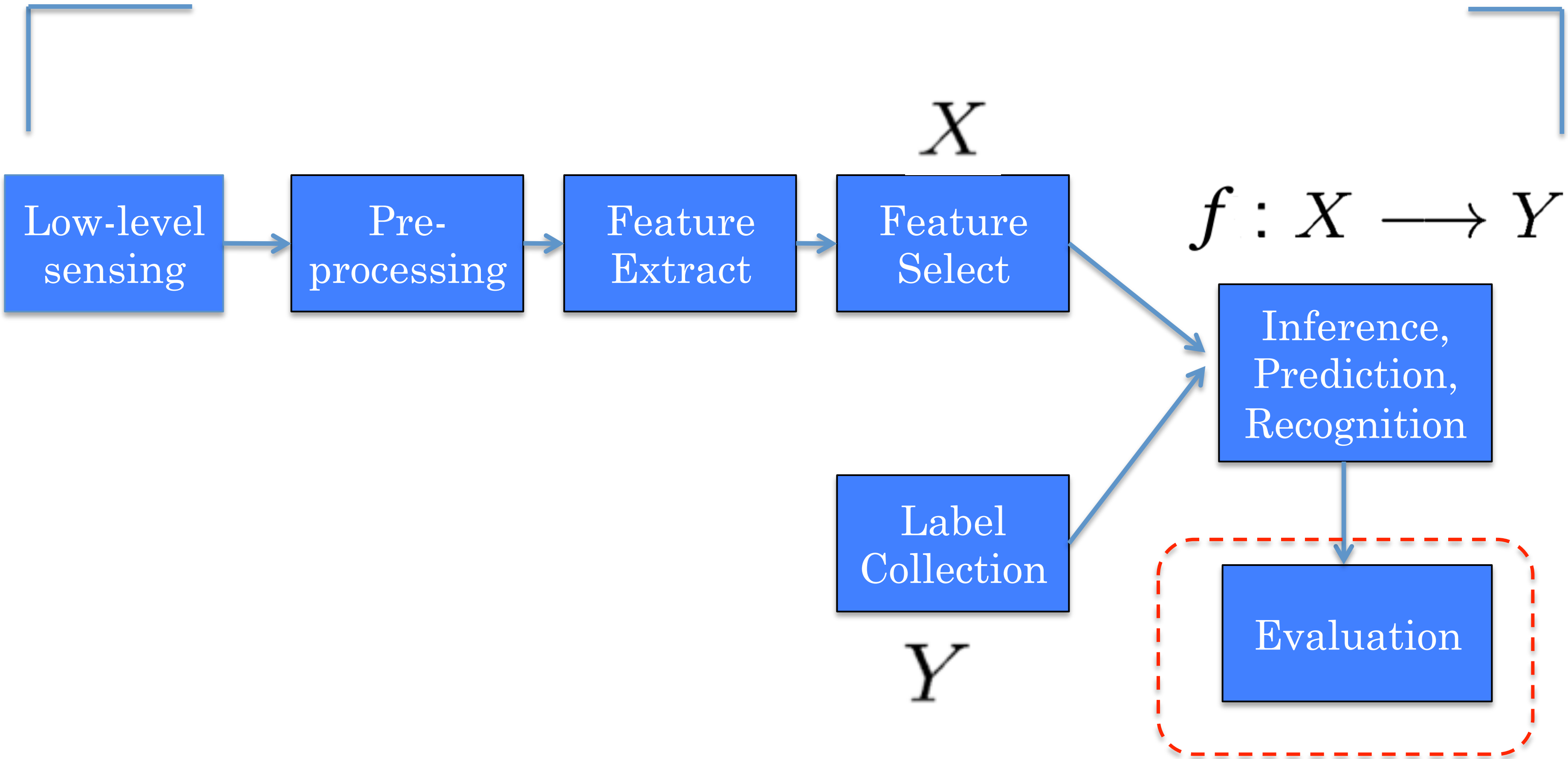
Simple example: Matrix multiplication



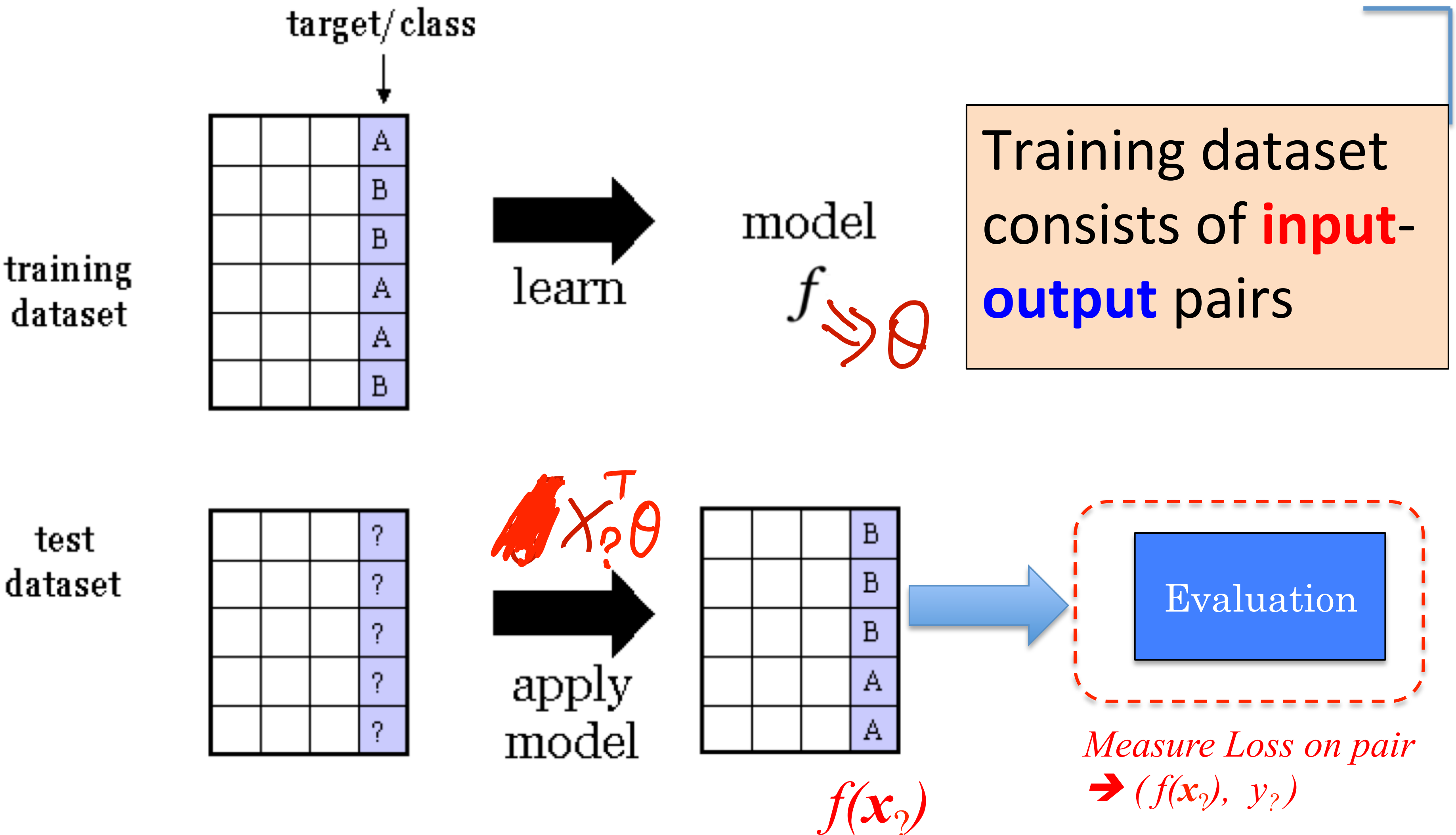
Today

- ❑ 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

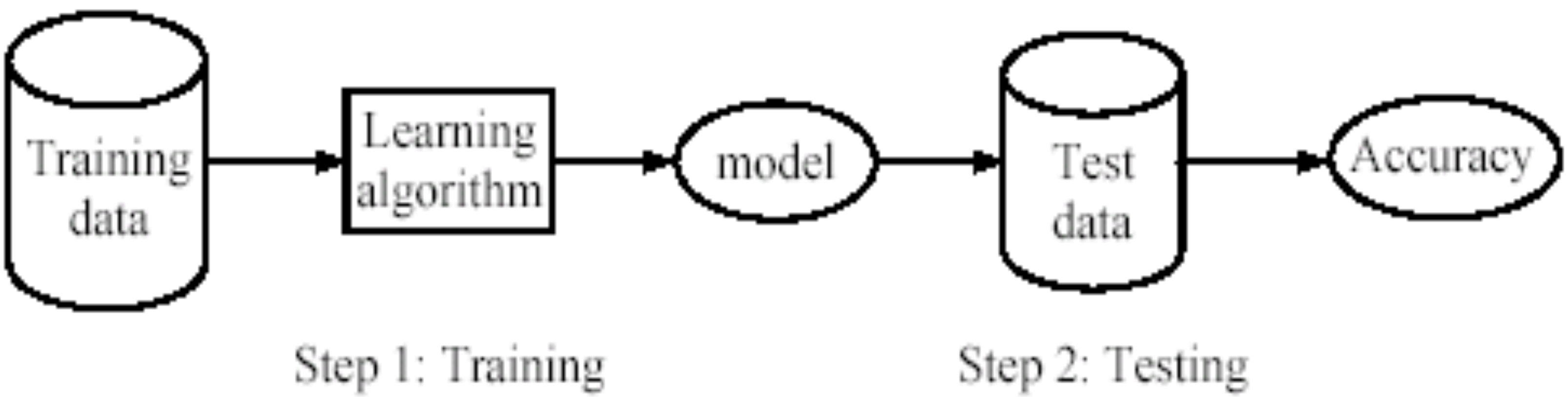


Evaluation Choice-I: Train and Test



Evaluation Choice-I: e.g. for supervised classification

- ✓ Training (Learning): Learn a model using the training data
- ✓ Testing: Test the model using unseen test data to assess the model accuracy



$$Accuracy = \frac{\text{Number of correct classifications}}{\text{Total number of test cases}},$$

Evaluation Choice-I: e.g. for linear regression models

training dataset

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

test dataset

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

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 $J_{train}(\theta)$ ➔ *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:

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

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 model; 1 for test;

  -Cycle through all N cases;

  -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.

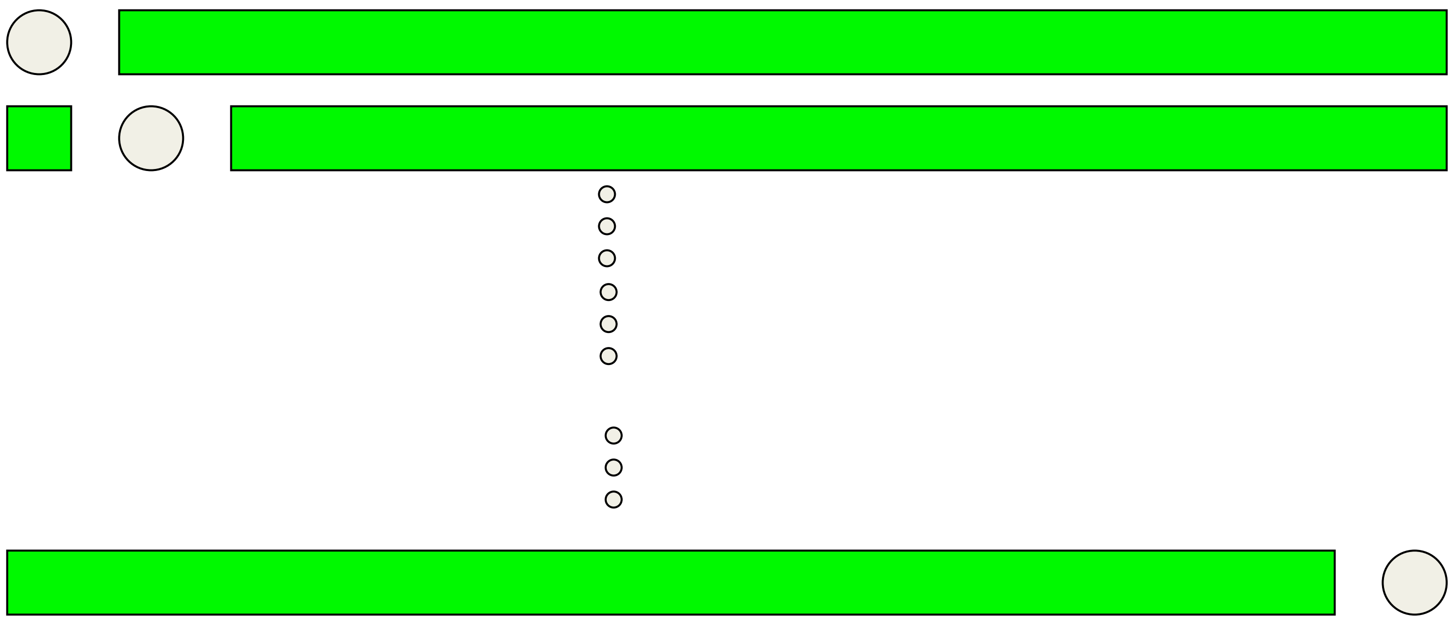
e.g. 10 fold Cross Validation

- 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	train	train	train	train	train	train	train	train	test
2	train	train	train	train	train	train	train	train	test	train
3	train	train	train	train	train	train	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	train	train	train	train	train	train
9	train	test	train	train	train	train	train	train	train	train
10	test	train	train	train	train	train	train	train	train	train

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

n 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

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