

Introduction to Machine Learning – Fundamentals of Machine Learning

The 8th KIAS CAC Summer School
2017. 7. 29 (Thu.)

Yung-Kyun Noh
Seoul National University



Overview

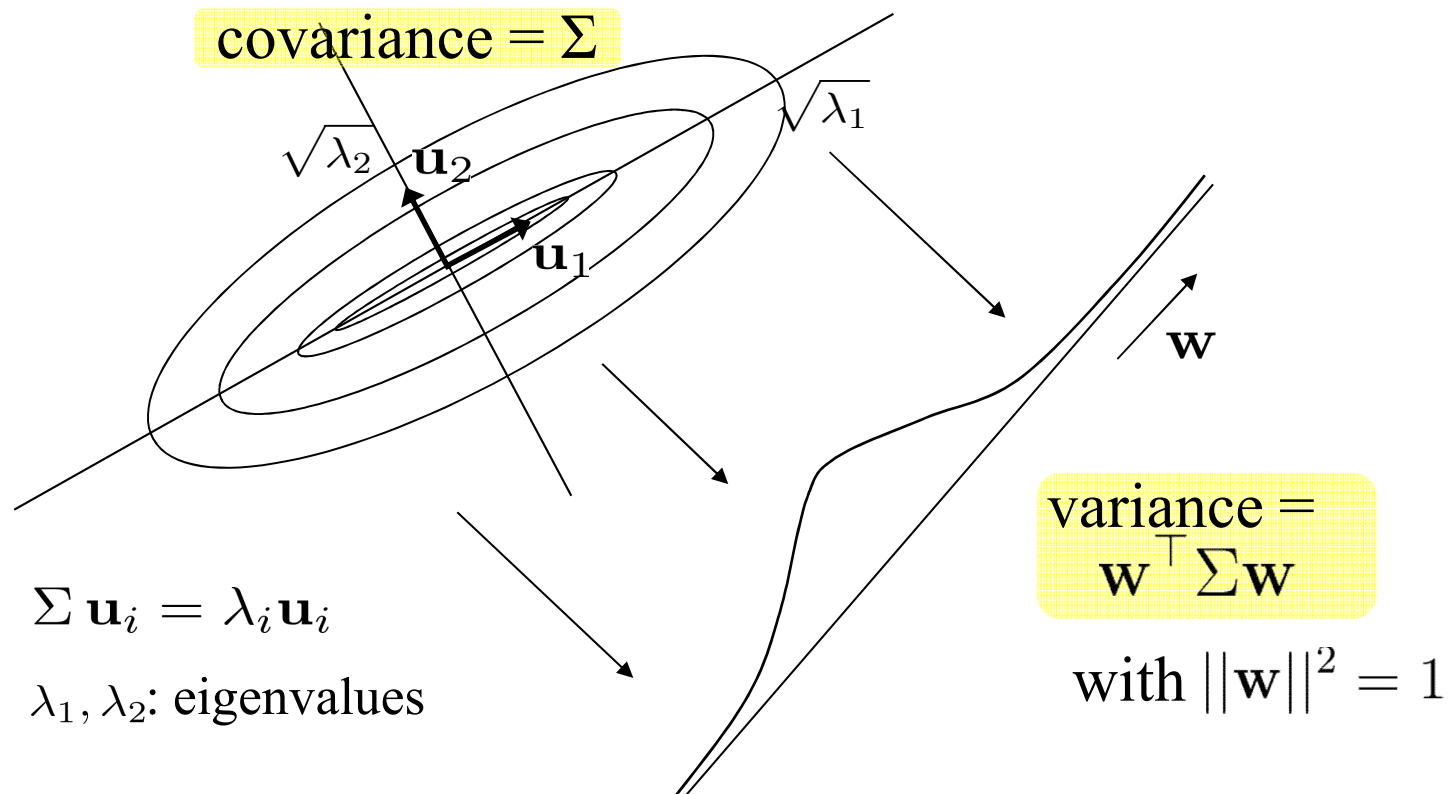
- Fisher discriminative analysis and generative methods
- Bayesian vs. Frequentist view
- Generalization issue
- Denoising autoencoder

FISHER DISCRIMINANT ANALYSIS (FDA) AND REGULARIZATION



Covariance Matrix and Projection

Covariance matrix: $\Sigma = \frac{1}{N} \sum_{i=1}^N (\mathbf{x}_i - \mu)(\mathbf{x}_i - \mu)^\top$
with $\mathbf{x}_1, \dots, \mathbf{x}_N \in \mathbb{R}^D$ and $\mu = \frac{1}{N} \sum_{i=1}^N \mathbf{x}_i$



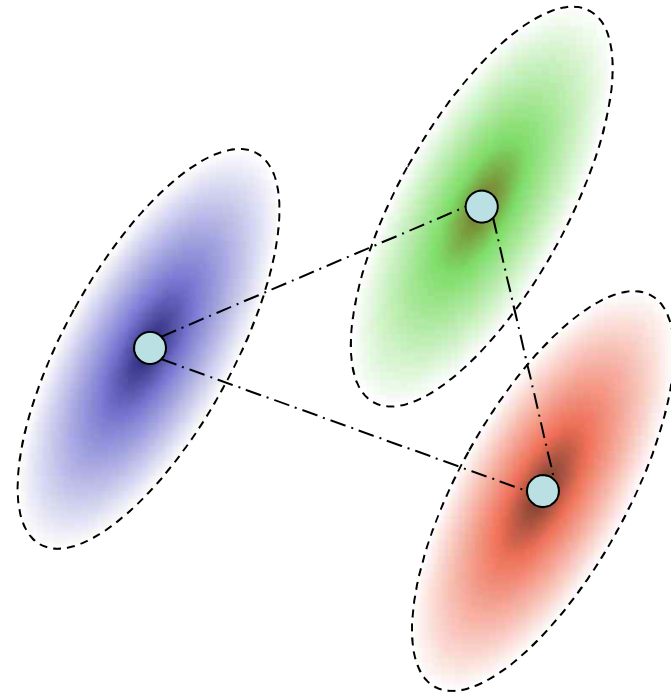
Fisher Discriminant Analysis - Consider Two Statistics

- Between-class variance

$$var_B(\mathbf{w}) = \mathbf{w}^\top S_B \mathbf{w}$$

$$S_B = \sum_{c=1}^C \frac{N_c}{N} (\mathbf{m}_c - \mathbf{m})(\mathbf{m}_c - \mathbf{m})^\top$$

$$\mathbf{m} = \sum_{i=1}^N \frac{1}{N} \mathbf{x}_i \quad \mathbf{m}_c = \frac{1}{N_c} \sum_{i \in C_c} \mathbf{x}_i$$



Fisher Discriminant Analysis - Consider Two Statistics

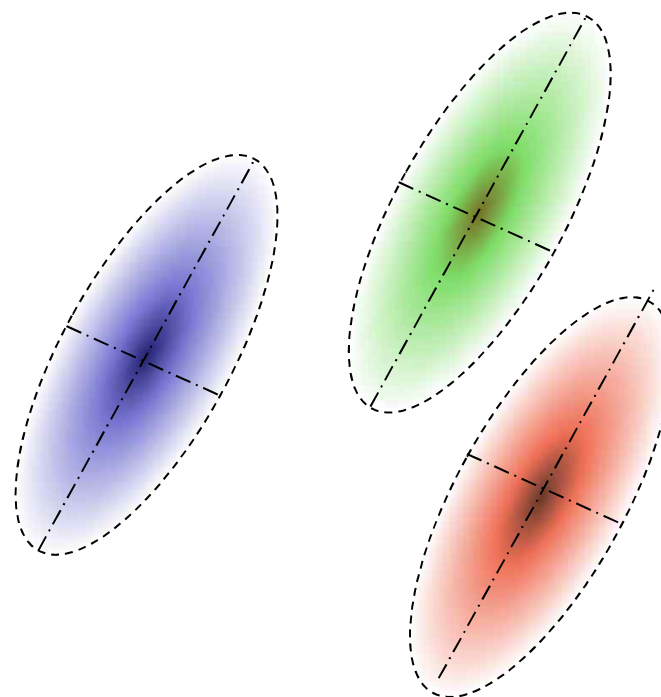
- Within-class variance

$$var_W(\mathbf{w}) = \mathbf{w}^\top S_W \mathbf{w}$$

$$S_W = \sum_{c=1}^C \frac{N_c}{N} S_c$$

S_c : covariance matrix of each class

$$S_c = \frac{1}{N_c} \sum_{i \in C_c} (\mathbf{x}_i - \mathbf{m}_c)(\mathbf{x}_i - \mathbf{m}_c)^\top \quad \mathbf{m}_c = \frac{1}{N_c} \sum_{i \in C_c} \mathbf{x}_i$$



Total Variance

- Total covariance matrix:

$$S_T = \frac{1}{N} \sum_{i=1}^N (\mathbf{x}_i - \mathbf{m})(\mathbf{x}_i - \mathbf{m})^\top$$

$$S_T = S_W + S_B$$

- Total variance along $\mathbf{w}$:

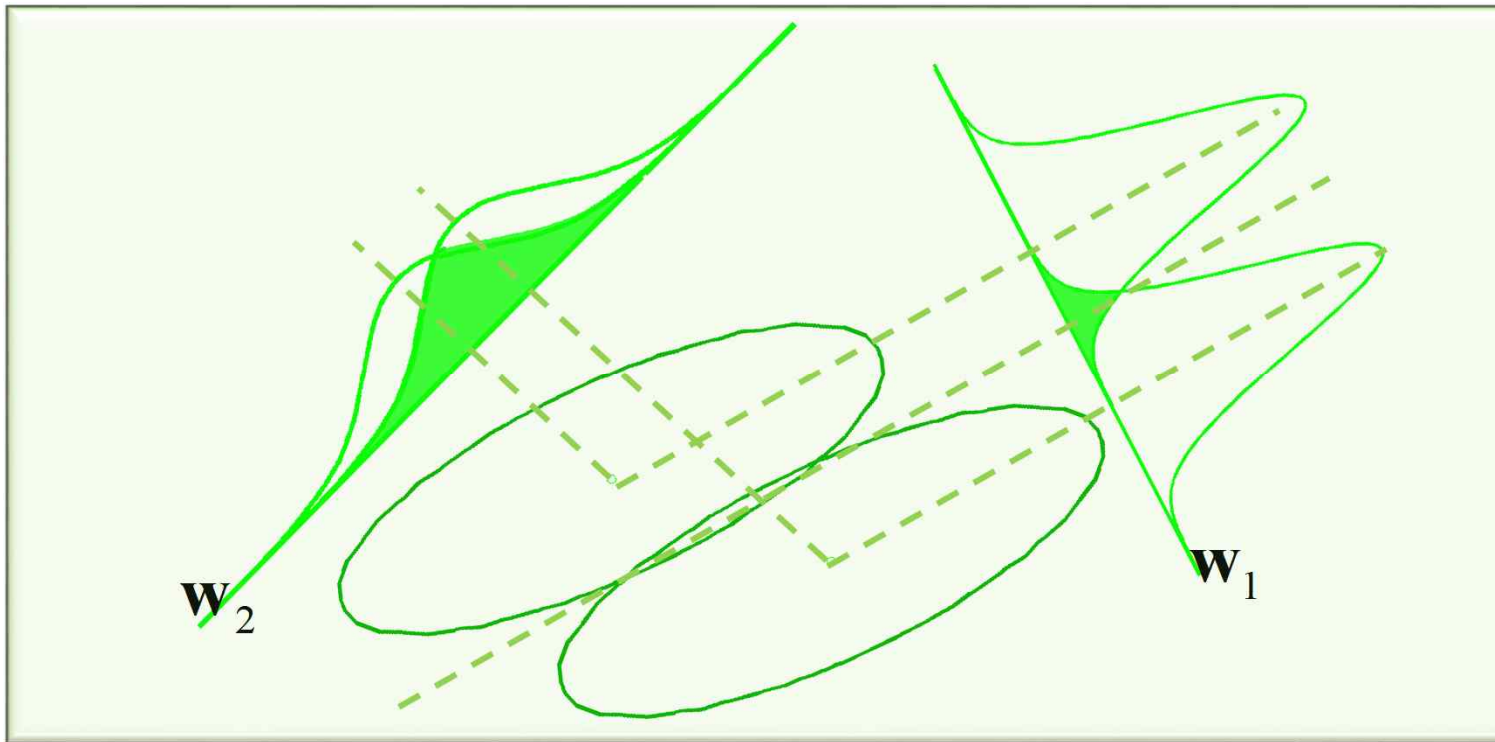
$$\text{var}(\mathbf{w}) = \text{var}_W(\mathbf{w}) + \text{var}_B(\mathbf{w})$$

Fisher Discriminant Analysis (FDA)

- Find $\mathbf{w}$ having maximal $var_B(\mathbf{w})$ for give $var_W(\mathbf{w})$.

$$\arg \max_{\mathbf{w}} J(\mathbf{w}) = \frac{\mathbf{w}^\top S_B \mathbf{w}}{\mathbf{w}^\top S_W \mathbf{w}}$$

← Between class variance
← Within class variance



Take the Derivative

$$\mathbf{w} = \arg \max_{\mathbf{w}} \frac{\text{var}_B(\mathbf{w})}{\text{var}_W(\mathbf{w})} = \arg \max_{\mathbf{w}} \frac{\mathbf{w}^\top S_B \mathbf{w}}{\mathbf{w}^\top S_W \mathbf{w}}$$

Take the derivative

$$J(\mathbf{w}) = \frac{\mathbf{w}^\top S_B \mathbf{w}}{\mathbf{w}^\top S_W \mathbf{w}}$$

$$J'(\mathbf{w}) = \frac{1}{(\mathbf{w}^\top S_W \mathbf{w})^2} [2S_B \mathbf{w}(\mathbf{w}^\top S_W \mathbf{w}) - 2\mathbf{w}^\top S_W (\mathbf{w} S_B \mathbf{w})] = 0$$

$$S_B \mathbf{w} = \left(\frac{\mathbf{w}^\top S_B \mathbf{w}}{\mathbf{w}^\top S_W \mathbf{w}} \right) S_W \mathbf{w} = \lambda S_W \mathbf{w}$$

➔ Generalized eigenvector problem

Quiz 1: Closed Form Solution

- Problem: $S_B \mathbf{w} = \lambda S_W \mathbf{w}$
- For two class problem with $N_1 = N_2$:

$$\begin{aligned} S_B &= \sum_{c=1}^2 \frac{N_c}{N} (\mathbf{m}_c - \mathbf{m})(\mathbf{m}_c - \mathbf{m})^\top \\ &= \frac{1}{2} (\mathbf{m}_1 - \mathbf{m}_2)(\mathbf{m}_1 - \mathbf{m}_2)^\top \end{aligned}$$

Assume S_W is a full-rank matrix.

- Find the closed form solution:

$$\mathbf{w} = ?$$

Quiz 2: How to solve the generalized eigenvector problem?

- Problem: $S_B \mathbf{w} = \lambda S_W \mathbf{w}$

Are the solution eigenvalues real?
(non-complex)?

If not, how are these eigenvalues are treated?

Quiz 3: Two different FDAs

- Some papers consider the criterion

$$J_T(\mathbf{w}) = \frac{\mathbf{w}^\top S_B \mathbf{w}}{\mathbf{w}^\top S_T \mathbf{w}}$$

instead of $J(\mathbf{w}) = \frac{\mathbf{w}^\top S_B \mathbf{w}}{\mathbf{w}^\top S_W \mathbf{w}}$

What is the difference?

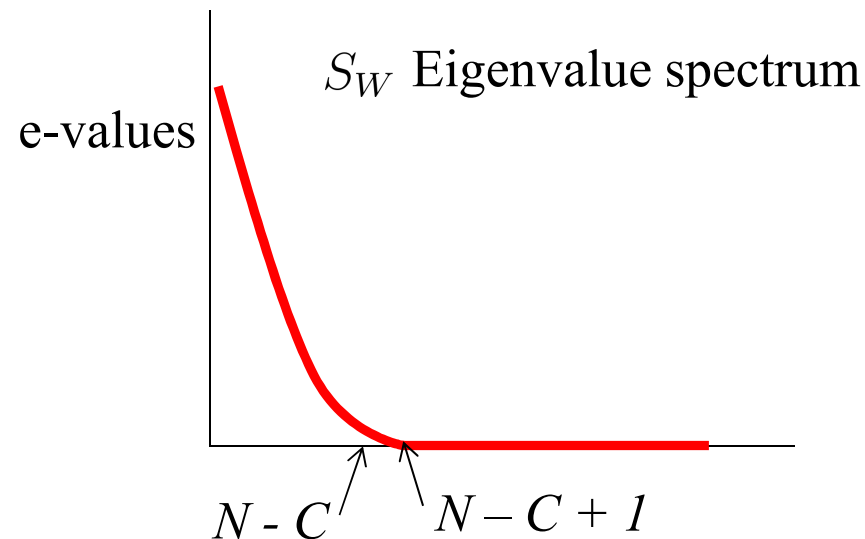
In High Dimensional Space (1/2)

$$D > N$$

$$\text{rank}(S_T = S_W + S_B) = N - 1$$

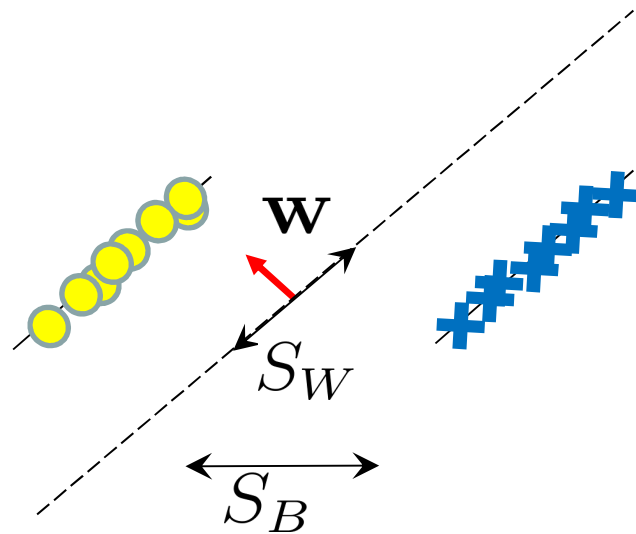
$$\text{rank}(S_W) = N - C \longrightarrow S_W: \text{not a full rank matrix}$$

$$\text{rank}(S_B) = C - 1$$



In High Dimensional Space (2/2)

- FDA will trivially pick up the null space of S_W as solution.



More seriously, the nullspace varies by sampling

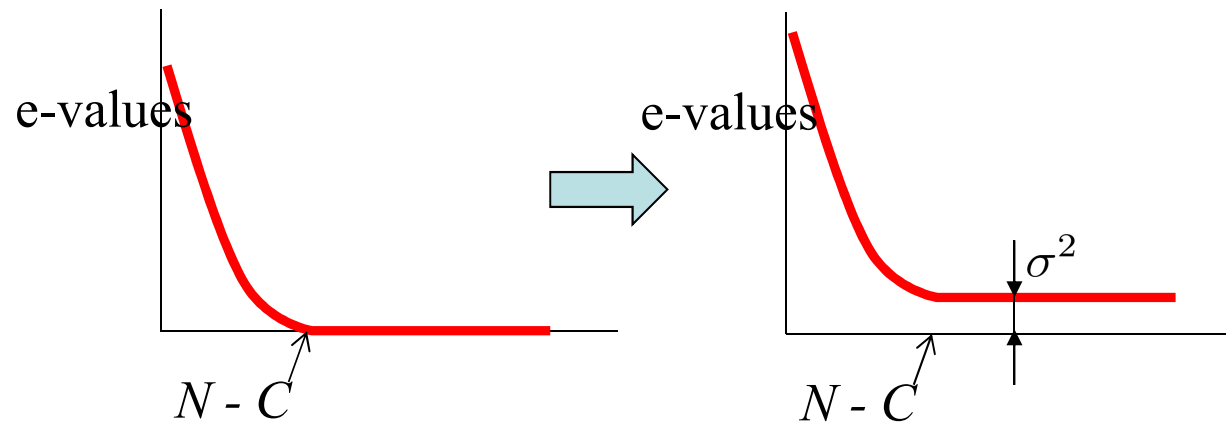
➔ Ill-posed problem

$$J(\mathbf{w}) = \infty$$

Regularization in FDA

- Regularize

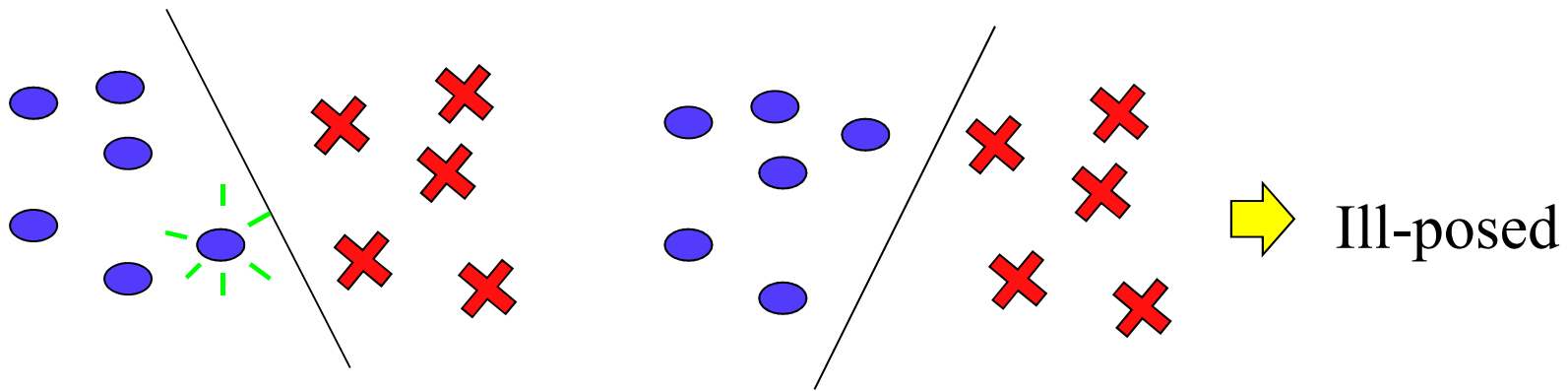
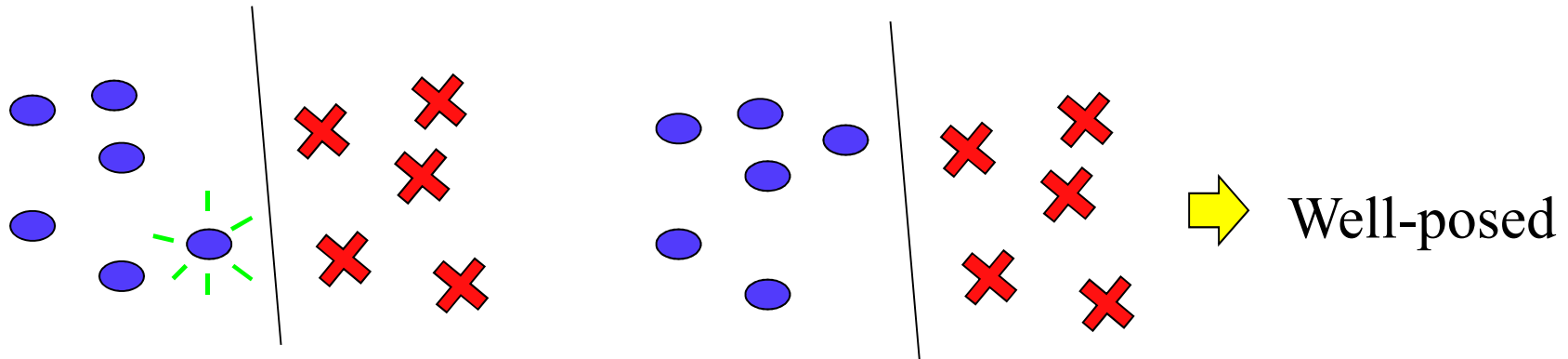
$$S_W \rightarrow S_W + \sigma^2 I$$



The problem becomes well-posed.

New solution: $\mathbf{w} = (S_W + \sigma^2 I)^{-1}(\mathbf{m}_1 - \mathbf{m}_2)$
for two classes

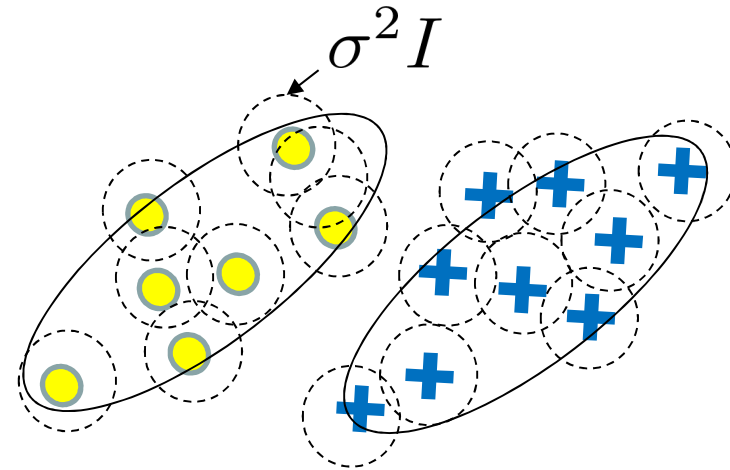
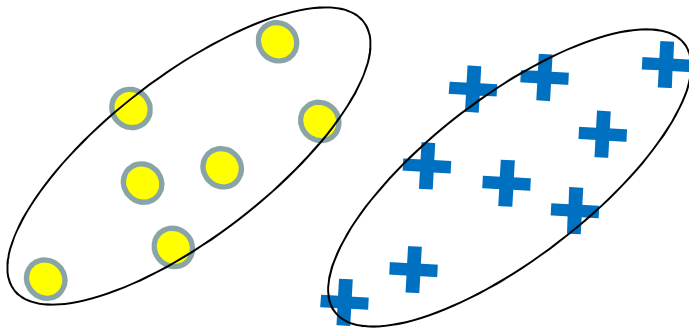
Well-posed Problem vs. Ill-Posed Problem



The variation of configuration may come from the sampling variation.

Quiz 4: New S_W with Infinite Data

- If infinite number of data are generated around each datum with isotropic Gaussian



$$S_W = \frac{1}{N} \sum_{c=1}^C \sum_{i \in C_c} (\mathbf{x}_i - \mathbf{m}_c)(\mathbf{x}_i - \mathbf{m}_c)^\top$$

$$S_W \rightarrow S_W + \sigma^2 I$$

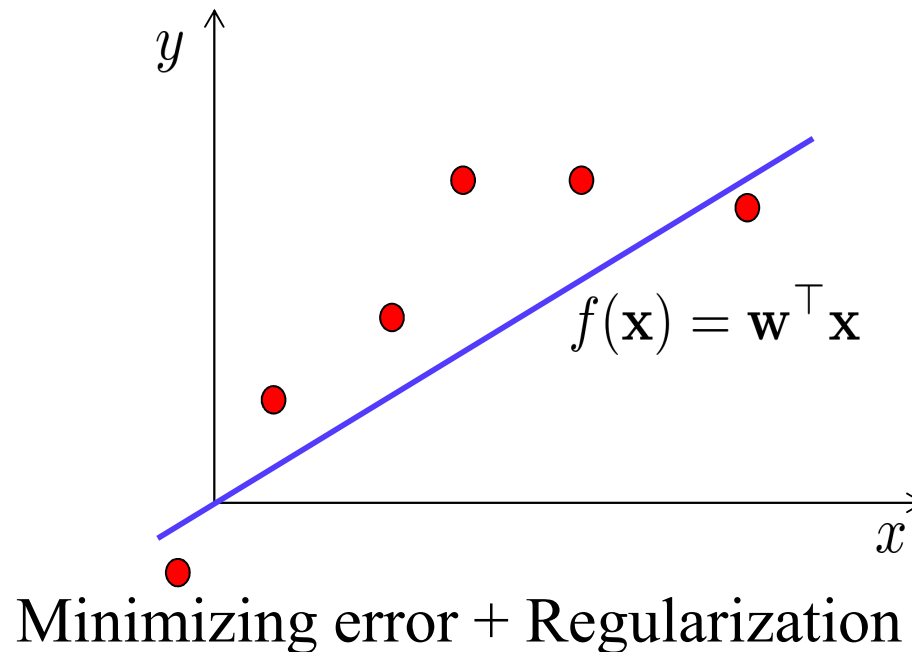
FREQUENTIST VIEW VS. BAYESIAN VIEW



Ridge Regression

- Find $f(\mathbf{x}) = \mathbf{w}^\top \mathbf{x}$, which minimizes

$$J(\mathbf{w}) = \frac{1}{2} \sum_n ||y_n - \mathbf{w}^\top \mathbf{x}_n||^2 + \frac{1}{2} \lambda \mathbf{w}^\top \mathbf{w}$$
$$\left(= ||\mathbf{y} - \mathbf{w}^\top X||^2 + \frac{1}{2} \lambda ||\mathbf{w}||^2 \right) \quad \begin{aligned} X &= [\mathbf{x}_1, \dots, \mathbf{x}_N], \quad \mathbf{x}_i \in \mathbb{R}^D \\ \mathbf{y} &= [y_1, \dots, y_N]^\top \end{aligned}$$



Ridge Regression

- Closed form solution
 - Without regularization

$$J(\mathbf{w}) = \frac{1}{2} \sum_n ||y_n - \mathbf{w}^\top \mathbf{x}_n||^2$$

$$\mathbf{w} = (X^T X)^{-1} X^\top \mathbf{y}$$

- With regularization

$$J(\mathbf{w}) = \frac{1}{2} \sum_n ||y_n - \mathbf{w}^\top \mathbf{x}_n||^2 + \frac{1}{2} \lambda \mathbf{w}^\top \mathbf{w}$$

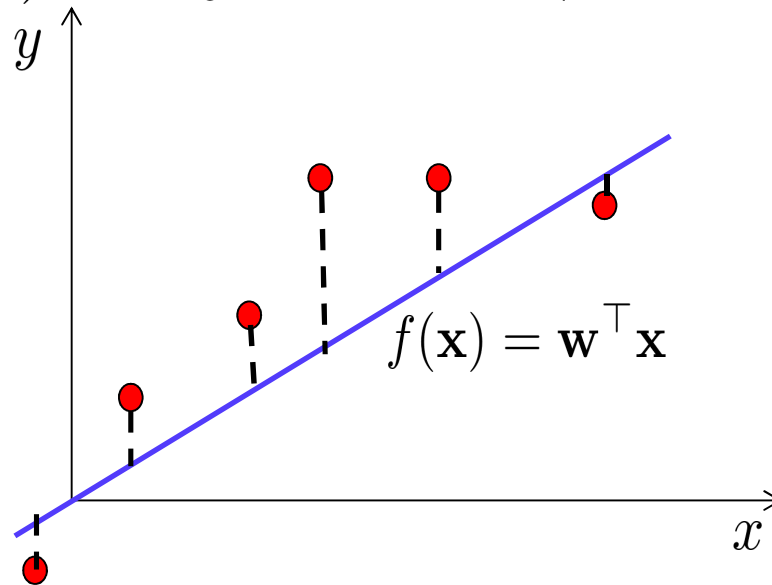
$$\mathbf{w} = (X^\top X + \lambda I)^{-1} X^\top \mathbf{y}$$

1) Frequentists' Point Estimation View

- Minimizing empirical L2 error + penalty

$$L_2 = \sum_n ||y_n - \mathbf{w}^\top \mathbf{x}||^2 \quad R = \frac{1}{2} \lambda \mathbf{w}^\top \mathbf{w}$$

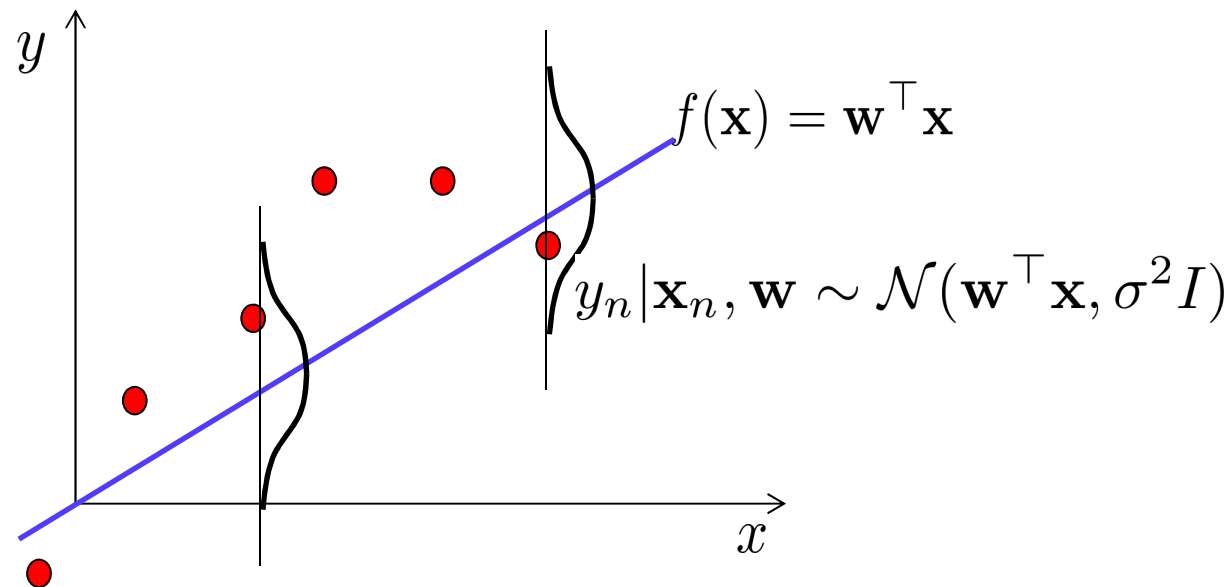
$$\mathbf{w} = (X^\top X)^{-1} X^\top \mathbf{y} \rightarrow \mathbf{w} = (X^\top X + \lambda I)^{-1} X^\top \mathbf{y}$$



2) Bayesian Inference View (1/3)

- Make a probability model

$$y_n = f(\mathbf{x}) = \mathbf{w}^\top \mathbf{x}_n + \epsilon_n, \quad \epsilon_n \sim \mathcal{N}(0, \sigma^2)$$



$$p(\mathbf{y} | X, \mathbf{w}) = \prod_{n=1}^N p(y_n | \mathbf{x}_n, \mathbf{w})$$

2) Bayesian Inference View (2/3)

- Maximizing log likelihood

$$\begin{aligned}\log p(\mathbf{y}|X, \mathbf{w}) &= \sum_{n=1}^N \log p(y_n|\mathbf{x}_n, \mathbf{w}) \\ &= -\frac{ND}{2} \log 2\pi\sigma^2 - \sum_{n=1}^N \frac{1}{2\sigma^2} \|y_n - \mathbf{w}^\top \mathbf{x}_n\|^2\end{aligned}$$

$$\mathbf{w}_{ML} = (X^\top X)^{-1} X^\top \mathbf{y}$$

2) Bayesian Inference View (3/3)

- Maximizing posterior

With a prior $\mathbf{w} \sim \mathcal{N}(0, \lambda^{-1}I)$

$$\log p(\mathbf{w}|X, \mathbf{y}) \propto - \sum_{n=1}^N ||y_n - \mathbf{w}^\top \mathbf{x}_n||^2 - \frac{\lambda}{2} \mathbf{w}^\top \mathbf{w}$$

$$\mathbf{w}_{MAP} = (X^\top X + \lambda I)^{-1} X^\top \mathbf{y} \quad \leftarrow \begin{array}{l} \text{Bayesian inference} \\ \text{of posterior using} \\ \text{linear model} \end{array}$$

Solving Optimization Problem

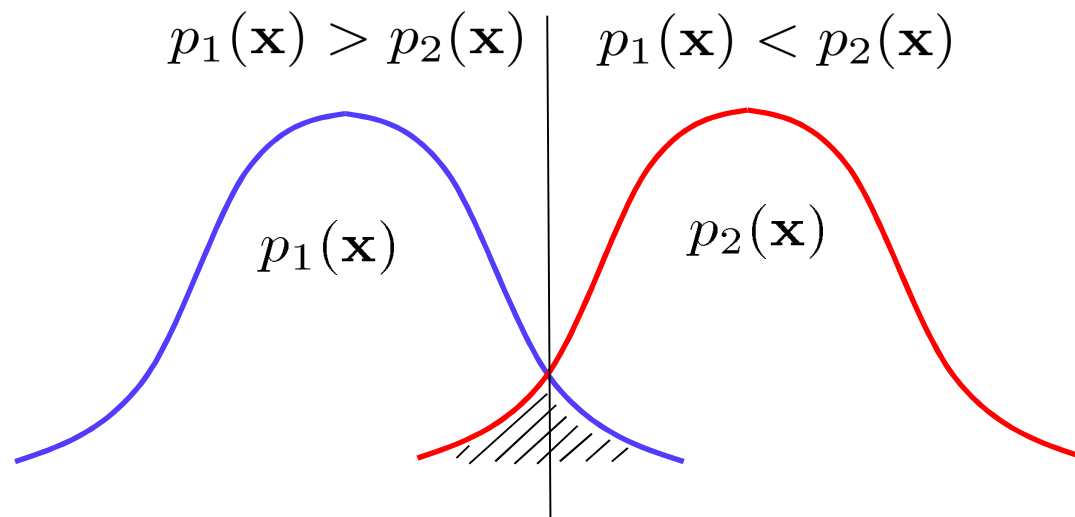
- Machine learning algorithms are optimization problems using training data
 - Minimizing empirical loss function + regularization term
 - Maximizing likelihood or posterior to estimate parameters
 - In Bayesian inference, marginalization is the main bottleneck. (e.g. through variational approximation, integration problems can be turned into optimization problems)

GENERALIZATION ISSUE



Probabilistic Assumption and Bayes Classification

- Bayes classification produces theoretical minimum error



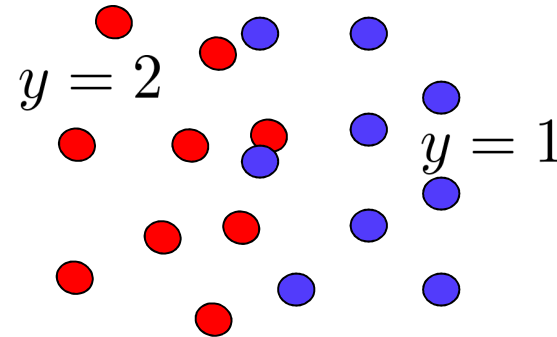
➔ Error:

$$\frac{1}{2} \int \min[p_1, p_2] d\mathbf{x}$$

Learning

- Data

$$\mathcal{D} = \{\mathbf{x}_i, y_i\}_{i=1}^N \sim P \quad (\text{Regularity})$$



- Prediction

$$\mathbf{x} \in \mathbb{R}^D \xrightarrow{y = f(\mathbf{x})} \begin{matrix} y \in \{1, 2, \dots, C\} \\ y \in \mathbb{R} \end{matrix}$$

- Learning

- Learn prediction function $f(\mathbf{x}) \in \mathcal{H}$
from data $\mathcal{D}$
($\mathcal{H}$: Hypothesis set/Candidate set)

Quantify the Evaluation

- Measure of quality: expected loss

$$L = \mathbb{E}_P[l(y, f(\mathbf{x}))] \quad l(y, y'): \text{loss function}$$

- Estimated error

$$\hat{L} = \sum_n l(y_n, f(\mathbf{x}_n)), \quad f(\mathbf{x}) \in \mathcal{H}$$

- Examples

- Classification

$$\hat{L} = \frac{1}{N} \sum_n \mathbb{I}(y_n \neq f(\mathbf{x}_n))$$

- Regression

$$\hat{L} = \frac{1}{N} \sum_n \|y_n - f(\mathbf{x}_n)\|^2$$

- Clustering

$$\hat{L} = \frac{1}{N} \sum_n \min_c \|y_n - f(\mathbf{x}_n)\|^2$$

Consistent Learner

- $\mathcal{H}$ satisfies

$$\hat{L} \xrightarrow{N \rightarrow \infty} L$$

$$P\left\{\sup_{f \in \mathcal{H}} (L(f) - \hat{L}(f)) > \epsilon\right\} \rightarrow 0 \quad \text{for } \epsilon > 0$$

<Uniform convergence>

- Caution:
 - The definition of consistency is *not*

$$\hat{L}(f) \rightarrow L(f) \quad \text{for } f \in \mathcal{H}$$

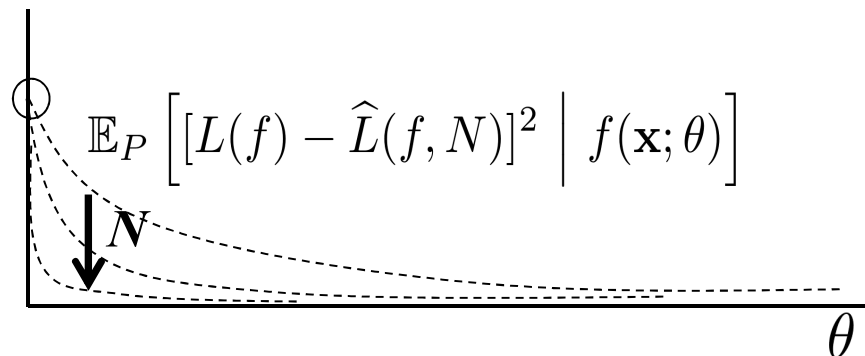
Quiz 1

- Consider a hypothesis set $\mathcal{H}$ which satisfies

$$\mathbb{E}_P \left[[L(f) - \hat{L}(f, N)]^2 \mid f(\mathbf{x}; \theta) \right] = \left(\frac{1}{N} \right)^\theta$$

$$\mathcal{H} = \{f(\mathbf{x}; \theta) \mid \theta > 0\}$$

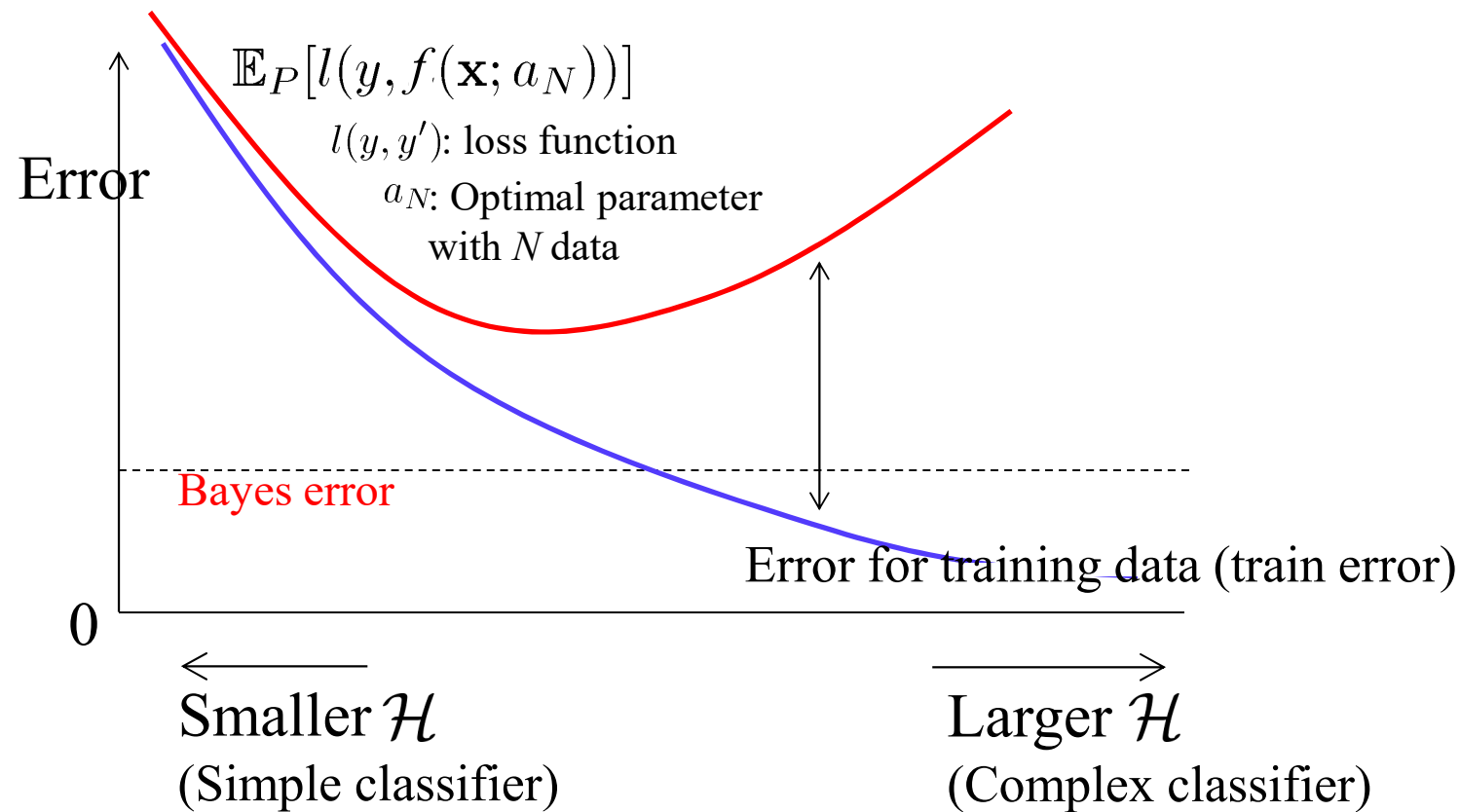
Explain that learning with $\mathcal{H}$ is not consistent though it satisfies $\hat{L}(f) \rightarrow L(f)$.



What is the possible problem in this case?

Consistency and Bayes Error

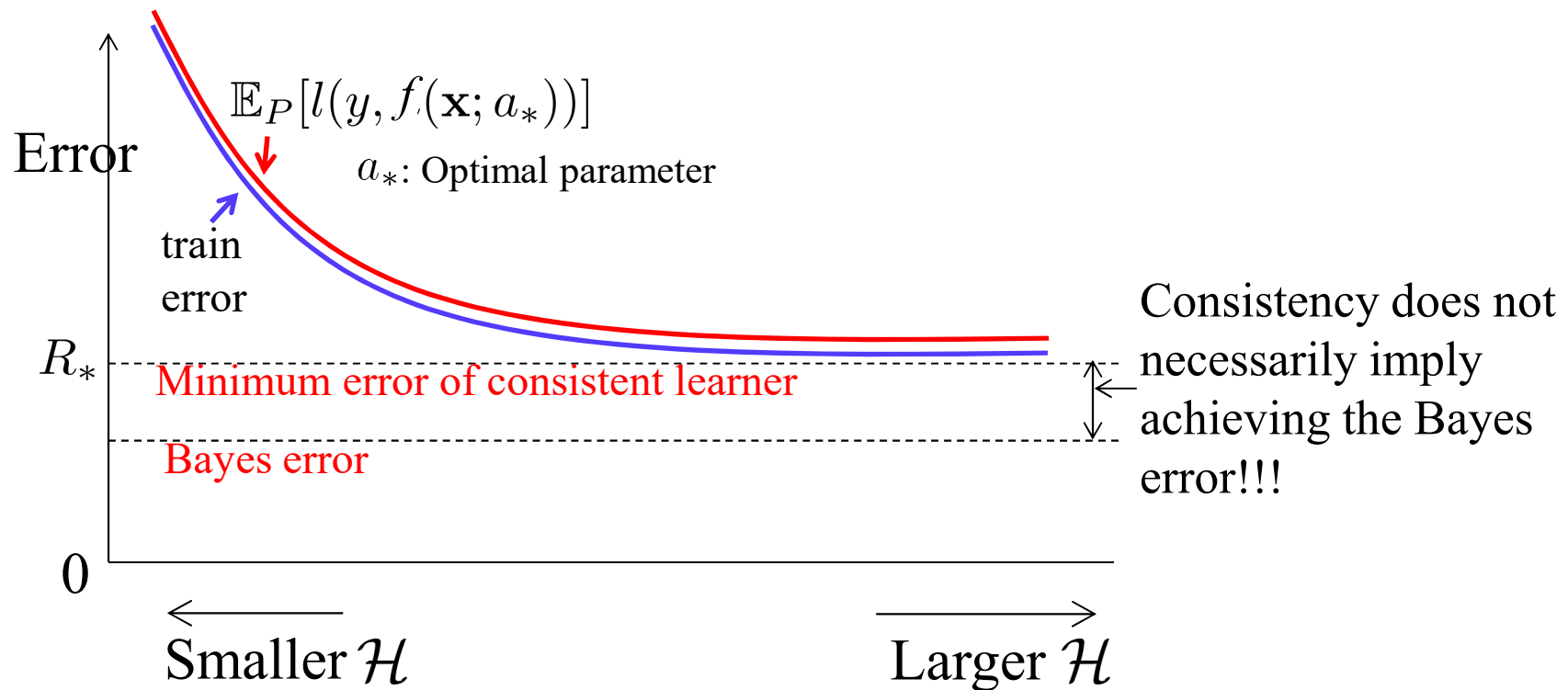
- Minimizing expected error (objective) vs. minimizing estimated error



(For example, a linear classifier with regularization)

Consistency and Bayes Error

- Consistent learner with many data



(For example, a linear classifier with regularization)

Related Terms with Confining H

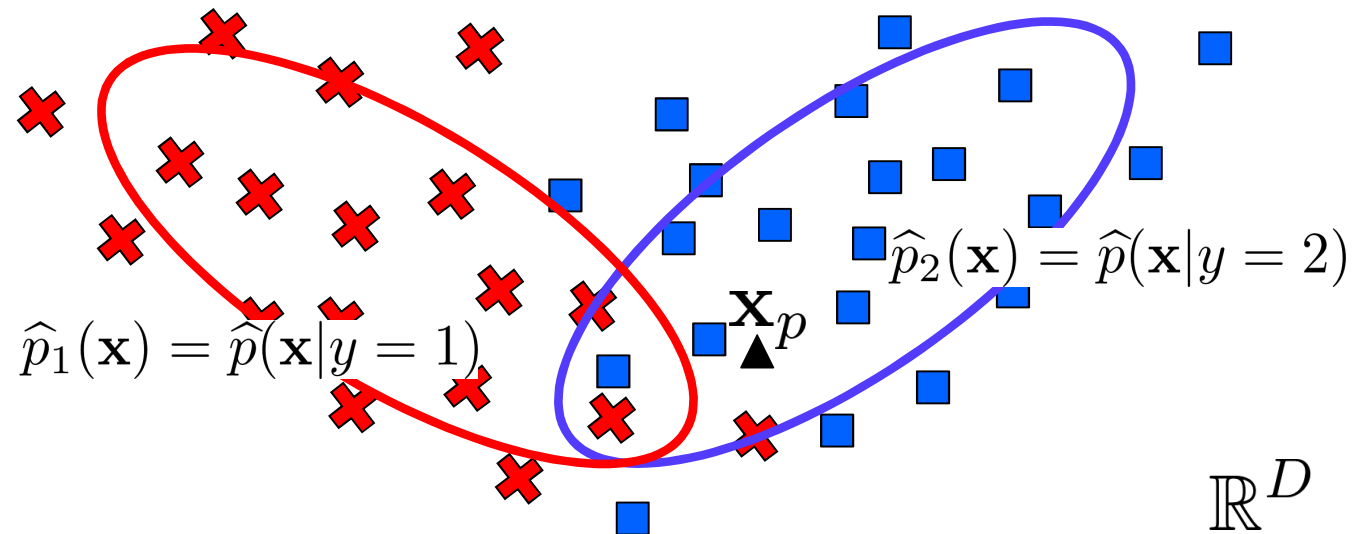
- Linear model →
VC-dim for classification = Dimensionality + 1
 - Small number of parameters
 - Large margin
 - Regularization
 - Bias-Variance trade-off
 - Generalization ability, overfitting
- ➔ Many terms are theoretically connected

GENERATIVE VS. DISCRIMINATIVE METHODS



Model + Estimated Parameters

- Ex. Gaussian model



$$\begin{aligned}\hat{p}_1(\mathbf{x}_p) &\geq \hat{p}_2(\mathbf{x}_p) \rightarrow y_p = 1 \\ \hat{p}_1(\mathbf{x}_p) &< \hat{p}_2(\mathbf{x}_p) \rightarrow y_p = 2\end{aligned}$$

Homoscedastic Two-class Gaussians

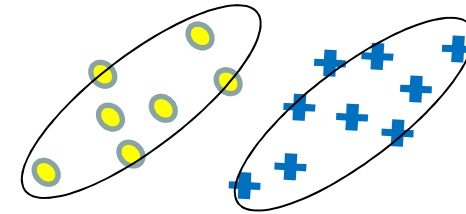
$$p_c(\mathbf{x}) = \frac{1}{\sqrt{2\pi}^D |\Sigma|^{\frac{1}{2}}} \exp \left(-\frac{1}{2} (\mathbf{x} - \mu_c)^\top \underset{\substack{\uparrow \\ \text{Common covariance matrix}}}{\Sigma^{-1}} (\mathbf{x} - \mu_c) \right) \quad c = 1, 2$$

$$\begin{aligned} \log \frac{p_2(\mathbf{x})}{p_1(\mathbf{x})} &= \frac{1}{2} (\mathbf{x} - \mu_1)^\top \Sigma^{-1} (\mathbf{x} - \mu_1) - \frac{1}{2} (\mathbf{x} - \mu_2)^\top \Sigma^{-1} (\mathbf{x} - \mu_2) \\ &= \cancel{\frac{1}{2} \mathbf{x}^\top \Sigma^{-1} \mathbf{x}} - \cancel{\frac{1}{2} \mathbf{x}^\top \Sigma^{-1} \mathbf{x}} - \mu_1^\top \Sigma^{-1} \mathbf{x} + \mu_2^\top \Sigma^{-1} \mathbf{x} + \frac{1}{2} (\mu_1^\top \Sigma^{-1} \mu_1 - \mu_2^\top \Sigma^{-1} \mu_2) \\ &= -\mu_1^\top \Sigma^{-1} \mathbf{x} + \mu_2^\top \Sigma^{-1} \mathbf{x} + \frac{1}{2} (\mu_1^\top \Sigma^{-1} \mu_1 - \mu_2^\top \Sigma^{-1} \mu_2) \underline{= 0} \quad (\text{boundary}) \end{aligned}$$

$$\boxed{\begin{aligned} \underline{(\mu_1 - \mu_2)^\top \Sigma^{-1} \mathbf{x}} &= \frac{1}{2} (\mu_1^\top \Sigma^{-1} \mu_1 - \mu_2^\top \Sigma^{-1} \mu_2) \\ &\underline{= \mathbf{w}^\top} \end{aligned}}$$

$$\mathbf{w} = \Sigma^{-1} (\mu_1 - \mu_2) \quad \leftarrow \text{Solution of } \arg \max \frac{\mathbf{w}^\top (\mu_1 - \mu_2) (\mu_1 - \mu_2)^\top \mathbf{w}}{\mathbf{w}^\top \Sigma \mathbf{w}}$$

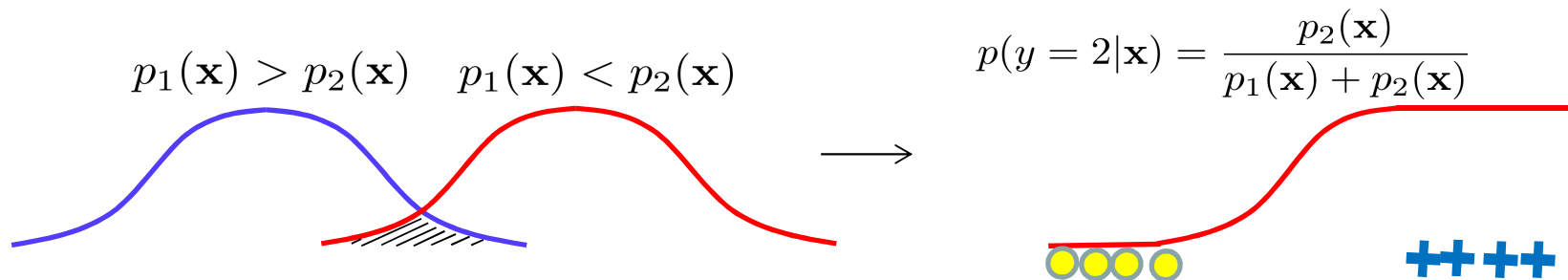
In Terms of the Posterior



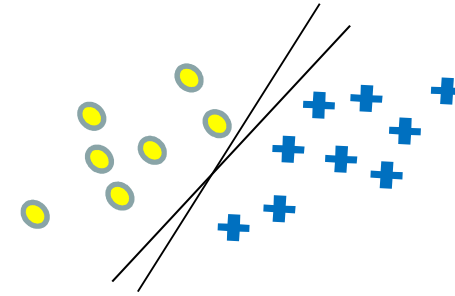
$$p(y = 1 | \mathbf{x}, \mathbf{w}, b) = \frac{p_1}{p_1 + p_2} = \frac{1}{1 + p_2/p_1} = \frac{1}{1 + \exp(\mathbf{w}^\top \mathbf{x} - b)}$$

$$p(y = 2 | \mathbf{x}, \mathbf{w}, b) = \frac{p_2}{p_1 + p_2} = 1 - p(y = 1 | \mathbf{x}) = \frac{\exp(\mathbf{w}^\top \mathbf{x} - b)}{1 + \exp(\mathbf{w}^\top \mathbf{x} - b)}$$

- Class-conditional model vs. Posterior model



Logistic Regression



- Starts from the posterior

$$p(\mathbf{y}|X, \mathbf{w}, b) = \prod_n p(y_n = 1 | \mathbf{x}_n, \mathbf{w}, b)$$

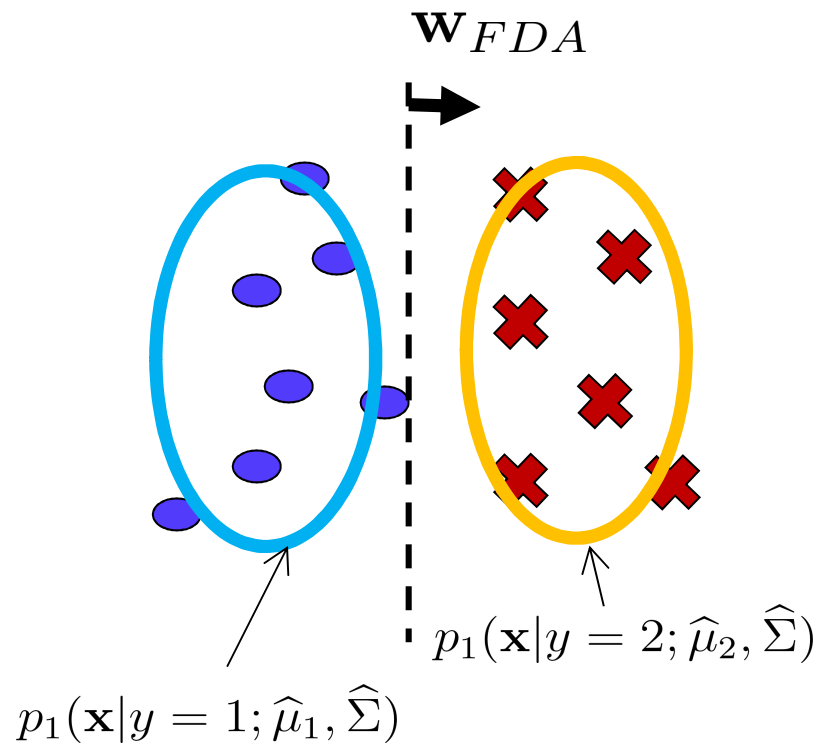
$$p(y_n | \mathbf{x}_n, \mathbf{w}, b) = \begin{cases} \frac{1}{1 + \exp(\mathbf{w}^\top \mathbf{x} - b)}, & y_n = 1 \\ \frac{\exp(\mathbf{w}^\top \mathbf{x} - b)}{1 + \exp(\mathbf{w}^\top \mathbf{x} - b)}, & y_n = 2 \end{cases}$$

$$\mathbf{w}, b = \arg \max_{\mathbf{w}, b} \underbrace{\ln p(\mathbf{y}|X, \mathbf{w}, b)}_{\sum_n \mathbb{I}(y_n = 1) \ln p(y_n = 1 | \mathbf{x}_n, \mathbf{w}, b) + \mathbb{I}(y_n = 2) \ln p(y_n = 2 | \mathbf{x}_n, \mathbf{w}, b)}$$

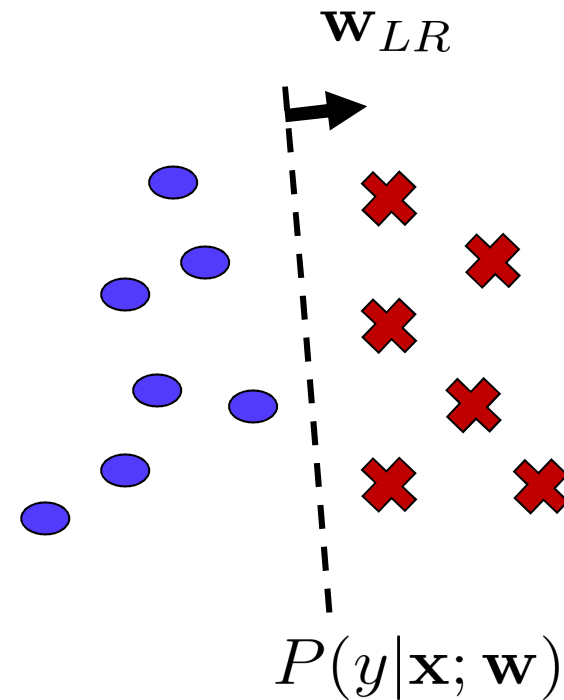
Learn $\mathbf{w}$ using posterior model (instead of class-conditional model)

FDA and Logistic Regression

- Are the results same or not?



$$\mathbf{w}_{FDA} = \hat{\Sigma}^{-1}(\hat{\mu}_1 - \hat{\mu}_2)$$



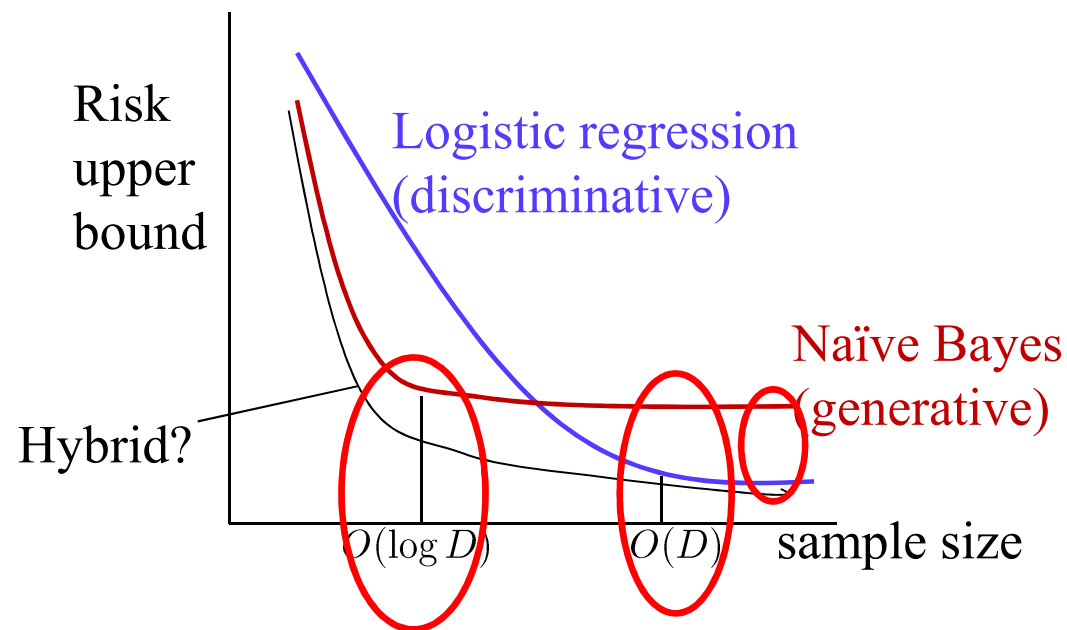
$$\mathbf{w}_{LR} = \arg \max_{\mathbf{w}} P(y|\mathbf{x}; \mathbf{w})$$

Generative vs. Discriminative Methods

- Generative methods
 - Gaussian class-conditional model
 - Graphical model
 - Restricted Boltzmann Machines
- Discriminative methods
 - Support Vector Machines (SVMs)
 - Logistic regression
 - Artificial Neural Networks

Comparative Study (1/2)

- Generative & Discriminative Pair
 - Same number of parameters, same form of $h(x)$



S. Lacoste-Julien et al. (2009) The generative and discriminative learning interface, *NIPS Workshop*
A. Y. Ng & M. I. Jordan (2001) On discriminative vs. generative classifiers: a comparison of logistic regression and naïve Bayes, *NIPS*

Comparative Study (2/2)

- Discriminative analog of naïve Bayes (or FDA) is logistic regression
- The error $err(f_{Disc}(\mathbf{x}))$ converges to $err(f_{Disc,\infty}(\mathbf{x}))$, and $err(f_{Disc,\infty}(\mathbf{x}))$ is no worse than linear classifier picked by naïve Bayes (or FDA).
- With $O(\log N)$, the parameters of f_{Gen} are uniformly close to $f_{Gen,\infty}$ uniformly.
- The parameter convergence implies $err(f_{Gen}(\mathbf{x}))$ approaches $err(f_{Gen,\infty}(\mathbf{x}))$.

Science of Learning Algorithms for Prediction

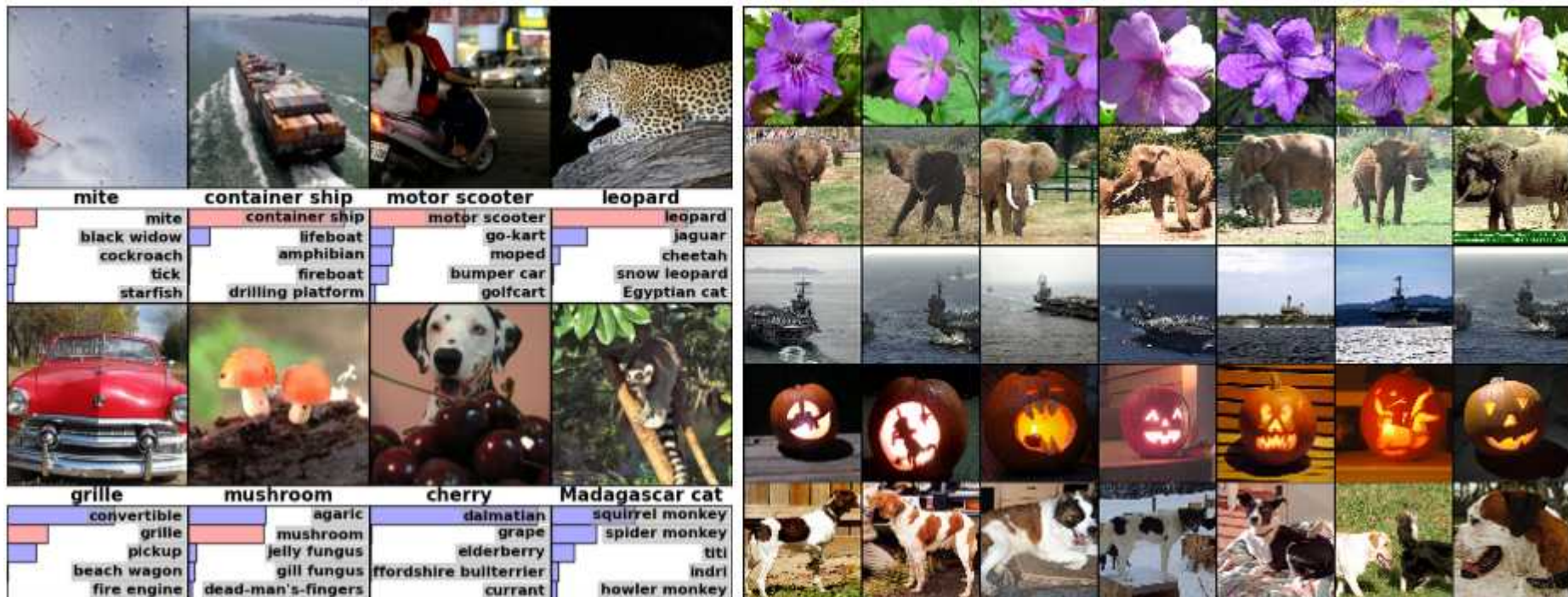
- Disentangling the various approaches to supervised learning: answer questions like
 - What are the relative strengths of the different approaches to supervised learning?
 - What are the relative weaknesses?
 - How do the questions they are addressing differ?
 - How do their assumptions differ?
 - How much do they duplicate one another?
 - What fundamentally distinct ideas and insights do they collectively embody?
 - What can be gained by combining techniques from the different approaches?

[David H. Wolpert, 1994]



AlexNet (2012 NIPS)

- ImageNet Large Scale Visual Recognition Challenge 2010 (ILSVRC2010)



ImageNet Classification with Deep Convolutional Neural Networks

Alex Krizhevsky
University of Toronto
kriz@cs.utoronto.ca

Ilya Sutskever
University of Toronto
ilya@cs.utoronto.ca

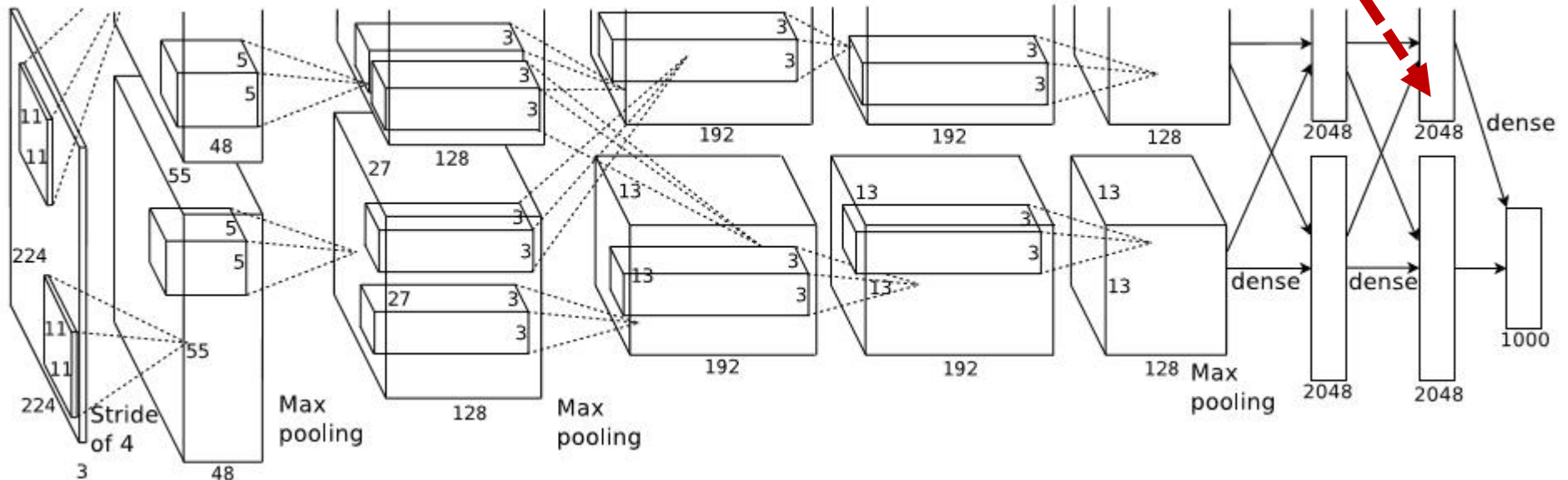
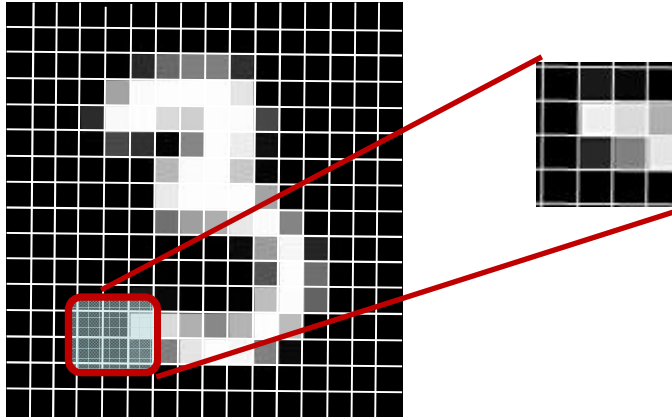
Geoffrey E. Hinton
University of Toronto
hinton@cs.utoronto.ca



Seoul National University



Architecture of AlexNet



NIPS 2013 Tutorial - Deep Learning for Computer Vision (Rob Fergus)

You Tube KR

검색



Tapping off Features at each Layer
Plug features from each layer into linear SVM or soft-max

	Cal-101 (30/class)	Cal-256 (60/class)
SVM (1)	44.8 $\pm$ 0.7	24.6 $\pm$ 0.4
SVM (2)	66.2 $\pm$ 0.5	39.6 $\pm$ 0.3
SVM (3)	72.3 $\pm$ 0.4	46.0 $\pm$ 0.3
SVM (4)	76.6 $\pm$ 0.4	51.3 $\pm$ 0.1
SVM (5)	86.2 $\pm$ 0.8	65.6 $\pm$ 0.3
SVM (7)	85.5 $\pm$ 0.4	71.7 $\pm$ 0.2
Softmax (5)	82.9 $\pm$ 0.4	65.7 $\pm$ 0.5
Softmax (7)	85.4 $\pm$ 0.4	72.6 $\pm$ 0.1

37:13 / 1:58:01

NIPS 2013 Tutorial - Deep Learning for Computer Vision (Rob Fergus)



NIPS

구독중



1,650

조회수 13,530회



Seoul National University

Matlab Computer Vision System Toolbox

- Image category classification using deep learning

Documentation

Search R2016b Documentation

Documentation

CONTENTS

달기

< Examples Home

< Computer Vision System Toolbox

< MATLAB Examples

< Object Detection and Recognition

Image Category Classification Using Deep Learning

ON THIS PAGE

Overview

Check System Requirements

Download Image Data

Load Images

Download Pre-trained Convolutional Neural Network (CNN)

Load Pre-trained CNN

Pre-process Images For CNN

Prepare Training and Test Image Sets

Extract Training Features Using CNN

Train A Multiclass SVM Classifier Using CNN Features

Evaluate Classifier

Try the Newly Trained Classifier on Test Images

References

ans =

23x1 Layer array with layers:

1	'input'	Image Input	227x227x3 images with 'zerocenter' normalization
2	'conv1'	Convolution	96 11x11x3 convolutions with stride [4 4] and padding [0 0]
3	'relu1'	ReLU	ReLU
4	'norm1'	Cross Channel Normalization	cross channel normalization with 5 channels per element
5	'pool1'	Max Pooling	3x3 max pooling with stride [2 2] and padding [0 0]
6	'conv2'	Convolution	256 5x5x48 convolutions with stride [1 1] and padding [2 2]
7	'relu2'	ReLU	ReLU
8	'norm2'	Cross Channel Normalization	cross channel normalization with 5 channels per element
9	'pool2'	Max Pooling	3x3 max pooling with stride [2 2] and padding [0 0]
10	'conv3'	Convolution	384 3x3x256 convolutions with stride [1 1] and padding [1 1]
11	'relu3'	ReLU	ReLU
12	'conv4'	Convolution	384 3x3x192 convolutions with stride [1 1] and padding [1 1]
13	'relu4'	ReLU	ReLU
14	'conv5'	Convolution	256 3x3x192 convolutions with stride [1 1] and padding [1 1]
15	'relu5'	ReLU	ReLU
16	'pool5'	Max Pooling	3x3 max pooling with stride [2 2] and padding [0 0]
17	'fc6'	Fully Connected	4096 fully connected layer
18	'relu6'	ReLU	ReLU
19	'fc7'	Fully Connected	4096 fully connected layer
20	'relu7'	ReLU	ReLU
21	'fc8'	Fully Connected	1000 fully connected layer
22	'prob'	Softmax	softmax
23	'classificationLayer'	Classification Output	cross-entropy with 'n01440764', 'n01443537', and 998 other classes

The first layer defines the input dimensions. Each CNN has a different input size requirements. The one used in this example requires image input that is 227-by-227-by-3.

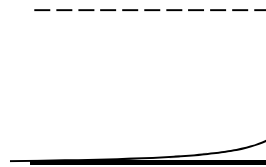
<http://kr.mathworks.com/help/vision/examples/image-category-classification-using-deep-learning.html>



ReLU Activation Function

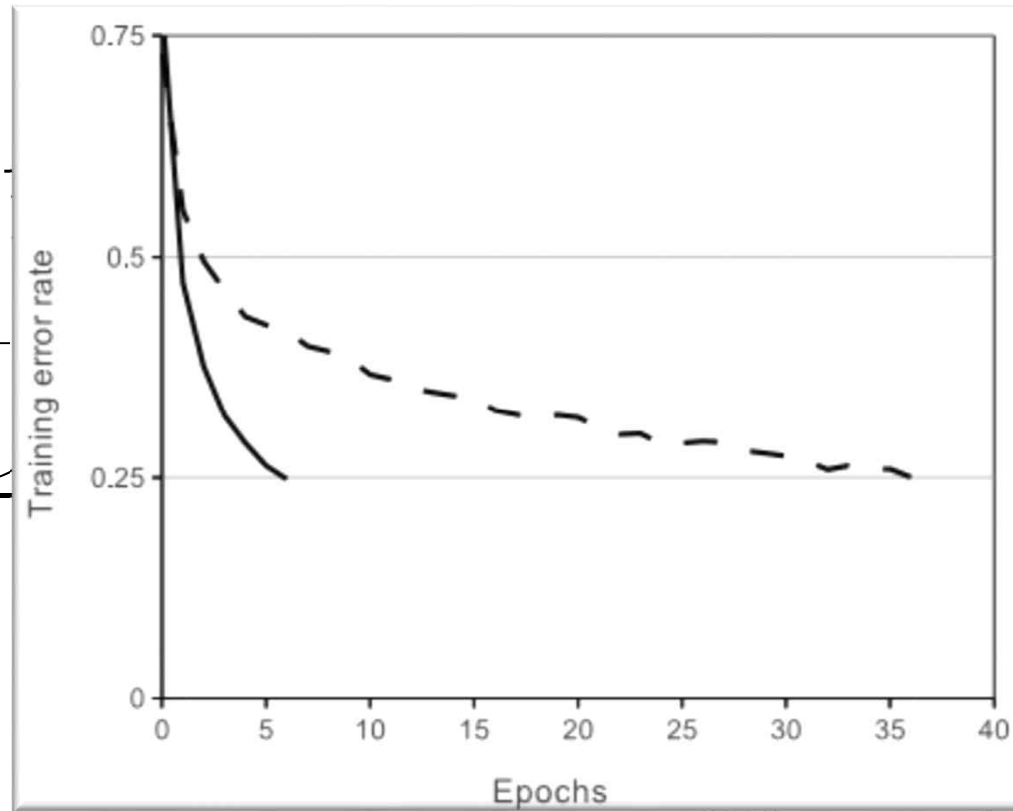
- Sigmoid

$$\sigma(a) =$$

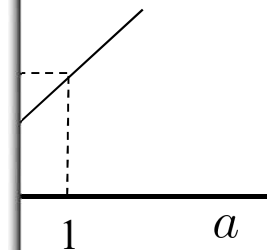


$$\frac{d\sigma(a)}{da}$$

- Rectified Linear Unit

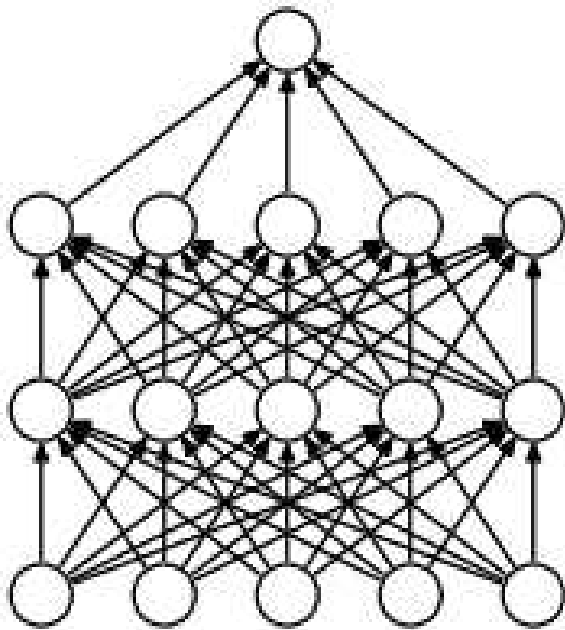


$$\max\{0, a\}$$

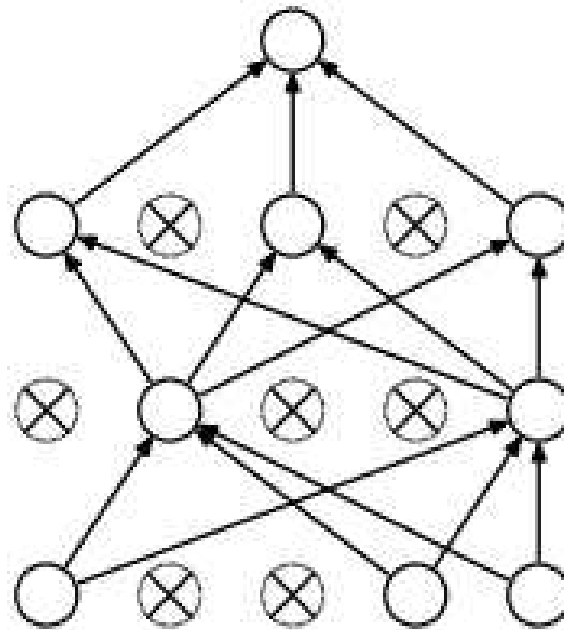


$$\begin{cases} 1 & a > 0 \\ 0 & a \leq 0 \end{cases}$$

Dropout



(a) Standard Neural Net



(b) After applying dropout.

Srivastava et al. (2014) Dropout: A Simple Way to Prevent Neural Networks from Overfitting, *JMLR*

GPGPU



- GTX 580, 3G memory

Deep Autoencoder Using RBM as Pretraining Procedure

- Science, Vol. 313 no. 5786 pp. 504-507
 - We introduce this pretraining procedure for binary data, generalize it to real-valued data, and show that it works well for a variety of data sets

the optical properties of the SRR constituent materials are identical for all configurations. The blue bars in Fig. 1 summarize the measured SHG signals. For excitation of the *LC* resonance in Fig. 1A (horizontal incident polarization), we find an SHG signal that is 500 times above the noise level. As expected for SHG, this signal closely scales with the square of the incident power (Fig. 2A). The polarization of the SHG emission is nearly vertical (Fig. 2B). The small angle with respect to the vertical is due to deviations from perfect mirror symmetry of the SRRs (see electron micrographs in Fig. 1). Small detuning of the *LC* resonance toward smaller wavelength (i.e., to 1.3- μ m wavelength) reduces the SHG signal strength from 100% to 20%. For excitation of the Mie resonance with vertical incident polarization in Fig. 1D, we find a small signal just above the noise level. For excitation of the Mie resonance with horizontal incident polarization in Fig. 1C, a small but significant SHG emission is found, which is again po-

of the nonlinear optical properties of metallic [10.1126/science.1129198](https://doi.org/10.1126/science.1129198)

Reducing the Dimensionality of Data with Neural Networks

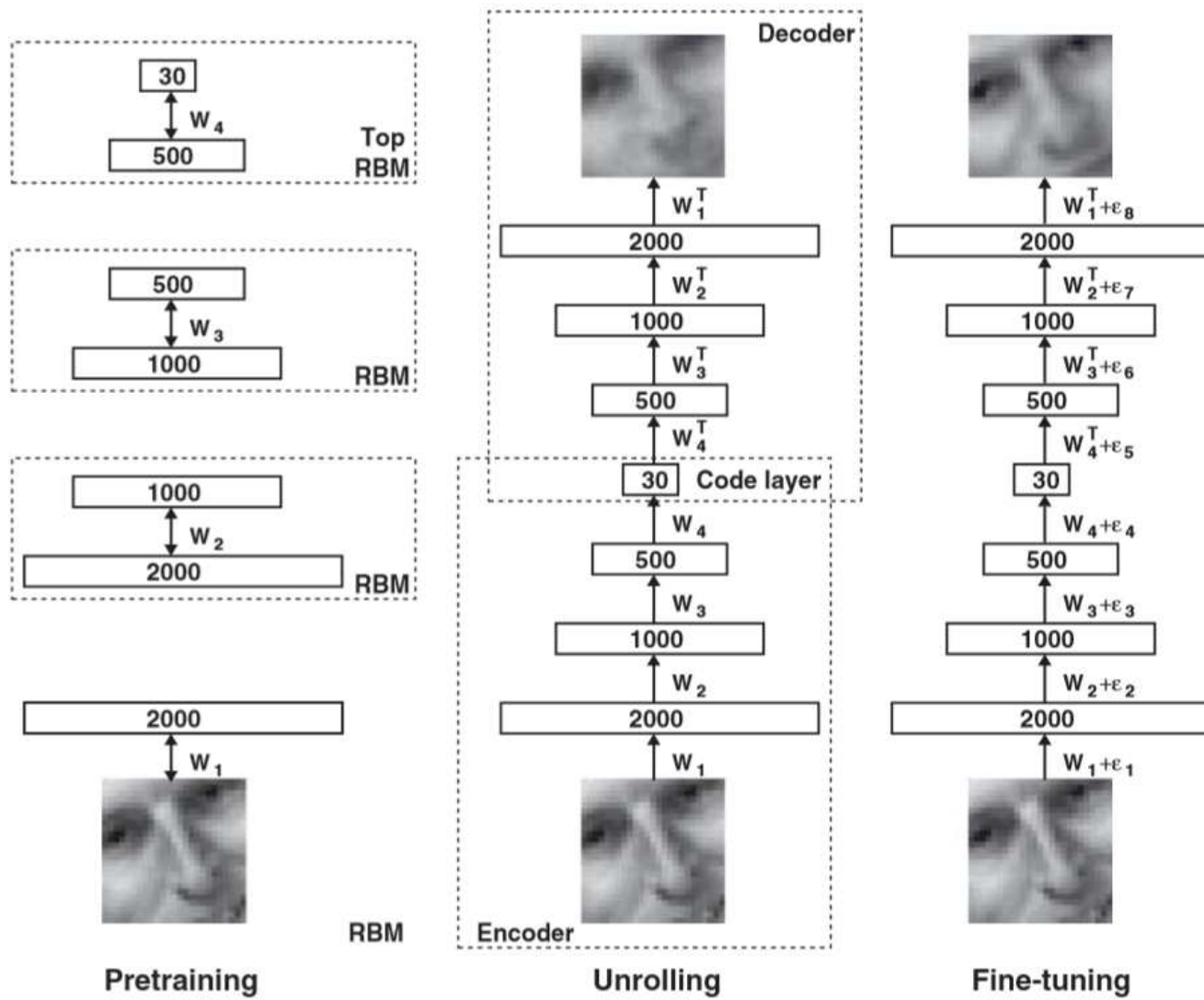
G. E. Hinton* and R. R. Salakhutdinov

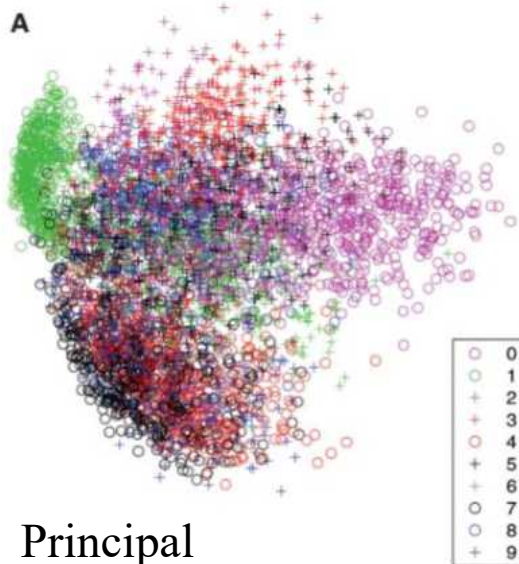
High-dimensional data can be converted to low-dimensional codes by training a multilayer neural network with a small central layer to reconstruct high-dimensional input vectors. Gradient descent can be used for fine-tuning the weights in such “autoencoder” networks, but this works well only if the initial weights are close to a good solution. We describe an effective way of initializing the weights that allows deep autoencoder networks to learn low-dimensional codes that work much better than principal components analysis as a tool to reduce the dimensionality of data.

Dimensionality reduction facilitates the classification, visualization, communication, and storage of high-dimensional data. A simple and widely used method is principal components analysis (PCA), which

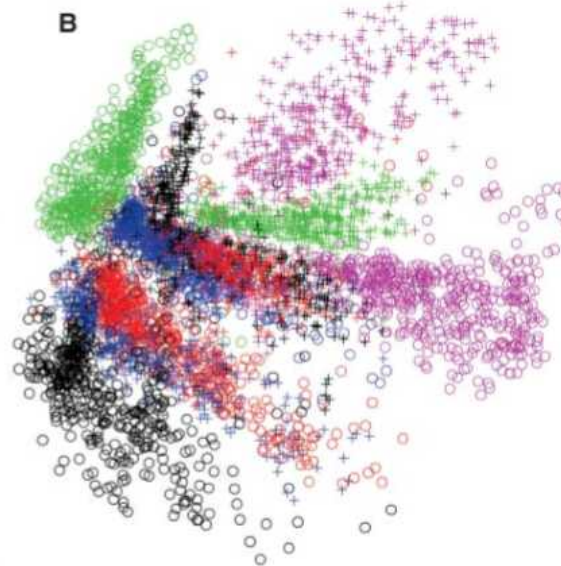
finds the directions of greatest variance in the data set and represents each data point by its coordinates along each of these directions. We describe a nonlinear generalization of PCA that uses an adaptive, multilayer “encoder” network



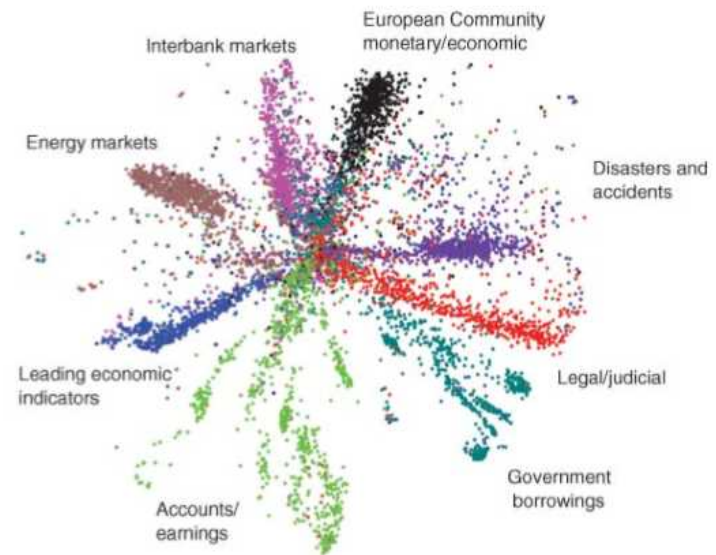
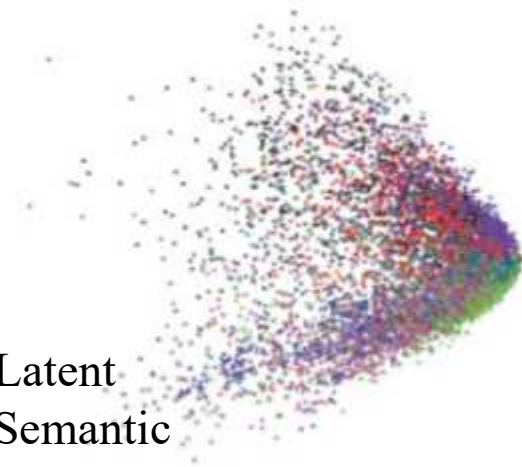




Principal
Component
Analysis

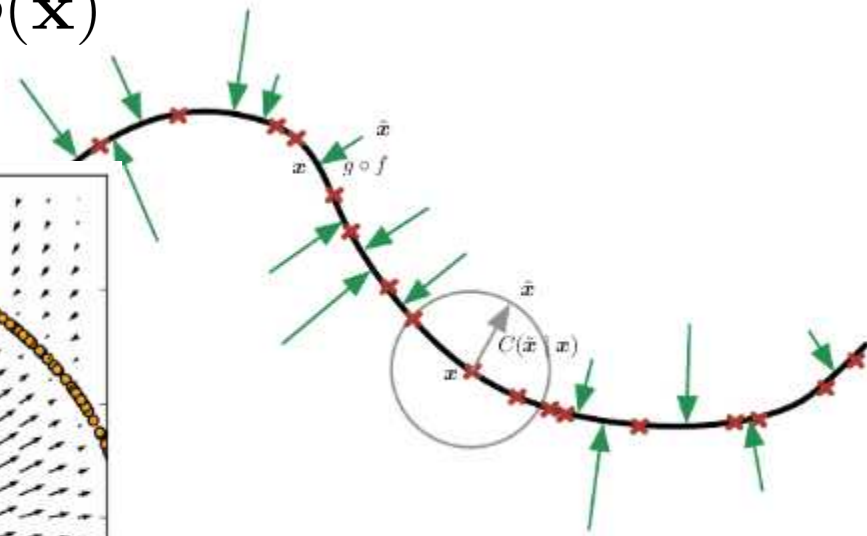
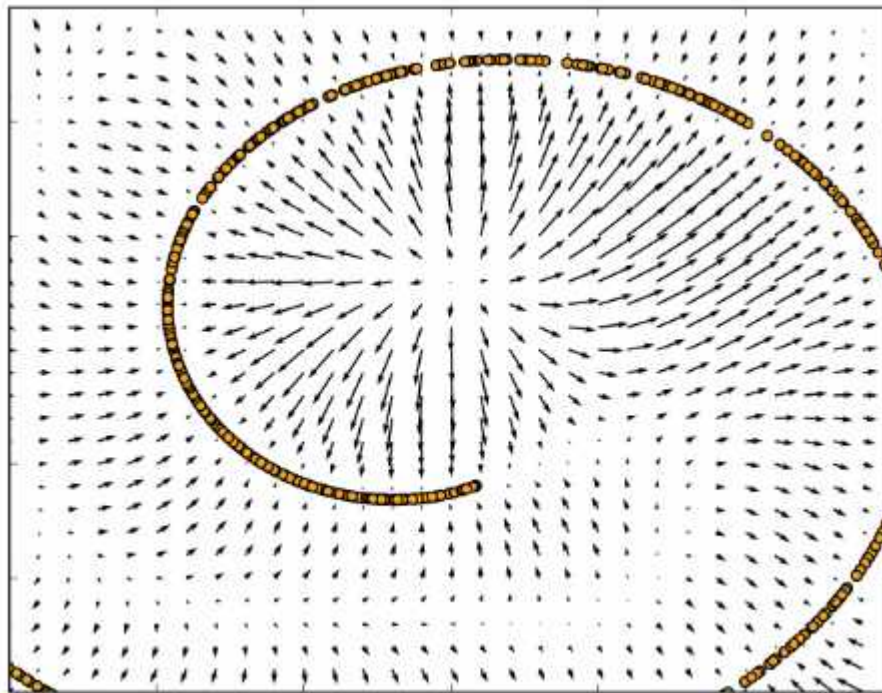


Latent
Semantic
Analysis

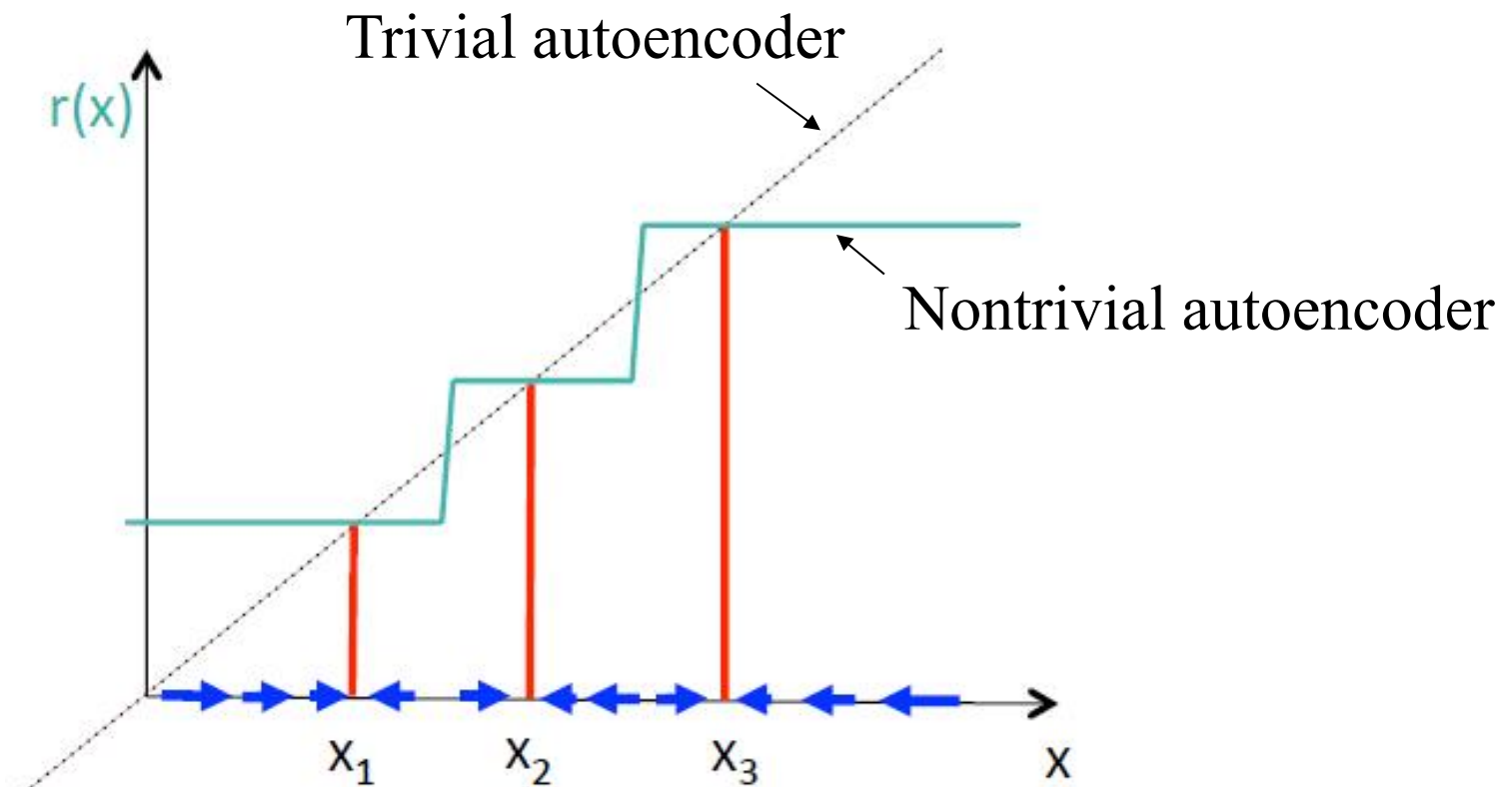


Denoising Autoencoder

- With Gaussian noise,
output – input = $\nabla_{\mathbf{x}} \log p(\mathbf{x})$



Trivial vs. Nontrivial Autoencoder



Denoising Autoencoder

$$\begin{aligned}\mathcal{L}_{DAE} &= \int_{\mathbb{R}^D} \mathbb{E}_{\epsilon \sim \mathcal{N}(0, \sigma^2 I)} [p(\mathbf{x}) ||r(\mathbf{x} + \epsilon) - \mathbf{x}||^2] d\mathbf{x} \\ &= \mathbb{E}_{\epsilon} \left[\int_{\mathbb{R}^D} p(\mathbf{x}) ||r(\mathbf{x} + \epsilon) - \mathbf{x}||^2 d\mathbf{x} \right] \\ &= \mathbb{E}_{\epsilon} \left[\int_{\mathbb{R}^D} p(\tilde{\mathbf{x}} - \epsilon) ||r(\tilde{\mathbf{x}}) - \tilde{\mathbf{x}} + \epsilon||^2 d\tilde{\mathbf{x}} \right] \quad \tilde{\mathbf{x}} = \mathbf{x} + \epsilon\end{aligned}$$

Euler-Lagrange equation

$$\frac{\partial \mathcal{L}_{DAE}}{\partial r} = 0 \quad \rightarrow \quad \mathbb{E}_{\epsilon} [p(\tilde{\mathbf{x}} - \epsilon) \cdot 2(r(\tilde{\mathbf{x}}) - \tilde{\mathbf{x}} + \epsilon)] = 0$$

$$r^*(\mathbf{x}) = \frac{\mathbb{E}_{\epsilon} [p(\mathbf{x} - \epsilon) \cdot (\mathbf{x} - \epsilon)]}{\mathbb{E}_{\epsilon} [p(\mathbf{x} - \epsilon)]}$$

Denoising Autoencoder

$$p(\mathbf{x} - \epsilon) \approx p(\mathbf{x}) - \epsilon^\top \frac{\partial p(\mathbf{x})}{\partial \mathbf{x}} + \frac{1}{2} \epsilon^\top \frac{\partial^2 p(\mathbf{x})}{\partial \mathbf{x}^2} \epsilon$$

$$\begin{aligned} \mathbb{E}_\epsilon [p(\mathbf{x} - \epsilon) \cdot (\mathbf{x} - \epsilon)] &= \int p(\mathbf{x} - \epsilon)(\mathbf{x} - \epsilon)p(\epsilon)d\epsilon \\ &= \mathbf{x}p(\mathbf{x}) + \frac{\mathbf{x}\sigma^2}{2} \nabla^2 p + \sigma^2 \nabla p \end{aligned}$$

$$\begin{aligned} \mathbb{E}_\epsilon [p(\mathbf{x} - \epsilon)] &= \int p(\mathbf{x} - \epsilon)p(\epsilon)d\epsilon \\ &= p(\mathbf{x}) + \frac{\sigma^2}{2} \nabla^2 p \end{aligned}$$

Denoising Autoencoder

$$r(\mathbf{x}) = \frac{\mathbb{E}_{\epsilon} [p(\mathbf{x} - \epsilon) \cdot (\mathbf{x} - \epsilon)]}{\mathbb{E}_{\epsilon} [p(\mathbf{x} - \epsilon)]} = \frac{\mathbf{x}p(\mathbf{x}) + \frac{\mathbf{x}\sigma^2}{2}\nabla^2 p + \sigma^2\nabla p}{p(\mathbf{x}) + \frac{\sigma^2}{2}\nabla^2 p}$$

$$\approx \mathbf{x} + \sigma^2 \frac{\nabla p}{p} = \mathbf{x} + \sigma^2 \nabla \log p$$

$$\frac{r(\mathbf{x}) - \mathbf{x}}{\sigma^2} = \nabla \log p$$

Theano Tutorial

deeplearning.net/tutorial/dA.html

DeepLearning 0.1 documentation »

previous | next | index

Denoising Autoencoders (dA)

Note

This section assumes the reader has already read through [Classifying MNIST digits using Logistic Regression](#) and [Multilayer Perceptron](#). Additionally it uses the following Theano functions and concepts : [T.tanh](#), [shared variables](#), [basic arithmetic ops](#), [T.grad](#), [Random numbers](#), [floatX](#). If you intend to run the code on GPU also read [GPU](#).

Note

The code for this section is available for download [here](#).

The Denoising Autoencoder (dA) is an extension of a classical autoencoder and it was introduced as a building block for deep networks in [\[Vincent08\]](#). We will start the tutorial with a short discussion on [Autoencoders](#).

Autoencoders

See section 4.6 of [\[Bengio09\]](#) for an overview of auto-encoders. An autoencoder takes an input $\mathbf{x} \in [0, 1]^d$ and first maps it (with an *encoder*) to a hidden representation $\mathbf{y} \in [0, 1]^{d'}$ through a deterministic mapping, e.g.:

$$\mathbf{v} = s(\mathbf{W}\mathbf{x} + \mathbf{b})$$

Table Of Contents

- Denoising Autoencoders (dA)
 - Autoencoders
 - Denoising Autoencoders
 - Putting it All Together
 - Running the Code

Previous topic

Convolutional Neural Networks (LeNet)

Next topic

Stacked Denoising Autoencoders (SdA)

This Page

Show Source

Quick search

Go

<http://deeplearning.net/tutorial/dA.html>



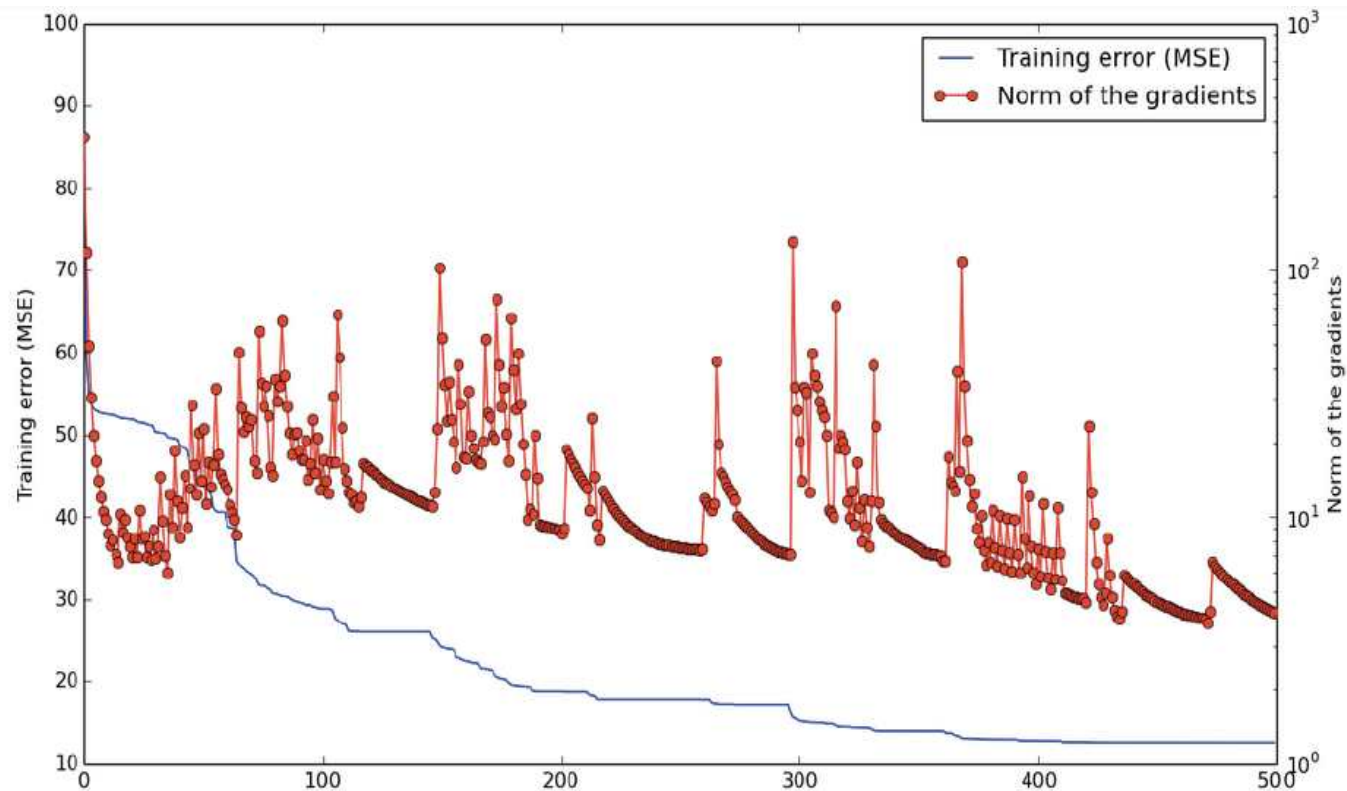
What Brought Successes to Deep Learning

- Many data and back propagation (Ruslan Salakhutdinov at NIPS 2013 workshop)
- We found a better way to initialize the weights, but that's sort of the minor things. (Geoffrey Hinton, at NIPS 2012 invited talk)
- Multilayer feedforward networks are universal approximators (Kurt Hornik, 1981)
- Deep Belief Networks are compact universal approximators (Yoshua Bengio, 2010)



Local Minima in ANN

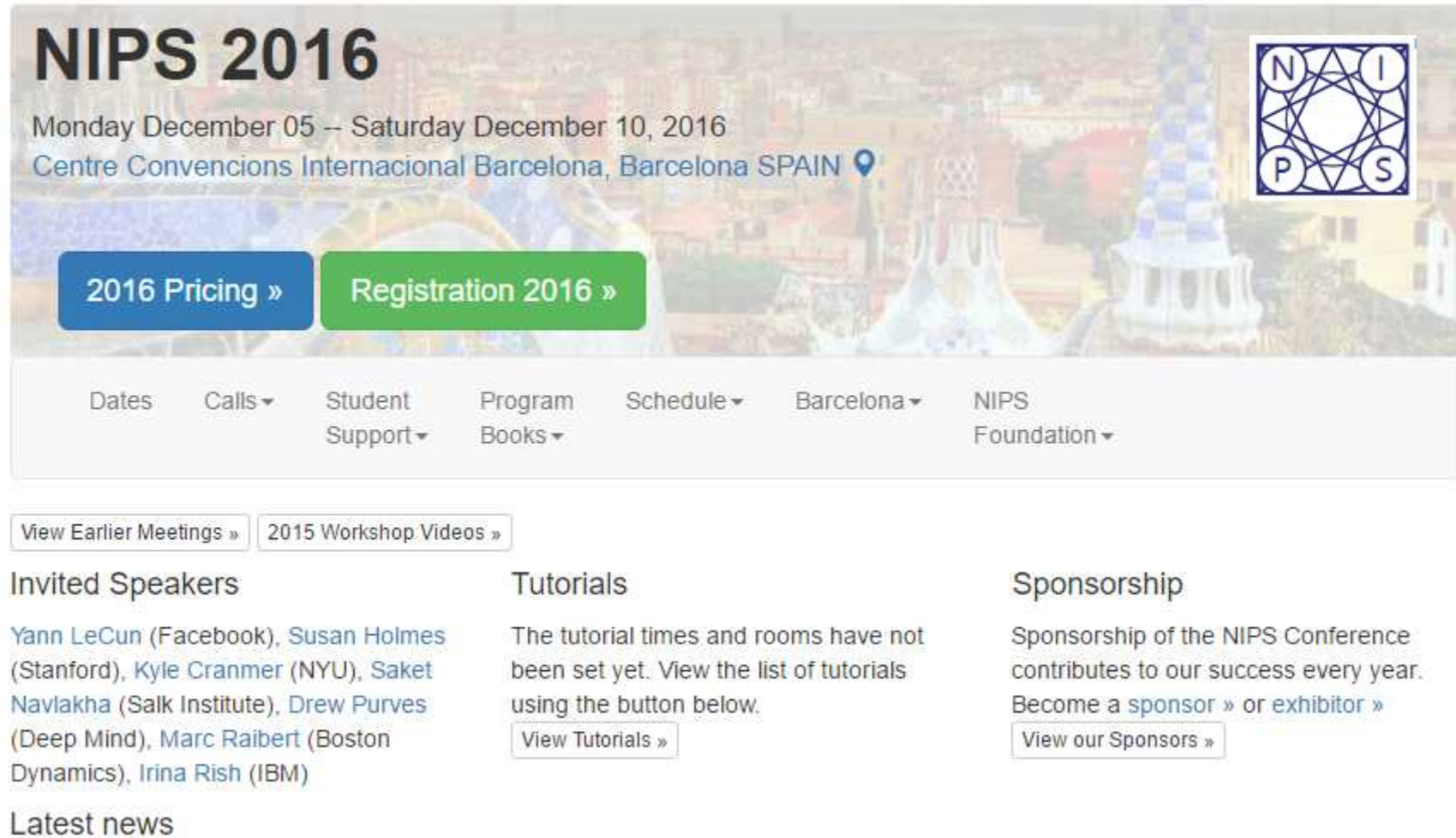
- Saddle point analysis



Razvan Pascanu et al. "On the saddle point problem for non-convex optimization." arXiv preprint arXiv:1405.4604 (2014)
Anna Choromanska et al. "The loss surfaces of multilayer networks." arXiv preprint arXiv:1412.0233 (2014)

Machine Learning Conferences

- Neural Information Processing Systems



NIPS 2016

Monday December 05 -- Saturday December 10, 2016
Centre Convencions Internacional Barcelona, Barcelona SPAIN

[2016 Pricing »](#) [Registration 2016 »](#)

[Dates](#) [Calls ▾](#) [Student Support ▾](#) [Program Books ▾](#) [Schedule ▾](#) [Barcelona ▾](#) [NIPS Foundation ▾](#)

[View Earlier Meetings »](#) [2015 Workshop Videos »](#)

Invited Speakers

[Yann LeCun](#) (Facebook), [Susan Holmes](#) (Stanford), [Kyle Cranmer](#) (NYU), [Saket Navekha](#) (Salk Institute), [Drew Purves](#) (Deep Mind), [Marc Raibert](#) (Boston Dynamics), [Irina Rish](#) (IBM)

Tutorials

The tutorial times and rooms have not been set yet. View the list of tutorials using the button below.

[View Tutorials »](#)

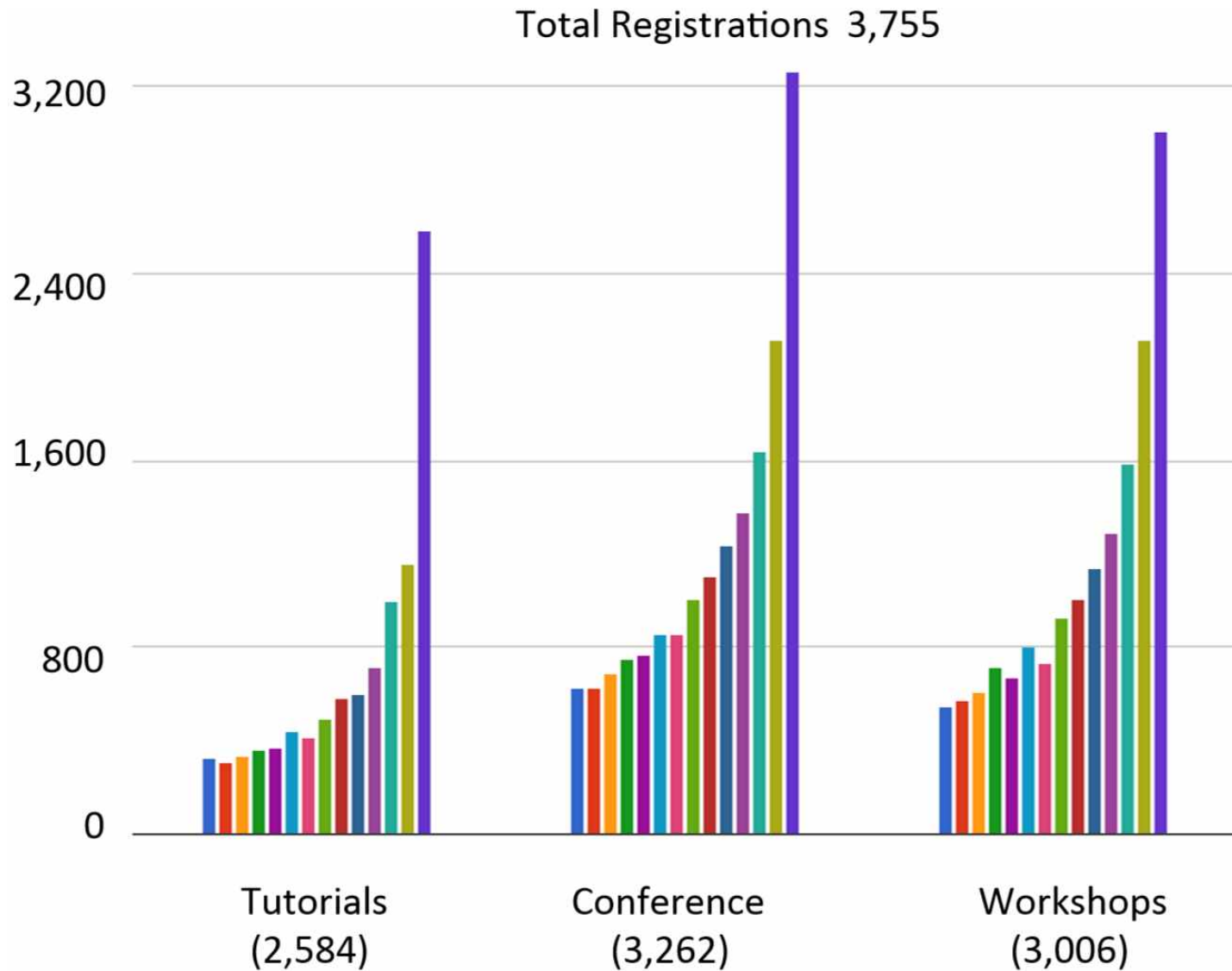
Sponsorship

Sponsorship of the NIPS Conference contributes to our success every year. Become a [sponsor »](#) or [exhibitor »](#)

[View our Sponsors »](#)

Latest news

Neural Information Processing Systems (NIPS) Growth



Neural Information Processing Systems (NIPS 2015)

- Oral talks: 15
- Spotlights: 37
- Accepted papers: 403
- Single session: more than 3000 participants are listening to the single presentation.
- 7pm – 12am (5hr) poster session every day

*Look at the poster session
how it does look →*



From Neil Lawrence's Blog

The New York Times

December 11, 2015

....

Last month, the Toyota Motor Corporation announced a five-year, billion-dollar investment to create a research center based next to Stanford University to focus on artificial intelligence and robotics.

Also, a formerly obscure academic conference, Neural Information Processing Systems, underway this week in Montreal, has doubled in size since the previous year and has attracted a growing list of brand-name corporate sponsors, including Apple for the first time.

“There is a sellers’ market right now — not enough talent to fill the demand from companies who need them,” said Terrence Sejnowski, the director of the Computational Neurobiology Laboratory at the Salk Institute for Biological Studies in San Diego. “Ph.D. students are getting hired out of graduate schools for salaries that are higher than faculty members who are teaching them.”



Machine Learning Conferences

- International Conference on Machine Learning (ICML)

ICML @ Sydney

Thirty-fourth International Conference on Machine Learning

[Year \(2017\) ▾](#)

[Help ▾](#)

[My Registrations](#)

[Profile ▾](#)

[Sponsor Info](#)

[Exhibitor Info](#)

[Future Meetings](#)

[Follow](#)

THE CONFERENCE

- [Schedule](#)
- [Invited Speakers](#)
- [Tutorials](#)
- [Workshops](#)
- [Accepted Papers](#)
- [Committees](#)
- [Sponsorship](#)

FOR AUTHORS

- [Call for Papers](#)
- [Call for Tutorials](#)
- [Call for Workshops](#)
- [Camera Ready](#)
- [Style and Author Instructions](#)
- [Reviewer Guidelines](#)

FOR PARTICIPANTS

- [Pricing](#)
- [Registration](#)
- [Student Volunteers](#)
- [Student Travel Support](#)
- [Sponsors](#)
- [Venue And Local](#)
- [Book a Room](#)
- [Poster](#)
- [VISA Info](#)

NOTICE: The 2017 ICML conference may close registration ahead of the declared deadline in case it reaches the capacity limits of the convention center. There may be no onsite registration. Please register early.



EARLY REGISTRATION ENDS JUNE 22nd!!

[International Convention Centre, Sydney AUSTRALIA](#)



Important Dates:

Sunday August 06 -- Friday August 11, 2017

[Sydney Virtual Tour »](#)

About:

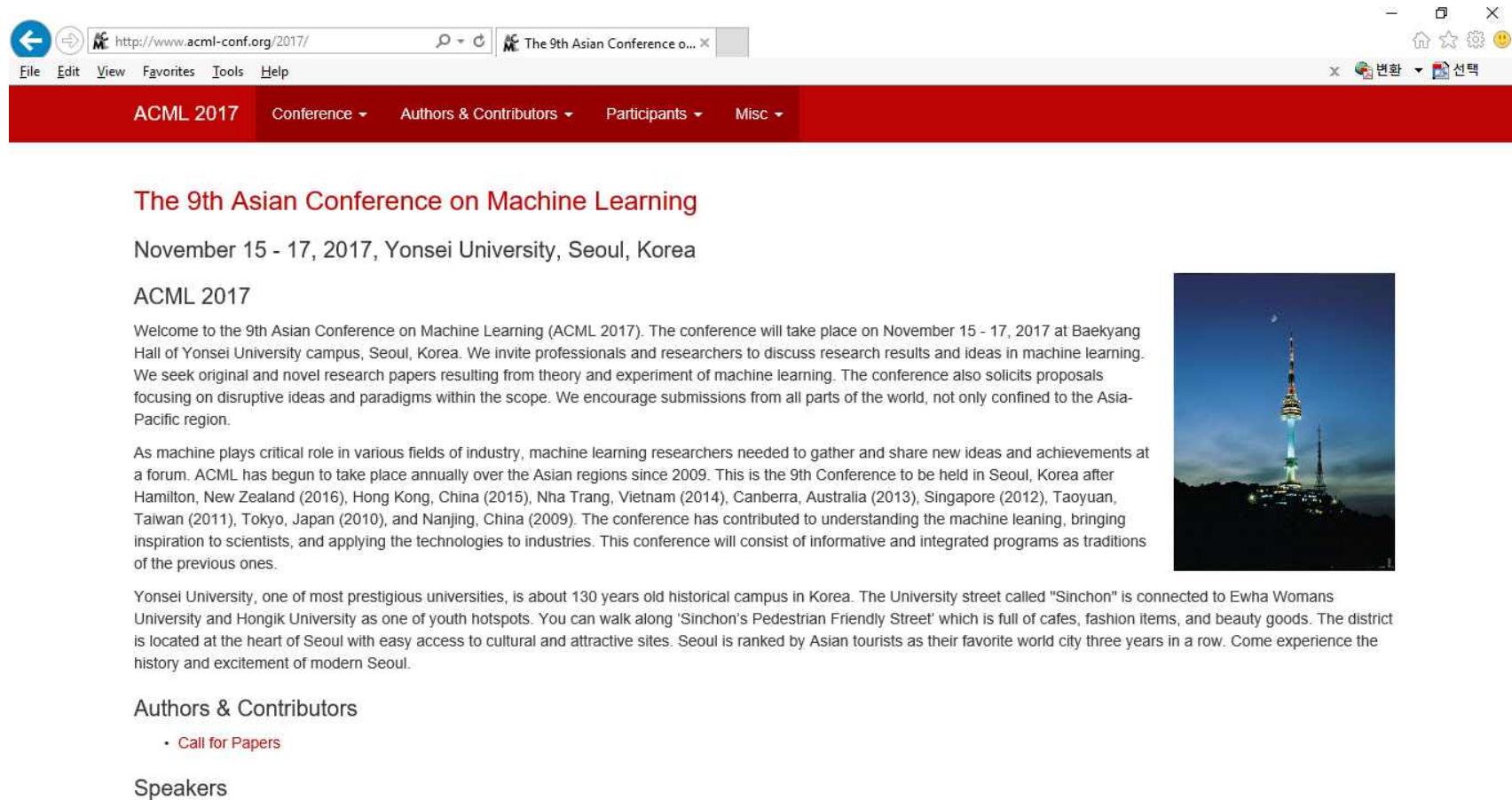


Machine Learning Conferences

- AI & Statistics (AIStats)
- Association for the Advancement of Artificial Intelligence (AAAI) Conference
- Conference on Uncertainty in Artificial Intelligence (UAI)
- European Conference on Machine Learning (ECML)
- Asian Conference on Machine Learning (ACML)

Asian Conference on Machine Learning (ACML) in Seoul

- <http://www.acml-conf.org/2017/>
- Nov. 15 - 17 (Wed. – Fri.), 2017



The screenshot shows a web browser window displaying the ACML 2017 website. The browser's address bar shows the URL <http://www.acml-conf.org/2017/>. The website has a red header with navigation links: ACML 2017, Conference, Authors & Contributors, Participants, and Misc. The main content area features the title "The 9th Asian Conference on Machine Learning" in red, followed by the dates "November 15 - 17, 2017, Yonsei University, Seoul, Korea". Below this is a section titled "ACML 2017" with a welcome message and details about the conference. A photograph of the N Seoul Tower at night is positioned to the right of the text. Further down, there are sections for "Authors & Contributors" with a "Call for Papers" link, and "Speakers".

The 9th Asian Conference on Machine Learning

November 15 - 17, 2017, Yonsei University, Seoul, Korea

ACML 2017

Welcome to the 9th Asian Conference on Machine Learning (ACML 2017). The conference will take place on November 15 - 17, 2017 at Baekyang Hall of Yonsei University campus, Seoul, Korea. We invite professionals and researchers to discuss research results and ideas in machine learning. We seek original and novel research papers resulting from theory and experiment of machine learning. The conference also solicits proposals focusing on disruptive ideas and paradigms within the scope. We encourage submissions from all parts of the world, not only confined to the Asia-Pacific region.

As machine plays critical role in various fields of industry, machine learning researchers needed to gather and share new ideas and achievements at a forum. ACML has begun to take place annually over the Asian regions since 2009. This is the 9th Conference to be held in Seoul, Korea after Hamilton, New Zealand (2016), Hong Kong, China (2015), Nha Trang, Vietnam (2014), Canberra, Australia (2013), Singapore (2012), Taoyuan, Taiwan (2011), Tokyo, Japan (2010), and Nanjing, China (2009). The conference has contributed to understanding the machine learning, bringing inspiration to scientists, and applying the technologies to industries. This conference will consist of informative and integrated programs as traditions of the previous ones.

Yonsei University, one of most prestigious universities, is about 130 years old historical campus in Korea. The University street called "Sinchon" is connected to Ewha Womans University and Hongik University as one of youth hotspots. You can walk along 'Sinchon's Pedestrian Friendly Street' which is full of cafes, fashion items, and beauty goods. The district is located at the heart of Seoul with easy access to cultural and attractive sites. Seoul is ranked by Asian tourists as their favorite world city three years in a row. Come experience the history and excitement of modern Seoul.

Authors & Contributors

- [Call for Papers](#)

Speakers



Deep Learning book

Deep Learning

An MIT Press book

Ian Goodfellow and Yoshua Bengio and Aaron Courville

[Exercises](#) [Lectures](#)

The Deep Learning textbook is a resource intended to help students and practitioners enter the field of machine learning in general and deep learning in particular. The online version of the book is now complete and will remain available online for free.

The deep learning textbook can now be pre-ordered on [Amazon](#). Pre-orders should ship on December 16, 2016.

For up to date announcements, join our [mailing list](#).

Citing the book

To cite this book, please use this bibtex entry:

```
@unpublished{Goodfellow-et-al-2016-Book,  
  title={Deep Learning},  
  author={Ian Goodfellow and Yoshua Bengio and Aaron Courville},  
  note={Book in preparation for MIT Press},  
  url={http://www.deeplearningbook.org},  
  year={2016}  
}
```

<http://www.deeplearningbook.org/>



THANK YOU

Yung-Kyun Noh
nohyung@snu.ac.kr

