

# **EE559/EE660 - Pattern Recognition and Machine Learning**

EE559 Pattern Recognition

1. Introduction
  - basic concept
  - features
  - Discriminant function:
  - Gradient Descent
  - Complexity: Degree of Freedom, VC dimension
  - Lagrange Optimization (Basic for SVM)
  - Feature selection
2. Distribution-Free Classification Methods
  - Perceptron
  - SVM
3. Validation and data reduction
  - validation
  - feature selection & dimension reduction
4. Statistical Classification
  - Basic conception
  - Bayes Minimum-Error classification
  - Density Estimate
  - Parameter Estimate
5. Artifical Neural Network
  - basic elements
  - activation function
  - neural network classify algorithm

EE660 Machine Learning

1. Introduction
  - Basic Concept
  - Bayes Estimation
2. Algorithm: Regression
  - Bayesian Regression
  - Logistic Regression
3. Complexity
  - feasibility of learning (theory of hypothesis set)
  - Dichotomies
  - VC Dimension

- Bias vs Variance
- 4. Fundation of Model and Training procedure
  - Whole Training Method
  - Overfitting
  - Regularization
  - Model Selection
  - Validation
- 5. Nonlinear Method
  - Decision Tree and Random Forest
  - Boosting
- 6. Semi-Supervised Learning
  - Self-Training
  - Propagating 1-Nearest-Neighbour
  - Mixture Model and Parameter Classification
- 7. Unsupervised Learning
  - Density Estimate
  - Clustering
  - Clustering Measurement

# EE559 Pattern Recognition

## 1. Introduction

### basic concept

Training/Classification procedure

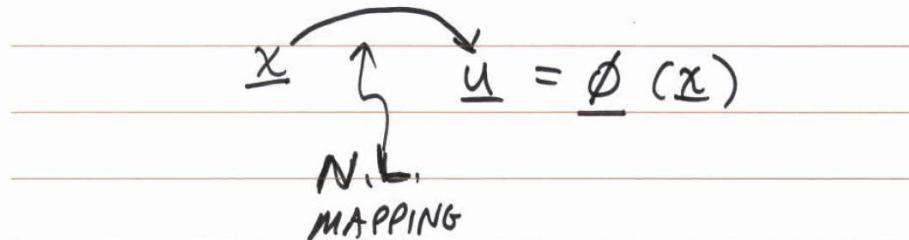
<https://courses.uscden.net/d2l/le/content/12552/viewContent/181665/View?ou=12552>

feature space, discriminant function

totally separated, pairwise separated

augment space

(i) E MACHINE (OR NONLINEAR TRANSFORMATION)

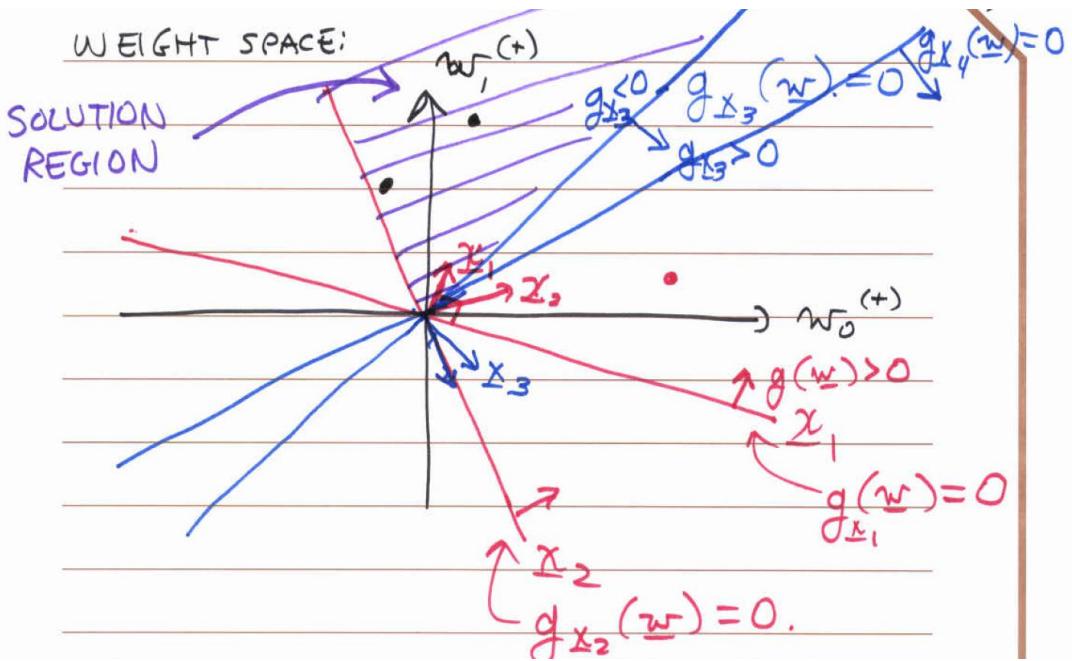


weight space

discriminate function: g(x)

criterion functions: J(w) (cost function is one kind of criterion function)

each line means a pair of weight



phi - machine (Non-linear discrimination function): X-space to U-space

## features

An input data point, showing its vector components (features):

$$\underline{x} = \left( x_1, x_2, \dots, x_j, \dots, x_D \right)^T$$

$\underline{x}_n^{(k)} = \left( x_{n1}^{(k)}, x_{n2}^{(k)}, \dots, x_{nj}^{(k)}, \dots, x_{nD}^{(k)} \right)^T$

$n^{\text{th}}$  DATA PT.  $= n^{\text{th}}$  data point of class  $S_k$   $\Rightarrow$  VAR. OF BEATS/MIN.

Thus,  $x_{nj}^{(k)}$  =  $j^{\text{th}}$  feature of  $n^{\text{th}}$  data point of class  $S_k$

## Discriminant function:

used to discriminate ( $>$  or  $<$  0) class (not minimize)

two class:

DEFINE  $g(\underline{x})$ , SUCH THAT:

DECISION RULE:

$$g(\underline{x}) > 0 \Rightarrow \underline{x} \in S_1$$

$$g(\underline{x}) < 0 \Rightarrow \underline{x} \in S_2$$

$$g(\underline{x}) = 0 \Rightarrow \underline{x} \text{ ON DEC. BOUNDARY}$$

multi class:

one vs one

ALTERNATE DECISION RULE ("VOTE") :

$$\underline{x} \in S_i \text{ IFF } i = \arg \max_j \sum_{\substack{k=1 \\ k \neq j}}^C [g_{jk}(\underline{x}) > 0]$$

IN WHICH  $[u]$  DENOTES "INDICATOR FUNCTION":

$$[u] \triangleq \begin{cases} 1 & \text{IF } u \text{ IS TRUE} \\ 0 & \text{IF } u \text{ IS FALSE.} \end{cases}$$

i.e., FOR  $\underline{x}$ , PICK THE CLASS WITH THE MOST

one vs rest

EACH IS FOR A BINARY (2-CLASS)

CLASSIFIER,

$\Rightarrow$  SOLVE C 2-CLASS PROBLEMS:

$$S'_i \text{ VS. } \overline{S}'_i \quad (i=1 \Rightarrow \text{ROCK VS. NOT ROCK})$$
$$\forall i=1, 2, \dots, C.$$

COMBINE RESULTS:

DECISION RULE (DEFAULT):

$$\underline{x} \in S_i \text{ IFF } \underline{x} \in S'_i \text{ AND } \underline{x} \in \overline{S}'_k \quad \forall k \neq i.$$

Maximum Value Method

## MAXIMAL VALUE METHOD (MVM)

(ALSO CALLED, FOR LINEAR CASE, A  
"LINEAR MACHINE").

EACH CLASS:  $g_k(x)$  (FOR  $S_k$ )  
DECISION RULE:  
 $x \in S_k$  IFF  $g_k(x) > g_j(x) \quad \forall j \neq k.$   
DECISION BOUNDARIES ARE GIVEN BY:

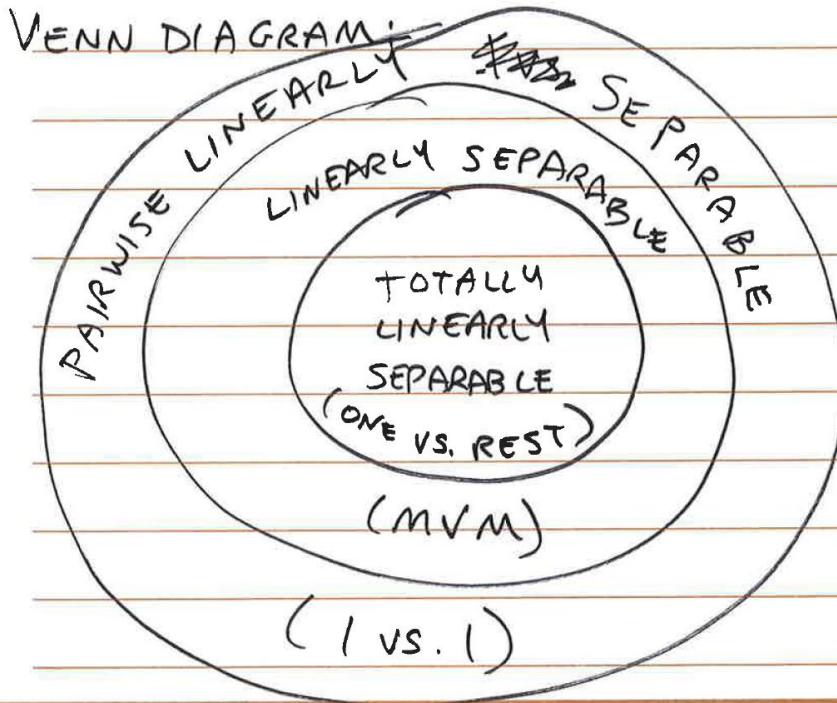
MVM

overall:

totally linear separable: a line can separate one class with all the rest

linear separable: a line can separate one class with other class

pairwise separable: some lines can separate one class with other class



### Criterion Function

loss function is one way of criterion function

criterion function can be based on misclassified data points

### Gradient Descent

### Complexity: Degree of Freedom, VC dimension

## DEGREES OF FREEDOM (d.o.f.) IN A LEARNING

PROBLEM IS THE NUMBER OF VARIABLES THAT THE LEARNING ALGORITHM CAN ADJUST INDEPENDENTLY.

$$\text{Ex: } y(x) = \underline{w}^T \underline{x} \quad \text{2-class.}$$

$$\begin{aligned} \text{d.o.f.} &= \cancel{\# \text{WEIGHT VALUES}} \\ &= D+1. \end{aligned}$$

CONSTRAINTS : CONSTRAINT CHOICES OF  $\underline{w}$  OR OTHER VARIABLES.

$$\text{e.g.: } \underline{x} \underline{w} = b \text{ IS A}$$

CAN BE FORMALIZED MATHEMATICALLY

"VC DIMENSION" [VAPNIK-CHERVENENKIS], WHICH QUANTIFIES THE "FLEXIBILITY" OR "CAPACITY" OF A DECISION BOUNDARY.

EXAMPLES:

• LINEAR BOUNDARY IN  $(D+1)$ -SPACE

[ $D$  FEATURES]

$$\text{VCdim} = D+1$$

$$\Rightarrow \text{d.o.f.} = \# \text{weights} = D+1.$$

## Lagrange Optimization (Basic for SVM)

SET UP A LAGRANGIAN FUNCTION:

$$L(\underline{x}, \lambda) = f(\underline{x}) + \lambda g(\underline{x}).$$

↑  
ORIG.  
FCN.

LAGRANGE  
MULTIPLIER.

EQUALITY  
CONSTRAINT:  
 $g=0$ .

NOTE: WHEN CONSTRAINT IS SATISFIED,

$$L(\underline{x}, \lambda) = f(\underline{x}).$$

LAGRANGE METHOD:

$$(*) \quad \boxed{\nabla_{\underline{x}, \lambda} L(\underline{x}, \lambda) = \underline{0}}$$

$$\Rightarrow \begin{cases} \nabla_{\underline{x}} L(\underline{x}, \lambda) = \nabla_{\underline{x}} f(\underline{x}) + \nabla_{\underline{x}} \lambda g(\underline{x}) = \underline{0} \\ \nabla_{\lambda} L(\underline{x}, \lambda) = g(\underline{x}) = 0 \end{cases}$$

IS OUR  
CONSTRAINT.

## Feature selection

## 2. Distribution-Free Classification Methods

### Perceptron

PERCEPTRON LEARNING ALGORITHM

$$\rightarrow J = - \sum_{\underline{x}_i \in \mathcal{X}} \underbrace{\underline{w}^T \underline{z}_i \underline{x}_i}_{\text{all negative}}$$

MINIMIZE  $J(\underline{w})$  BY GRADIENT DESCENT.

$$\nabla_{\underline{w}} J(\underline{w}) = - \sum_{x_n \in \chi} z_n x_n$$

$$\underline{w}(i+1) = \underline{w}(i) + \eta(i) \sum_{x_n \in \chi} z_n x_n$$

$\chi$  IS SET OF TRAINING DATA PTS. MISCLASSIFIED  
BY  $H_{\underline{w}(i)}$

$$\eta(i) \geq 0 \quad \forall i.$$

STOP WHEN  $\chi \neq \emptyset \Rightarrow [\underline{w}(i+1) = \underline{w}(i)]$

### PERCEPTRON LEARNING USING BATCH

#### GRADIENT DESCENT (GD) (OR BATCH UPDATE)

WE CAN INSTEAD UPDATE  $\underline{w}$  AFTER EACH DATA POINT,

NOTE:  $J(\underline{w}) = \sum_{n=1}^N J_n(\underline{w})$

$\underbrace{\phantom{\sum_{n=1}^N} J_n(\underline{w})}_{\begin{array}{l} \text{crit. fcn. based on} \\ | \text{ training pt. } (x_n) \end{array}}$

$$J_n(\underline{w}) = -\underline{w}^\top z_n x_n \underbrace{[\underline{w}^\top z_n x_n \leq 0]}_{| \text{ IF TRUE}}$$

SO WE CAN DO | ITERATION OF GRADIENT DESCENT U IF FALSE.

FOR EACH DATA PT.:

$$\underline{w}(i+1) = \underline{w}(i) - \eta(i) \nabla_{\underline{w}} J_n(\underline{w})$$

$$\nabla_{\underline{w}} J_n(\underline{w}) = -z_n x_n [\underline{w}^T z_n x_n \leq 0]$$

$$\Rightarrow \begin{cases} \underline{w}(i+1) = \underline{w}(i) + \eta(i) z_n x_n, & \text{IF } \underline{w}^T z_n x_n \leq 0 \\ \underline{w}(i+1) = \underline{w}(i), & \text{IF } \underline{w}^T z_n x_n > 0. \end{cases}$$

BASIS OF SEQUENTIAL GRADIENT DESCENT (SGD)

AND STOCHASTIC G.D.

batch gradient descent

Iterate until convergence

$$\underline{w}(i+1) = \underline{w}(i) + \eta(i) \sum_{\substack{\text{all } n \text{ s.t.} \\ x_n \in \mathcal{X}}} z_n x_n, \quad \text{in which } \mathcal{X} \text{ is the set of training}$$

points misclassified by  $H_{\underline{w}(i)}$ , and  $\eta(i) \geq 0^*$ .

Stop when  $\mathcal{X} = 0$

sequential gradient descent

Randomly shuffle the order of training data points

Initialize  $\underline{w}(0)**$

Define one epoch as:

For  $n$  in  $\{1, 2, \dots, N\}$

$$i = n - 1$$

$$\begin{cases} \underline{w}(i+1) = \underline{w}(i) + \eta(i) z_n x_n & \text{if } \underline{w}^T z_n x_n \leq 0 \\ \underline{w}(i+1) = \underline{w}(i) & \text{if } \underline{w}^T z_n x_n > 0 \end{cases}$$

Iterate over epochs until no change in  $\underline{w}$  over 1 full epoch

(ALSO CALLED "SINGLE SAMPLE" UPDATE, OR  
"ONE-AT-A-TIME" UPDATE.)

stochastic gradient descent

### Stochastic GD – variant 1 (randomly shuffle before each epoch)

Iterate over epochs until convergence:

- (i) Randomly shuffle dataset
- (ii) Perform single-sample updates over 1 epoch

### Stochastic GD – variant 2 (choose each data point randomly, with replacement)

Iterate over data points until convergence:

- (i) Randomly pick a training data point (with replacement)
- (ii) Perform single-sample update

### Mini-Batch GD (compromise between batch GD and stochastic GD)

Iterate until convergence:

- (i) Pick  $M$  data points out of  $N$  at random  $(M < N)$
- (ii) Perform batch-update GD on the  $M$  data points (1 iteration)
- (iii) Replace the data points to the training set

if use MSE (1,0) rather than misclassified data, linear regression

$\underline{w}(i+1) = \underline{w}(i) + \gamma(i) [b_n - \underline{w}^\top(i) z_n x_n]$

$n = (i \bmod N) + 1$  (BASIC SEQUENTIAL G.D.)

$\underline{w}(0) = \text{ARBITRARY}$ .

$\gamma(i) > 0 \quad \forall i$ .

WIDROW-HOFF LEARNING

### SVM

aim: maximize the margin

support vector: closest data to the boundary ( $g(x)=0$ )

SPARSE DATA  
INCLUDING  
UNDER-  
DETERMINED  
SYSTEMS).

ADDING CONSTRAINT(S)  
TO THE LEARNING  
PROBLEM

SVM

NONLINEAR MAPPING  
TO EXPANDED FEATURE

(ii) OPTIMIZED VERSION OF USING A (SAFETY)  
MARGIN.

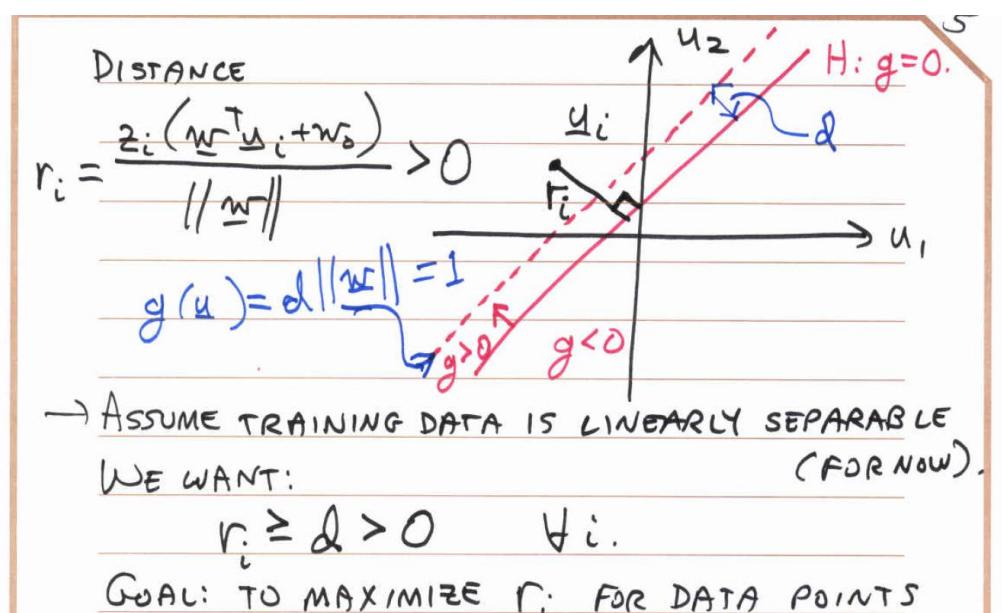
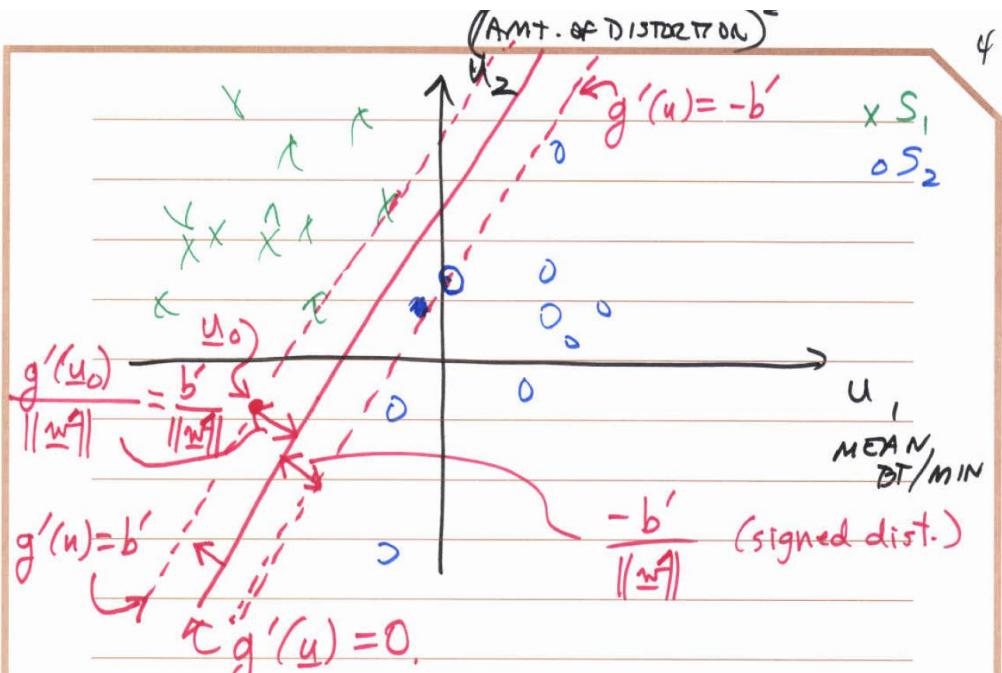
$$g(\underline{x}) = \underline{w}^T \underline{x} + w_0 \geq b > 0$$

FOR  $\underline{x} \in S_1$ .

$$g'(\underline{u}) = \underline{w'}^T \underline{u} + w'_0 \geq b' > 0$$

FOR  $\underline{u} \in S_1$ .

SVM CHOOSES THE DECISION BOUNDARY  
(IN EXPANDED FEAT. SPACE) THAT  
MAXIMIZES THE MARGIN  $b'$ .



to reduce complexity, add a constrain, what we don't need to define  $d$  (minimal distance)

To REDUCE THIS d.o.f. IMPOSE CONSTRAINT:

$$\boxed{\|w\| \cdot d = 1}$$

training method:

REQUIRE  $z_i(\underline{w}^T \underline{u}_i + w_0) \geq 1 \quad \forall i$

$$d = \frac{1}{\|\underline{w}\|}$$

WITH MAXIMAL  $d$ , OR

WITH MINIMAL  $\|\underline{w}\|$ .

DATA POINTS WILL SATISFY:

$$z_i(\underline{w}^T \underline{u}_i + w_0) = 1$$

(AT EDGE OF MARGIN), THESE ARE  
THE SUPPORT VECTORS.

### SVM learning

transfer minimal  $\|\underline{w}\|$  with requirement  $Zi(\underline{w}^T \underline{u}_i + w_0) \geq 0$  into a lagrange equation and find the optimal

SET UP  $L: J_p \quad J(\underline{w}) = \frac{1}{2} \|\underline{w}\|^2$  (by constraints)

"PRIMAL FORM"  $\rightarrow$   
 $L(\underline{w}, w_0, \lambda) = \frac{1}{2} \|\underline{w}\|^2 - \sum_{i=1}^N \lambda_i [z_i(\underline{w}^T \underline{u}_i + w_0) - 1]$   
 $\lambda_i \geq 0 \quad \forall i$ .  
 $\lambda_i [z_i(\underline{w}^T \underline{u}_i + w_0) - 1] = 0 \quad \forall i$

$$(z_i(\underline{w}^T \underline{u}_i + w_0) - 1 \geq 0 \quad \forall i)$$

$\min L() \text{ w.r.t } \underline{w}, w_0 \mid \max L() \text{ w.r.t } \lambda$

$$L_D(\lambda) = -\frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \lambda_i \lambda_j z_i z_j y_i^T u_j + \sum_{i=1}^N \lambda_i$$

WITH:  $\sum_{i=1}^N \lambda_i z_i = 0$

$$\lambda_i \geq 0 \quad \forall i$$

$$\underline{w}^* = \sum_{i=1}^N \lambda_i z_i u_i$$

$$\lambda_i [z_i (\underline{w}^{*T} u_i + w_0) - 1] = 0 \quad \forall i$$

ONLY HAVE TO OPTIMIZE w.r.t.  $\lambda$ .  
YIELDS SOLUTION  $\underline{\lambda}^*$ .

slack variable

INTRODUCE SLACK VARIABLES  $\xi_i, i=1, 2, \dots, N$ ,

SUCH THAT:

$\xi_i = 0$  IF  $u_i$  IS ON CORRECT SIDE OF MARGIN

$\xi_i = \text{NORMALIZED DISTANCE } u_i \text{ TO MARGIN}$

BOUNDARY, IF ON INCORRECT SIDE  
OF MARGIN BOUNDARY.

$\xi_i = 1$  IF  $u_i$  IS ON DECISION BOUNDARY  $H$ .

Now, CRITERION Fcn width

BECOMES:

$$J(\underline{w}) = \frac{1}{2} \|\underline{w}\|^2 + C \sum_{i=1}^N \xi_i$$

NEW TERM.

CONSTRAINTS:

$$z_i (\underline{w}^\top \underline{u}_i + w_0) \geq 1 - \xi_i \quad \forall i$$

NEW TERM.

primal form

Now, LAGRANGIAN BECOMES:

$$\begin{aligned} L(\underline{w}, w_0, \underline{\lambda}, \underline{\mu}) &= \frac{1}{2} \|\underline{w}\|^2 + C \sum_{i=1}^N \xi_i \\ &- \sum_{i=1}^N \lambda_i [z_i (\underline{w}^\top \underline{u}_i + w_0) - 1 + \xi_i] \\ &- \sum_{i=1}^N \mu_i \xi_i \end{aligned}$$

$$\lambda_i \geq 0 \quad \forall i$$

$$\mu_i \geq 0 \quad \forall i$$

dual form

CAN BE EXPRESSED IN DUAL FORM :

$L_D(\underline{\lambda})$  HAS FORM AS BEFORE,  
EXCEPT THAT:

CONSTRAINTS:  $\lambda_i \geq 0 \quad \forall i$

BECOMES:  $0 \leq \lambda_i \leq C \quad \forall i$ .

nolinear mapping - kernel

$$L_D(\underline{\lambda}) = \sum_{i=1}^N \lambda_i - \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \lambda_i \lambda_j z_i z_j \underline{\underline{u}}_i^\top \underline{\underline{u}}_j$$

$$z_i z_j \underline{\underline{\phi}}^\top(\underline{x}_i) \underline{\underline{\phi}}(\underline{x}_j)$$

$$\begin{aligned} K(\underline{x}_i, \underline{x}_j) &= \underline{\underline{\phi}}^\top(\underline{x}_i) \underline{\underline{\phi}}(\underline{x}_j) \\ &= \underline{\underline{\text{KERNEL FUNCTION}}} \end{aligned}$$

$$L_D(\underline{\lambda}) = \sum_{i=1}^N \lambda_i - \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \lambda_i \lambda_j z_i z_j K(\underline{x}_i, \underline{x}_j)$$

COMMON KERNEL FUNCTIONS:

LINEAR:  $K_{\text{Linear}}(\underline{x}_i, \underline{x}_j) = \alpha \underline{x}_i^\top \underline{x}_j$ .

POLYNOMIAL:  $K_{\text{Poly}}(\underline{x}_i, \underline{x}_j) = (\underline{x}_i^\top \underline{x}_j + 1)^P, P \in \mathbb{Z}^+$

RADIAL BASIS FUNCTION:

$$K_{\text{RBF}}(\underline{x}_i, \underline{x}_j) = \exp\{-\gamma \|\underline{x}_i - \underline{x}_j\|^2\}, \quad \gamma > 0.$$

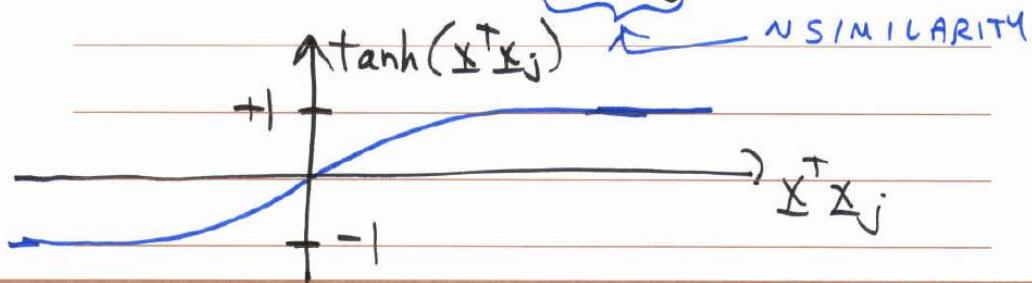
SIGMOID:

$$k_{\text{Sigmoid}}(x_i, x_j) = \tanh \left[ s(x_i^T x_j) + c \right]$$

Interpret:

$$\text{Let } s=1, c=0$$

$$k = \tanh(x_i^T x_j)$$



### 3. Validation and data reduction

#### validation

training set, test set, validation set

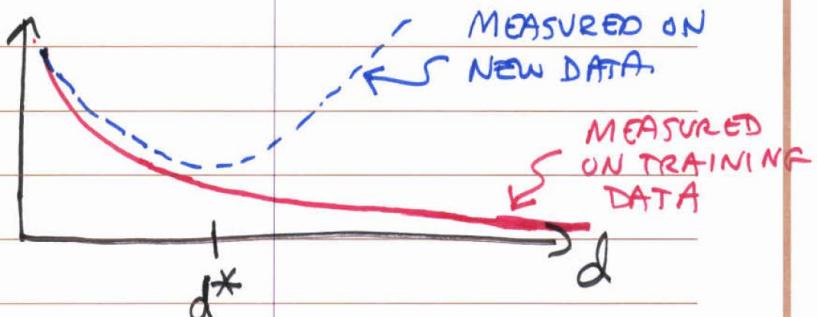
VALIDATION SET — FOR VALIDATION OR MODEL SELECTION

Ex: SUPPOSE — LINEAR CLASSIFIER

#d.o.f. = #weights = D+1.

PARAMETER d = #DIMENSIONS (VARIABLE).

NUMERICAL  
ERROR  
RATE



cross validation

## CROSS VALIDATION FOR ESTIMATION OF ERROR (OR PERFORMANCE) ON UNKNOWN

RE-USE (CAREFULLY) DATA POINTS IN  
THESE SUBSETS.

$D \rightarrow M$  NEQUAL-SIZE SUBSETS.

(TYPICALLY CHOSEN RANDOMLY, BUT  
USUALLY SUBJECT TO CONSTRAINT  
THAT % REPRESENTATION OF EACH  
CLASS IS EQUAL IN EACH

cross validation for parameter selection

## **feature selection & dimension reduction**

To solve too many D.o.F

two type:

choose (select) most influence feature: Sequential Forward Selection | remove high correlated features

transform to new features: PCA | LDA | ---> new features are linear combine of old features

### **1.PCA**

some times not good (because view all data in the space and transform them all)

(1)  $\rightarrow$  APPLY ORTHONORMAL TRANSFORMATION  
TO FIND EIGEN VECTORS & EIGEN VALUES (E.V.'S)  
OF THE COVARIANCE MATRIX \*

(2)  $\rightarrow$  ORDER THEM BY SIZE OF E.V.'S.

(3)  $\rightarrow$  KEEP P' LARGEST E.V. & EIGEN VECTORS.

\* USE SAMPLE COVARIANCE MATRIX :

$$\hat{\Sigma} = \frac{1}{N} \sum_{j=1}^N (\underline{x}_j - \underline{m})(\underline{x}_j - \underline{m})^T$$

(4) TRANSFORM ALL DATA PTS. TO NEW SPACE:

$$\underline{x}' = \underline{E}^T \underline{x}$$

## 2. Fisher linear discriminant (LDA, NDA)

for a 2-D two class dataset, transfer into a line, modify the difference between classes

for a N-D class dataset, build a N-1 dimension space, project data into this space and data has the maximize separation

→ FIND DIRECTION OF A LINE (1-D SPACE)

THAT MAXIMIZES THE SEPARABILITY OF PROJECTED TRAINING DATA, IN THE 1-D SPACE.

CRITERION:

$$J = \frac{[\text{DISTANCE BETWEEN CLASS MEANS}]^2}{[\text{VARIANCE OF } S_1 + \text{VARIANCE OF } S_2]}$$

THEN  $\max_{\underline{w}} \{ J(\underline{w}) \}$  TO FIND OPTIMAL DIRECTION.

DEF: FISHER'S LINEAR DISCRIMINANT IS THE

LINEAR FUNCTION  $\underline{w}^T \underline{x}$  FOR WHICH THE

CRITERION FCN.:  $| \tilde{m}_1 - \tilde{m}_2 |^2$

$$J(\underline{w}) = \frac{| \tilde{m}_1 - \tilde{m}_2 |^2}{\tilde{S}_1^2 + \tilde{S}_2^2}$$

IS MAXIMIZED.

$J(\underline{w})$  CAN BE MAXIMIZED BY:

$$\nabla_{\underline{w}} J(\underline{w}) = \underline{0}.$$

$\Rightarrow$  IF  $\underline{\Sigma}_w$  IS NONSINGULAR:

$$\underline{w} = \underline{\Sigma}_w^{-1} [\underline{m}_1 - \underline{m}_2] \quad \leftarrow \text{F.L.D. SOLUTION}$$

NOTE: DIRECTION OF  $\underline{w}$  DEFINES THE NEW

(1D) FEATURE SPACE.

i.e. WE CAN NORMALIZE ~~to~~  $\underline{w}$  TO  $\|\underline{w}\|=1$   
FOR PURE PROJECTION.

NOTE: THERE ARE TECHNIQUES THAT EXTEND F.L.D  
TO RESULT IN ANY # OF DIMENSIONS  $D'$ ,  
S.T.  $1 \leq D' \leq D$ . (SOME OF THE APPLY  
F.L.D. ITERATIVELY.)

multiple discriminant analysis

GENERALIZE F.L.D. QUANTITIES:

$\underline{\Sigma}_i$  = SCATTER MATRIX FOR CLASS  $S_i$   
= SAME AS F.L.D.

$\underline{m}_i$  = MEAN OF CLASS  $S_i$   
 = SAME AS F.L.D.

$\underline{\underline{S}}_W$  = TOTAL WITHIN-CLASS SCATTER :

$$\underline{\underline{S}}_W = \sum_{i=1}^C \underline{\underline{S}}_i$$

$\underline{\underline{S}}_B$  = BETWEEN-CLASS SCATTER MATRIX:

$$\underline{\underline{S}}_B = \sum_{i=1}^C N_i (\underline{m}_i - \underline{m}) (\underline{m}_i - \underline{m})^T$$

$\underline{m}$  = MEAN OVER ALL DATA POINTS.

COMMENT: EACH MATRIX  $(\underline{m}_i - \underline{m})(\underline{m}_i - \underline{m})^T$   
 HAS RANK 1; AT MOST  $C-1$  OF THEM ARE  
 INDEPENDENT.

$$\therefore \text{rank } \{\underline{\underline{S}}_B\} \leq C-1.$$

"OPTIMAL" SUBSPACE DEFINED BY PROJECTION  
 MATRIX  $\underline{\underline{W}}$ :

$$\underline{\underline{W}} \triangleq \begin{bmatrix} ; & ; & ; & ; \\ \underline{w}_1 & \underline{w}_2 & \cdots & \underline{w}_{C-1} \end{bmatrix}$$

$\underline{w}_i$  ARE AXES OF NEW FEAT. SPACE.

THEN:

$$\underbrace{\underline{\underline{u}}_j}_{\text{typ. } (C-1)-\text{DIM.}} = \underline{\underline{W}}^T \underbrace{\underline{x}_j}_{D-\text{DIM.}}$$

## 4. Statistical Classification

### Basic conception

random vector, cross-correlation matrix, covariance, correlated/uncorrelated orthonormal transformation

Bayes Decision Theory

## Bayes Minimum-Error classification

naive bayes a basic assumption: each feature is independent

it's like a Maximum A Posteriori in decision making

### RELATION BETWEEN MAP AND NAIVE BAYES

MAP use data point to estimate parameter -->  $\theta$ ,  $\theta$  is the posterior and hypothesis

NAIVE BAYES is a classification method, so the final estimated is H, class, and know data distribution ( $\theta$ ), use theta as the input data information

BAYES MINIMUM-ERROR CLASSIFICATION  
( $C = 2$  CLASSES)

GOAL: CHOOSE DEC. BOUNDARY AND REGIONS

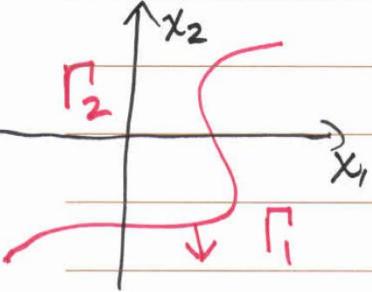
THAT MINIMIZE  $P(\text{ERROR})$ .

$$\begin{aligned} P(\text{ERROR}) &= P(\text{ERROR}, S_1) + P(\text{ERROR}, S_2) \\ &= \underbrace{P(\text{ERROR}|S_1)}_{\int_{\Gamma_2} p(x|S_1) dx} P(S_1) + \underbrace{P(\text{ERROR}|S_2)}_{\int_{\Gamma_1} p(x|S_2) dx} P(S_2) \end{aligned}$$

$$\therefore P(\text{ERROR}) = P_e = P(S_1) \int_{\Gamma_1}^{\Gamma_2} p(x|S_1) dx + P(S_2) \int_{\Gamma_1}^{\Gamma_2} p(x|S_2) dx$$

(1)

GOAL: FIND REGIONS  $\Gamma_1, \Gamma_2$  THAT MINIMIZE  $P_e$ .



KNOW:  $\Gamma_1$  AND  $\Gamma_2$  ARE NONOVERLAPPING.

$\Gamma_1 \cup \Gamma_2$  COVERS ALL IF  
FEATURE SPACE. (EXCEPT  
POSSIBLY BOUNDARIES THAT  
HAVE MEASURE 0.)

$\Rightarrow$  FOR EACH POINT  $x$ , ASSIGN  $x$  TO  
THE DEC. REGION THAT ~~GIVES THE~~ GIVES THE  
SMALLER TERM IN (1).

$$(2) \quad p(x|S_1)P(S_1) \stackrel{S_1}{<} p(x|S_2)P(S_2)$$

BAYES DEC. RULE FOR MIN. ERROR

OR:

$$\ln [p(x|S_1)P(S_1)] \geq \ln [p(x|S_2)P(S_2)]$$

(3)  ~~$\rightarrow$~~

$$P(S_1 | \underline{x}) \stackrel{S_1}{\geq} P(S_2 | \underline{x})$$

2

BAYES DEC. RULE FOR MIN.  $P_{\alpha_i}$

IN TERMS POSTERIOR PROBABILITIES

$\Rightarrow$  ASSIGN  $\underline{x}$  TO CATEGORY WITH LARGER POSTERIOR.

OTHER FORMS:

REARRANGE (2):  $\frac{p(\underline{x} | S_1)}{p(\underline{x} | S_2)} \stackrel{S_1}{\geq} \frac{P(S_2)}{P(S_1)}$

LIKELIHOOD RATIO:  $\ell(\underline{x}) \triangleq \frac{p(\underline{x} | S_1)}{p(\underline{x} | S_2)}$

THRESHOLD VALUE:  $T \triangleq \frac{P(S_2)}{P(S_1)}$

$$\Rightarrow \left[ \begin{array}{c} S_1 \\ \ell(\underline{x}) \stackrel{\geq}{\sim} T \\ S_2 \end{array} \right] \quad \downarrow$$

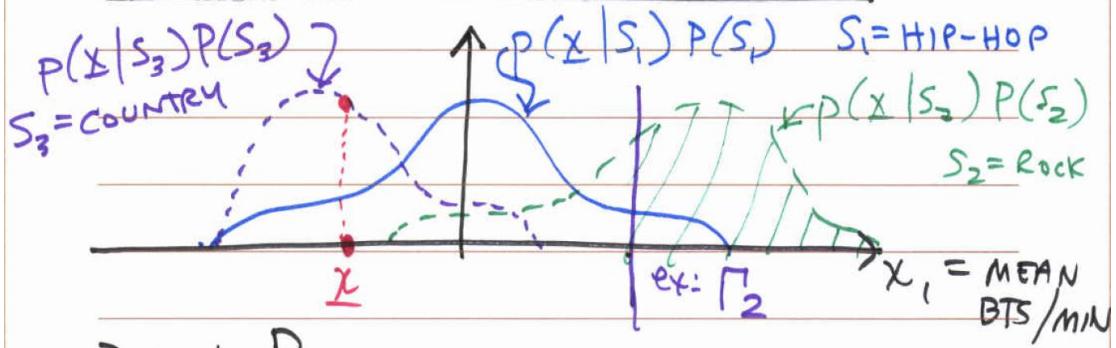
$$-\ln \ell(\underline{x}) \triangleq h(\underline{x}) \quad \begin{aligned} \ln [\ell(\underline{x})] &\geq \ln T \\ -\ln [\ell(\underline{x})] &\leq -\ln T \end{aligned}$$

$$h(\underline{x}) = \ln p(\underline{x} | S_2) \quad \begin{array}{c} \\ \downarrow \\ -\ln p(\underline{x} | S_1) \end{array}$$

$$\left[ \begin{array}{c} S_1 \\ S_2 \\ h(\underline{x}) \leq \ln \frac{P(S_1)}{P(S_2)} \triangleq \gamma \end{array} \right]$$

multiclass bayes

### BAYES MIN. ERROR : $C > 2$ CLASSES



$$P_e = 1 - P_{\text{correct}}$$

$$P_{\text{correct}} = \sum_{i=1}^C P(S_i) \int_{\Gamma_i}^{\infty} p(x|S_i) dx$$

min.  $P_e$  w.r.t. dec. boundary ( $\Gamma_i$  region)  
locations:

$$\Rightarrow p(x|S_i)P(S_i) > p(x|S_j)P(S_j) \quad \forall j \neq i$$

$$\Rightarrow x \in S_i$$

$\hookrightarrow$  BAYES MIN. ERROR DEC. RULE,  $C \geq 2$ ,  
THIS IS A MAXIMAL VALUE METHOD.

summary

CAN DEFINE:

$$g_i(\underline{x}) = p(x|S_i)P(S_i)$$

OR

$$g_i(\underline{x}) = \ln [p(x|S_i)P(S_i)]$$

## 1. BAYES MIN. ERROR CLASSIFIER - DEC. RULE

MULTICLASS:

$$p(\underline{x} | S_i) P(S_i) > p(\underline{x} | S_j) P(S_j) \quad \forall j \neq i$$

$$\Rightarrow \underline{x} \in S_i$$

2-CLASS:

$$p(\underline{x} | S_1) P(S_1) \geq p(\underline{x} | S_2) P(S_2)$$

## 2. PROBABILITY OF ERROR:

MULTICLASS:

$$P_e = 1 - \sum_{i=1}^c \int_{S_i} p(\underline{x} | S_i) P(S_i) d\underline{x}$$

2-CLASS:

$$P_e = \int_{S_2} p(\underline{x} | S_1) P(S_1) d\underline{x} + \int_{S_1} p(\underline{x} | S_2) P(S_2) d\underline{x}$$

finally, compared several distribution function, if  $\sigma_1 = \sigma_2$ , it's a linear classifier, else, quadratic

## Density Estimate

conditions in density estimate

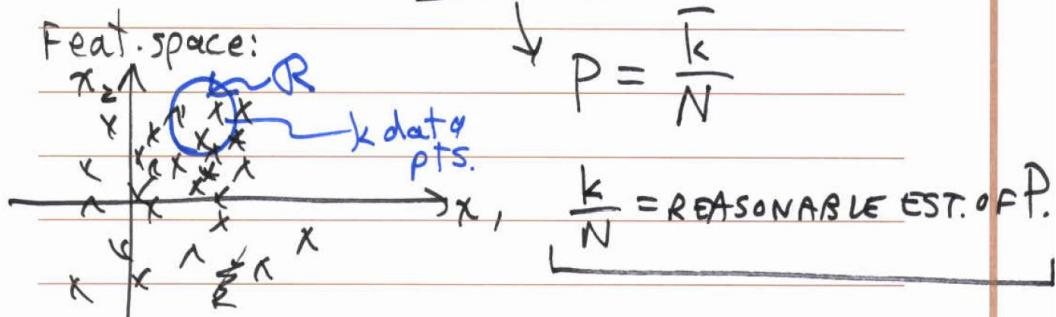
$p(\underline{x} | S_i), \quad P(S_i)$   
DON'T KNOW  $p'(\underline{x} | S_i),$   
MAYBE DON'T ~~NEVER~~ know  $P(S_i).$   
DON'T MAKE ANY ASSUMPTIONS ON  $p(\underline{x} | S_i)$

basic binary

GOAL: ESTIMATE  $p(\underline{x})$  GIVEN A SET OF  
DATA POINTS (DRAWN i.i.d. FROM  $p(\underline{x})$ )  
 $\underline{x}_1, \underline{x}_2, \dots, \underline{x}_N.$

$$P_k = \binom{N}{k} p^k (1-p)^{N-k}$$

MEAN:  $E\{k\} = NP = \bar{k}$



ASSUME  $R$  IS SMALL, HAS VOLUME  $V$ ,  
HAS  $\underline{x}$  AT ITS CENTER.

Also ASSUME  $p(\underline{x})$  IS CONTINUOUS AT  $\underline{x}$ .



$$P = \int_R p(\underline{x}') d\underline{x}' \quad \begin{aligned} &\text{approx. } p(\underline{x}') \approx p(\underline{x}) \\ &= \text{const.} \\ &\text{in } R. \end{aligned}$$

$$P \approx p(\underline{x}) V$$

$$p(\underline{x}) \approx \frac{P}{V} = \frac{\int_R p(\underline{x}') d\underline{x}'}{\int_R d\underline{x}'}$$

PLUG IN FOR P:

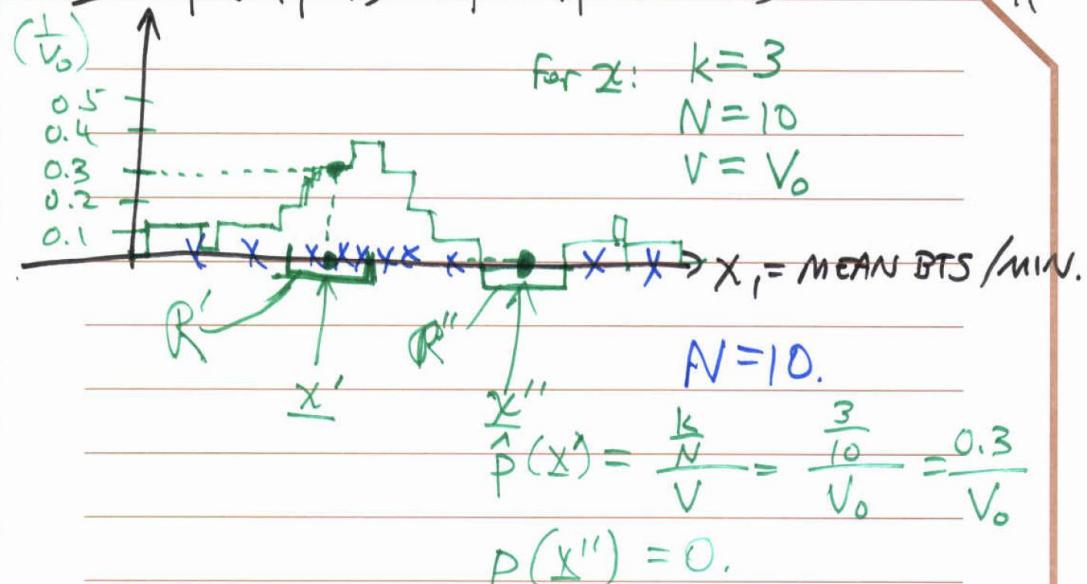
$$\hat{p}(\underline{x}) = \frac{k}{N} \frac{1}{V}$$

↑      ↗      ↓

# TRAINING DATA POINTS THAT LIE IN  $R$ .      TOTAL # TRAINING DATA POINTS      VOLUME OF  $R$ .

FUNCTION OF  $\underline{x}$ .

$$\text{Ex: } \hat{p}(x_1 | s_i) = \hat{p}(x_1 | \text{H1P-H0P})$$



2 methods to make density estimation

## 2 PRIMARY APPROACHES TO ESTIMATING $p(x)$

1. SPECIFY  $V_n = f_1(n)$

(PERHAPS ALSO SPECIFY SHAPE OR PROFILE OF  $R_n$ ).

e.g.:  $V_n = \frac{1}{\sqrt{n}}$

$\Rightarrow$  PARZEN WINDOWS (PW)

2. SPECIFY  $k_n = f_2(n)$

e.g.:  $k_n = \sqrt{n}$ .

$\Rightarrow$  k-NEAREST NEIGHBORS

BOTH CAN CONVERGE TO  $p(x)$  AS  $n \rightarrow \infty$   
IF  $V_n$  OR  $k_n$  IS CHOSEN APPROPRIATELY.

### 1. Parzen Windows

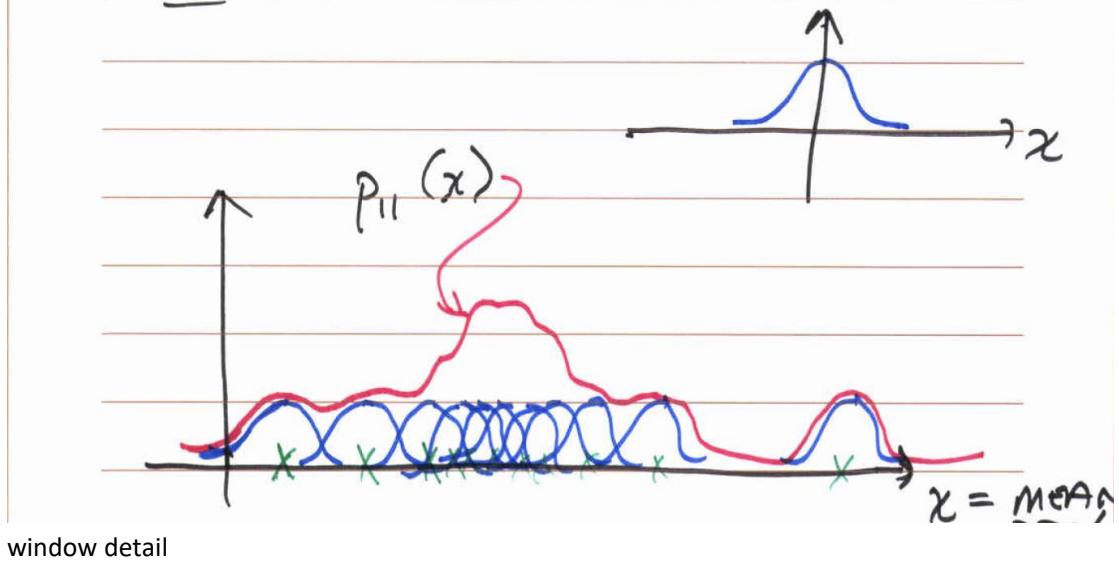
#### PARZEN WINDOWS ESTIMATION (PW)

1. CREATE A WINDOW FUNCTION  ~~$Ef(n)$~~ .

2. CENTER THE WINDOW FN. AT EACH DATA POINT.

3. TAKE SUM OF ALL WINDOW FUNCTIONS,  
DIVIDED BY  $n$ .

Ex : 1. GAUSSIAN WINDOW FCN.:



window detail

WINDOW FCN:  $\Delta(x)$

$\Delta(x - x_i)$  CENTERED AT  $x_i$ .

$$p_n(x) = \frac{1}{n} \sum_{i=1}^n \Delta(x - x_i) \quad \leftarrow \text{PW EST.}$$

FOR  $p_n(x)$  TO BE A DENSITY, REQUIRE:

(\*)  ~~$\int \Delta(u) du = 1$~~

$$\left\{ \begin{array}{l} \int \Delta(u) du = 1 \\ \Delta(u) \geq 0 \end{array} \right.$$

IF  $\Phi(x)$  IS AN UNNORMALIZED WINDOW FCN.,

WE CAN LET:

$$\Delta_n(x) = \frac{1}{V_n} \Phi\left(\frac{x}{h_n}\right)$$

WINDOW WIDTH  
PARAMETER.

CHOOSE  $V_n$  ~~( $\Phi$ )~~ TO ENSURE

$$\int \Delta_n(x) dx = 1.$$

## 2.K- Nearest Neighbors

## K-NEAREST NEIGHBORS (kNN) FOR DENSITY EST.

No width parameter ( $h_n$ ).

No choice of window profile.

Let window width vary according to local density of data points.

Center  $R_n$  at  $\underline{x}$ . Must it just large enough to capture  $k_n$  data points.

Specify  $k_n = f(n)$ .

Still:

$$p_n(\underline{x}) \approx \frac{\frac{k_n}{n}}{V_n} \quad \begin{matrix} \leftarrow \\ \text{SPECIFY THIS.} \end{matrix}$$

$\leftarrow$  CALCULATE FROM THE DATA.

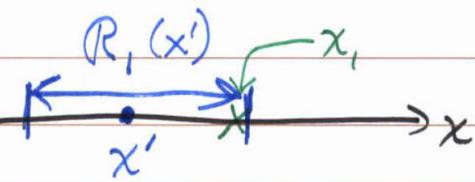
Ex: common choice:  $k_n = \sqrt{n}$ .

$$\Rightarrow p_n(\underline{x}) = \frac{\frac{\sqrt{n}}{n}}{V_n} = \frac{\frac{1}{\sqrt{n}}}{V_n}$$

Ex: Let  $n=1 \Rightarrow k_n = \sqrt{n} = 1$ .

$$p_1(\underline{x}) = \frac{1}{V_1}$$

1D case:



$$P_1(x') = ?$$

$$P_1(x') = \frac{1}{V_1}$$

$$V_1 = 2|x' - x_1|$$

so:

$$P_1(x') = \frac{1}{2|x' - x_1|}$$

classification based on density estimation

approach1: Generative Model

↳ find  $p(\underline{x}, S_i)$  from data  
(or  $p(\underline{x}|S_i)$  and  $P(S_i)$ )  
then use in classifier.

1. APPLY kNN or PW TECHNIQUE TO EST.

$p(\underline{x}|S_i)$  FOR EACH CLASS SEPARATELY  
(FROM DATA LABELED  $S_i$ ).

2. USE OR EST. PRIORES  $S_i$ , FROM DATA

$$(e.g.) \hat{P}(S_i) = \frac{n_i}{N}$$

3. USE IN A BAYES CLASSIFIER (MIN. ERROR)

OR MIN. RISK):

$$\hat{P}(\underline{x}|S_i)\hat{P}(S_i) > \hat{P}(\underline{x}|S_j)\hat{P}(S_j)$$

$$\forall j \neq i \Rightarrow \underline{x} \in S_i.$$

approach2: Discrimination Model

FOR THE CASE OF  $k$ -NN EST. OR  
 PW EST. USING A BINARY-VALUED  
 WINDOW FUNCTION ( $\boxed{\quad}$  INSTEAD OF  
 $\tilde{\quad}$ ), CAN DERIVE AN EXPRESSION IN  
 ADVANCE BASED ON EST. OF POSTERIOR  
 PROBABILITIES:

$$P(S_i | \underline{x}) > P(S_j | \underline{x})$$

$$\text{if } j \neq i \Rightarrow \underline{x} \in S_i.$$

AS FOLLOWS.

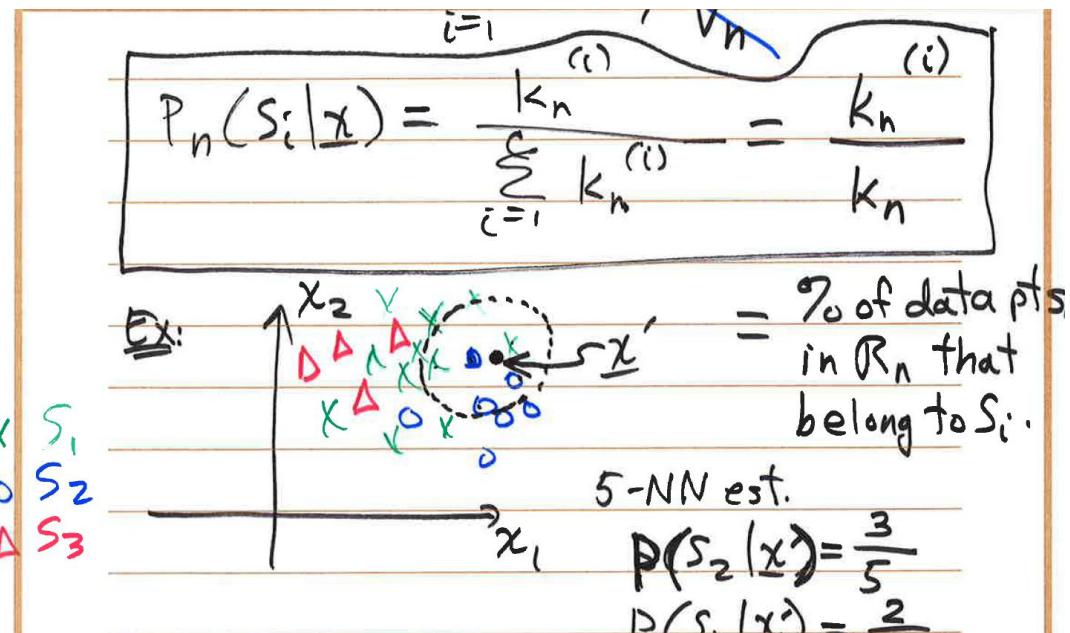
$$P_n(\underline{x}, S_i) = \frac{k_n^{(i)}}{n V_n}$$

$k_n^{(i)}$  = # DATA PTS. IN  $R_n$  LABELED  $S_i$ .

$n$  = TOTAL # DATA PTS. OVER ALL CLASSES.

$V_n$  = VOLUME OF  $R_n$

$k_n$  = TOTAL # DATA PTS. IN  $R_n$  (will  
 use later).



dimension in density estimate: -- curse of dimensionality

## Parameter Estimate

two basic assumption

### PARAMETER ESTIMATION

- INCLUDE ADDITIONAL CONSTRAINTS BY  
MAKING ASSUMPTIONS ON  $p(\underline{x} | S_i)$ .

e.g., ASSUME  $p(\underline{x} | S_i) = f(\underline{x}, \underline{\theta})$ :

- $f$  IS A KNOWN Fcn.
- $\underline{\theta}$  IS A VECTOR (SET) OF UNKNOWN PARAMETERS.

explanation:

### 2 VIEWS

1. PARAMETERS  $\underline{\theta}$  ARE DETERMINISTIC BUT UNKNOWN

2. PARAMETERS  $\underline{\theta}$  ARE RANDOM, AND THEMSELVES HAVE UNDERLYING pdf's.

IN BOTH CASES, TRAINING DATA IS DRAWN AT

Frist View:

View 1 -  $\theta$  is deterministic

LET  $\hat{\theta}$  BE THE ESTIMATE OF  $\theta$

LET  $x_1, x_2, \dots, x_n$  BE THE DATA, DRAWN  
I.I.D. FROM DENSITY  $p(x)$  [OR  $p(x|S_i)$ ].

$$\text{LET } \underline{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$

Also:  $S_i = \{x_1, x_2, \dots, x_n\}$  = DATA PTS.  
Labeled  $S_i$ .

$$\begin{aligned}\hat{\theta} &= G(x_1, x_2, \dots, x_n) \\ &= G(\underline{x}) \\ &= \hat{\theta}(\underline{x})\end{aligned}$$

$x_i$  ARE DRAWN AT RANDOM  $\Rightarrow \hat{\theta}$  IS RANDOM

use knowledge of unbias and consistant:  $\hat{\theta}$  is the estimate of  $\theta$

two method used in view 1:

Maximum likelihood Estimate

CRITERION FCN:  $J_{ML}(\theta) = p(\underline{x}|\theta)$

WANT:  $\max_{\theta} \{p(\underline{x}|\theta)\}$

YIELDS THE  $\hat{\theta}$  THAT MAXIMIZES THE

PROBABILITY OF OBTAINING THE DATA POINTS  
THAT WERE ACTUALLY OBSERVED.

To maximize  $J$ :

$$\nabla_{\underline{\theta}} J = \underline{0} \text{ solve algebraically.}$$

$$(1) \quad \begin{cases} \nabla_{\underline{\theta}} p(\underline{x} | \underline{\theta}) \Big|_{\underline{\theta} = \hat{\underline{\theta}}} = \underline{0} \\ \text{OR } \nabla_{\underline{\theta}} \ln p(\underline{x} | \underline{\theta}) \Big|_{\underline{\theta} = \hat{\underline{\theta}}} = \underline{0} \end{cases}$$

$$\text{Thus: } p(\underline{x} | \underline{\theta}) = \prod_{i=1}^n p(x_i | \theta)$$

$$\ln p(\underline{x} | \underline{\theta}) = \sum_{i=1}^n \ln p(x_i | \theta)$$

$$\nabla_{\underline{\theta}} \left[ \underbrace{\sum_{i=1}^n \ln p(x_i | \theta)} \right] \Big|_{\underline{\theta} = \hat{\underline{\theta}}} = \underline{0}.$$

known as  
 $f(\underline{\theta})$ .

$J(\underline{\theta})$  might not be concave (depends on  $f(\underline{\theta})$ ). If it isn't concave:

Maximum a posterior

$$ML: J_{ML}(\underline{\theta}) = p(\underline{z} | \underline{\theta})$$

$$MAP: J_{MAP}(\underline{\theta}) = p(\underline{\theta} | \underline{z})$$

= POSTERIOR DENSITY  
OF  $\underline{\theta}$ .

$$MAP: \hat{\underline{\theta}}_{MAP} = \underset{\underline{\theta}}{\operatorname{argmax}} p(\underline{\theta} | \underline{z})$$

$$\text{or } \hat{\underline{\theta}}_{MAP} = \underset{\underline{\theta}}{\operatorname{argmax}} \ln p(\underline{\theta} | \underline{z})$$

$$\text{or } \ln p(\underline{\theta} | \underline{z}) = \ln p(\underline{z} | \underline{\theta}) + \ln p(\underline{\theta})$$

~~$-\ln p(\underline{z})$~~   
const. of  $\underline{\theta}$

$$\hat{\underline{\theta}}_{MAP} = \underset{\underline{\theta}}{\operatorname{argmax}} \left[ \ln p(\underline{z} | \underline{\theta}) + \ln p(\underline{\theta}) \right]$$

✓      PRIOR  
INFO. ON  $\underline{\theta}$ .

view 2:

2. INTEGRATING OVER PARAMETER POSTERIOR DHS  
3.3

→ USE FULL PARAMETER POSTERIOR DENSITY

~~$p(\underline{z} | \underline{\theta})$~~   $p(\underline{\theta} | \underline{z})$  IN (2).

WE CAN WRITE:

$$p(x|\underline{z}) = \int p(x, \underline{\theta} | \underline{z}) d\underline{\theta}$$

$$p(x|\underline{\theta}, \underline{z}) p(\underline{\theta}|\underline{z})$$

(GIVEN  $\underline{\theta}$ ,  $\underline{z}$  PROVIDES NO  
( $\underline{z}$  IS INDEP. OF  $x$ ). USEFUL INFO.)

$$\Rightarrow p(x|\underline{z}) = \int p(x|\underline{\theta}) p(\underline{\theta}|\underline{z}) d\underline{\theta}$$

INSERTING CLASS INDEXET AND  $S_i$ :

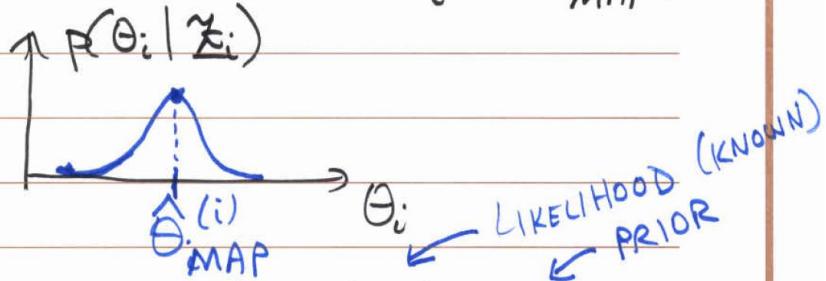
$$(3) p(x|S_i, \underline{z}_i) = \int p(x|\underline{\theta}_i, S_i) p(\underline{\theta}_i|\underline{z}_i) d\underline{\theta}_i$$

KNOWN.

$p(\underline{\theta}_i|\underline{z}_i)$  = PARAMETER POSTERIOR

→ CONTAINS OUR EST. OF  $\underline{\theta}_i$  (IN FORM OF ENTIRE DENSITY), BASED ON DATA  $\underline{z}_i$  AND OUR PRIOR KNOWLEDGE OF  $\underline{\theta}_i$ .

(3) TAKES WEIGHTED AVERAGE OVER ALL  $\underline{\theta}_i$ ,  
INSTEAD OF PLUGGING IN  $\underline{\theta}_i \approx \hat{\underline{\theta}}^{(i)}_{MAP}$ .



$$(4) p(\underline{\theta}_i | \underline{z}_i) = \frac{p(\underline{z}_i | \underline{\theta}_i) p(\underline{\theta}_i)}{\int p(\underline{z}_i | \underline{\theta}_i) p(\underline{\theta}_i) d\underline{\theta}_i}$$

NORMALIZING FACTOR

FOR BAYES MIN. ERROR CLASSIFIER:

IN (2), USE:

$$p(x | S_i) = p(x | S_i, \underline{z}_i) \text{ FROM (3) & (4).}$$

COMMENT: IF  $p(\underline{\theta}_i | \underline{z}_i) \approx \delta(\underline{\theta}_i - \hat{\underline{\theta}}^{(i)}_{MAP})$

THEN  ~~$\Rightarrow$~~  (3)  $\Rightarrow$

$$p(x | S_i, \underline{z}_i) \approx p(x | \hat{\underline{\theta}}^{(i)}_{MAP}, S_i)$$

$\Rightarrow$  SAME AS APPROACH 1.

## 5. Artificial Neural Network

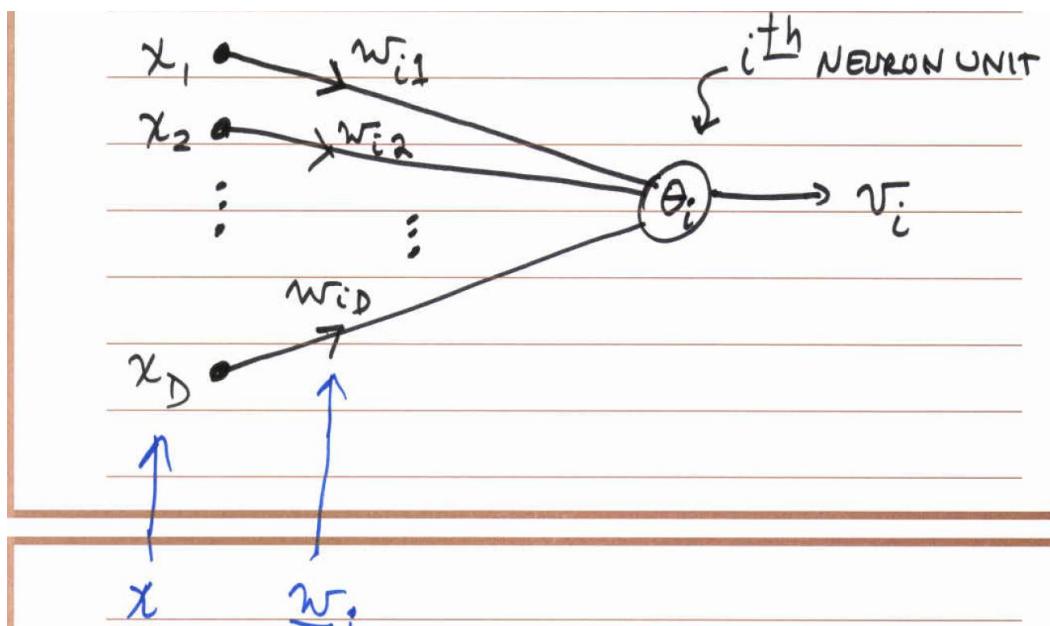
# ARTIFICIAL NEURAL NETWORKS (ANN)

[DHS Ch. 6 & ELSEWHERE]

## WHAT IS A.N.N.?

- SEQUENCE OF LINEAR COMBINATIONS OF INPUTS FOLLOWED BY THRESHOLDING
- NETWORK OF NEURON UNITS AND WEIGHTED INTERCONNECTIONS.

### basic elements

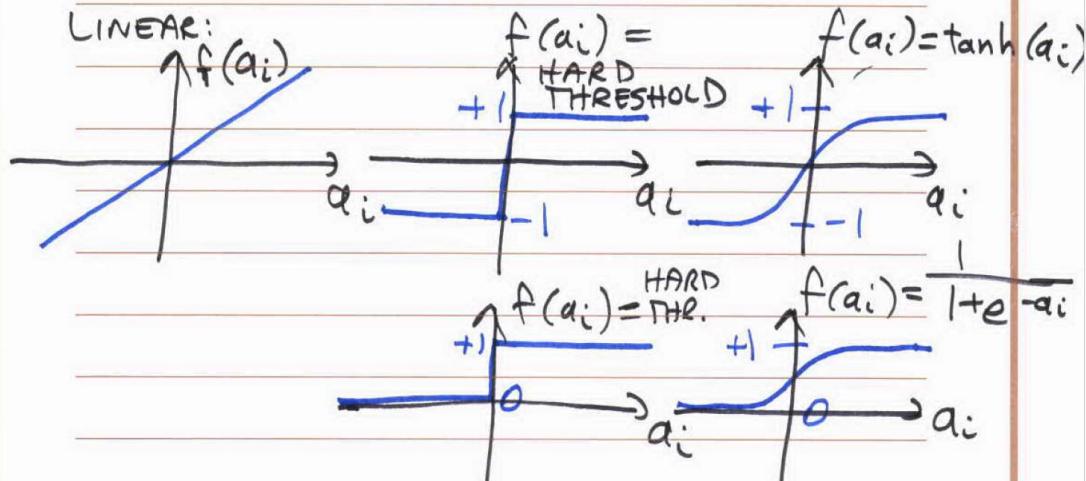


$$(1) v_i = f(w_i^T x - \theta_i) = f(a_i)$$

↑ ACTIVATION FUNCTION.      ↑ THRESHOLD      ↑ (MEMBRANE) POTENTIAL

### activation function

## COMMON ACTIVATION FUNCTIONS $f$



### Relu Function family

#### Noisy ReLUs [edit]

Rectified linear units can be extended to include Gaussian noise, making them noisy ReLUs, giving<sup>[7]</sup>

$$f(x) = \max(0, x + Y), \text{ with } Y \sim \mathcal{N}(0, \sigma(x)).$$

Noisy ReLUs have been used with some success in restricted Boltzmann machines for computer-vision tasks.<sup>[7]</sup>

#### Leaky ReLUs [edit]

Leaky ReLUs allow a small, positive gradient when the unit is not active.<sup>[9]</sup>

$$f(x) = \begin{cases} x & \text{if } x > 0, \\ 0.01x & \text{otherwise.} \end{cases}$$

#### Parametric ReLUs [edit]

Parametric ReLUs (PReLUs) take this idea further by making the coefficient of leakage into a parameter that is learned along with the other neural-network parameters.<sup>[12]</sup>

$$f(x) = \begin{cases} x & \text{if } x > 0, \\ ax & \text{otherwise.} \end{cases}$$

Note that for  $a \leq 1$ , this is equivalent to

$$f(x) = \max(x, ax)$$

and thus has a relation to "maxout" networks.<sup>[12]</sup>

#### ELUs [edit]

Exponential linear units try to make the mean activations closer to zero, which speeds up learning. It has been shown that ELUs can obtain higher classification accuracy than ReLUs.<sup>[13]</sup>

$$f(x) = \begin{cases} x & \text{if } x > 0, \\ a(e^x - 1) & \text{otherwise,} \end{cases}$$

where  $a$  is a hyper-parameter to be tuned, and  $a \geq 0$  is a constraint.

## neural network classify algorithm

### understanding of nerual network: single neuron

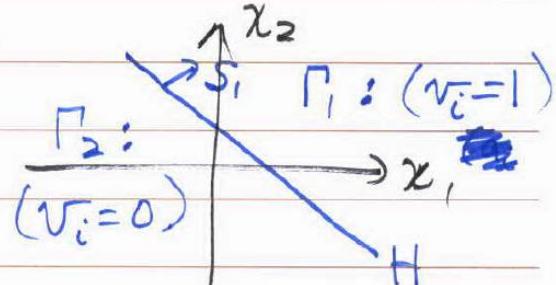
CONSIDER  $f = \text{HARD THRESHOLD}$ :

$$f(q_i) = \begin{cases} 1, & q_i > 0 \\ 0, & q_i \leq 0 \end{cases}$$

$$q_i = g_i(\underline{x}).$$

$\Rightarrow$  SINGLE NEURON UNIT (WITH  $\underline{x}$  INPUT)

IMPLEMENTES A 2-CLASS LINEAR  
CLASSIFIER.



algorithm: perceptron

### PERCEPTRON LEARNING

ORIG. PERCEPTRON RULE:

$$n = (i \bmod N) + 1$$

If  $\underline{x}_n$  is misclassified :

$$\underline{w}(i+1) = \underline{w}(i) + \gamma(i) \underline{z}_n \underline{x}_n$$

otherwise

$$\underline{w}(i+1) = \underline{w}(i)$$

Next i

perceptron in Neural Network  
based on the w update function

$$(2) \quad \underline{w}(i+1) = \underline{w}(i) + \gamma(i) \left[ \underline{v}_t^{(n)} - \hat{\underline{v}}^{(n)} \right] \underline{x}_n$$

$$\text{LET } \Delta \underline{w} \stackrel{\triangle}{=} \underline{w}(i+1) - \underline{w}(i)$$

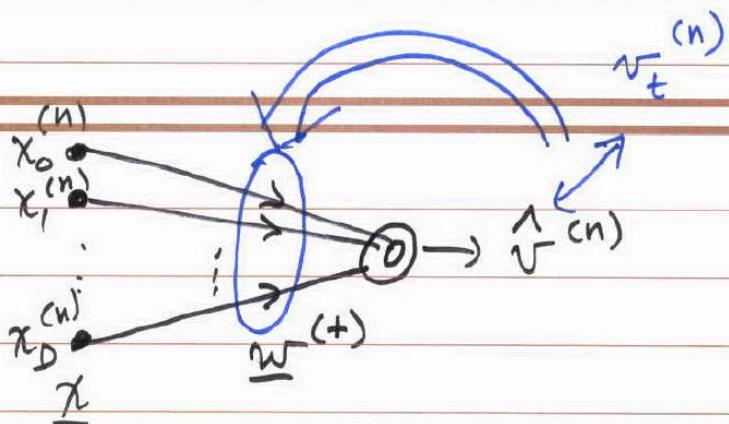
$$\Delta \underline{v}^{(n)} \stackrel{\triangle}{=} \underline{v}_t^{(n)} - \hat{\underline{v}}^{(n)}$$

(2)  $\Rightarrow$

$$(2') \quad \boxed{\Delta \underline{w} = \eta \underline{x}^{(n)} \Delta v^{(n)}} \quad \leftarrow \text{PERCEPTRON LEARNING, (ANN)}$$

Choose:

$$f(a) = \begin{cases} 1, & a \geq 0 \\ 0, & a < 0 \end{cases}$$



MORE GENERALLY

$$(3) \quad \boxed{\Delta \underline{w} = \eta \delta^{(n)} \underline{x}^{(n)}} \quad \begin{matrix} n = \text{DATA PT.} \\ \text{INDEX} \end{matrix}$$

OUTER PRODUCT LEARNING

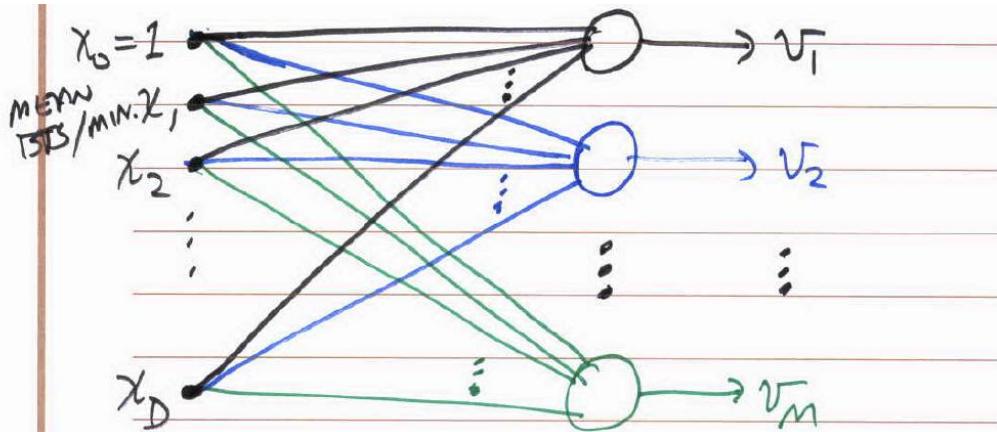
use  $\delta x$  to explain output error and as input for back prop

$\delta^{(n)}$  IS CHOSEN FOR THE PARTICULAR  
LEARNING ALGORITHM.

$$\text{Ex: } \delta^{(n)} = \Delta v^{(n)} = v_t^{(n)} - \hat{v}^{(n)}$$

$\Rightarrow$  PERCEPTRON  $f = \text{HARD-CLIPPING}$   
 $0, 1$  THRESHOLD.

multi-neuron units (single layer)



$$v_1 = f(a_1) = f(\underline{w}_1^{(+)\top} \underline{x}^{(+)})$$

$$v_2 = f(a_2) = f(\underline{w}_2^{(+)\top} \underline{x}^{(+)})$$

$\vdots$

$$v_M = f(a_M) = f(\underline{w}_M^{(+)\top} \underline{x}^{(+)})$$

For  $\star f = \text{HARD THRESHOLD} \Rightarrow$

$M$  2-CLASS CLASSIFIERS

OR SINGLE  $M$ -CLASS vs. Rest

( $\hat{v}_i = 1, \hat{v}_j = 0 \forall j \neq i \Rightarrow$  CLASSIFIER  $\underline{x} \in S_i$ ).

das

## EE660 Machine Learning

# 1. Introduction

## Basic Concept

Hypothesis set

1. HYPOTHESIS SET (models being considered)

E.g., 1D REGRESSION

Models:  $\hat{f}_1(x) = w_0 + w_1 x$

$$\hat{f}_2(x) = w_0 + w_1 x + w_2 x^2.$$

OR MORE GENERALLY:

$$\hat{f}_d(x) = \sum_{i=0}^d w_i x^i, \quad 1 \leq d \leq d_{\max}$$

CHOSES AMONG ALL  $d$  AND AMONG VALUES OF  $w$ .

$\Rightarrow$  OUR HYPOTHESIS SET.

objective function

2. OBJECTIVE FUNCTION (FCN. BEING OPTIMIZED).

Also: "CRITERION" FCN.

Ex:

$$J(w, \sigma) = \text{MSE}(\hat{y}, y_i)$$

$$= \frac{1}{N} \sum_{i=1}^N (\hat{y} - y_i)^2$$

$$\text{IN WHICH: } D = \left\{ \underline{x}_i, y_i \right\}_{i=1}^N$$

optimization method

complexity of hypothesis set, data,  
augmented space ( $\pi$  machine)

Non-augmented space

$$\underline{w} = \underline{w}^{(0)} = \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_D \end{bmatrix}$$

$$\underline{x} = \underline{x}^{(0)} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_D \end{bmatrix}$$

Augmented space

$$\underline{w} = \underline{w}^{(+)} = \begin{bmatrix} w_0 \\ w_1 \\ \vdots \\ w_D \end{bmatrix}$$

$$\underline{x} = \underline{x}^{(+)} = \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_D \end{bmatrix} = \begin{bmatrix} 1 \\ x_1 \\ \vdots \\ x_D \end{bmatrix}$$

$$\text{Linear } \hat{f}(\underline{x}) = w_0 + \underline{w}^T \underline{x}$$

$$\text{Linear } \hat{f}(\underline{x}) = \underline{w}^T \underline{x}$$

$\nwarrow$  (dropping superscripts)  $\nearrow$

## Bayes Estimation

WE WANT TO ESTIMATE  $\underline{\theta}$

INSTEAD OF FINDING A POINT ESTIMATE  $\hat{\underline{\theta}}$ ,

LET'S ESTIMATE THE DENSITY:

$$p(\underline{\theta} | \omega).$$

WE HAVE A MODEL:

(i)  $p(y | \underline{x}, \underline{\theta})$  [REGRESSION] [DISCRIMINATIVE]

(ii) PR  $p(\underline{x} | y, \underline{\theta})$  [CLASSIFICATION] [GENERATIVE]

USE BAYES' THEOREM:

$$(1) \quad p(\underline{\theta} | \omega) = \frac{p(\omega | \underline{\theta}) p(\underline{\theta})}{p(\omega)}$$

$$\text{IN WHICH: } p(\omega) = \int p(\omega | \underline{\theta}) p(\underline{\theta}) d\underline{\theta}$$

(OR SUM IF  $\underline{\theta}$  IS DISCRETE)

$$p(\underline{\theta} | \mathcal{D}) = \text{DENS. OF } w_j \text{ COEFF. OF } x_j$$

$$= \left[ \prod_{i=1}^N p(y_i | x_i, \underline{\theta}) \right] p(\underline{\theta}) / k$$

USE OUR MODEL, E.G.:  $N(y_i | w^\top x_i, \sigma^2)$ .

THEN GET POSTERIOR PREDICTIVE

$$p(y | \underline{x}, \mathcal{D})$$

From:  $p(y) = \int p(y | \underline{\theta}) p(\underline{\theta}) d\underline{\theta}$

$$(2) \quad p(y | \underline{x}, \mathcal{D}) = \int p(y | \underline{x}, \underline{\theta}) p(\underline{\theta} | \mathcal{D}) d\underline{\theta}$$

USUALLY:  $= \int p(y | \underline{x}, \underline{\theta}) p(\underline{\theta} | \mathcal{D}) d\underline{\theta}$

$p(w | \mathcal{D})$   
- GET FROM (1).

OUR MODEL  
(REGR,  
DISCRIMINATIVE)

OR CAN BE OBTAINED FROM  
 $p(x | y, \underline{\theta})$  (CLASS'N, GENERATIVE)

## 2. Algorithm: Regression

### Bayesian Regression

#### Linear Regression

MAP & ML

## REGRESSION USING MAXIMUM LIKELIHOOD ESTIMATE

$\underline{w} = \underline{w}^{(+)}$  THROUGHOUT

(MLE)

[M 7.3]

$p(\underline{\delta} | \underline{\theta})$  = LIKELIHOOD OF  $\underline{\theta}$ .

EST.  $\underline{\theta}$  USING MLE:

$$\hat{\underline{\theta}}_{MLE} = \underset{\underline{\theta}}{\operatorname{argmax}} \ln p(\underline{\delta} | \underline{\theta})$$

AND:  $\hat{\underline{w}}_{MLE} = \underset{\underline{w}}{\operatorname{argmax}} \ln p(\underline{\delta} | \underline{w})$

$$\sum_{i=1}^N \ln p(y_i | x_i, \underline{w})$$

ASSUMING POINTS IN  $\underline{\delta}$  ARE

ARE INDEPENDENTLY DISTRIBUTED  
(i.d.).

$$\begin{aligned}
 \text{Using } p(\underline{\alpha}|\theta) &= \prod_{i=1}^N p(y_i | \underline{x}_i, \theta) \\
 &= \prod_{i=1}^N N(y_i | \underline{w}^\top \underline{x}_i, \sigma^2) \\
 \Rightarrow J_1(\underline{w}) &= \underbrace{\frac{1}{2\sigma^2} \sum_{i=1}^N}_{\text{Drop}} (y_i - \underline{w}^\top \underline{x}_i)^2 \\
 &\quad + \underbrace{\frac{N}{2} \log(2\pi\sigma^2)}_{\text{const. of } \underline{w}}
 \end{aligned}$$

$$\begin{aligned}
 \text{Let } J(\underline{w}) &= \frac{1}{2} \text{ RSS}(\underline{w}) \\
 &= \frac{1}{2} \sum_{i=1}^N (y_i - \underline{w}^\top \underline{x}_i)^2 \\
 &= \frac{1}{2} \| \underline{y} - \underline{X} \underline{w} \|_2^2 \\
 &= \frac{1}{2} (\underline{y} - \underline{X} \underline{w})^\top (\underline{y} - \underline{X} \underline{w})
 \end{aligned}$$

$\nabla_{\underline{w}} J(\underline{w}, \underline{\theta}) = \underline{0}$   $\nabla$  SOLVE ALGEBRAICALLY

SOLVING GIVES  $\hat{\underline{w}}$ :

$$\underline{X}^T \underline{X} \hat{\underline{w}} = \underline{X}^T \underline{y}$$

IF  $(\underline{X}^T \underline{X})$  IS INVERTABLE, THEN

$$\hat{\underline{w}}_{OLS} = \underline{X}^{-1} \underline{y} = (\underline{X}^T \underline{X})^{-1} \underline{X}^T \underline{y}$$

conclusion of MLE and MAP

### SUMMARY OF ESTIMATION TECHNIQUES

$$\text{MLE: } \hat{\underline{\theta}}_{MLE} = \underset{\underline{\theta}}{\operatorname{argmax}} \left\{ \ln p(\underline{y} | \underline{\theta}) \right\}$$

$$\text{GAUSSIAN CASE} \Rightarrow J_{MLE}(\underline{w}, \underline{\theta}) = \text{MSE}$$

$$\text{MAP: } \hat{\underline{\theta}}_{MAP} = \underset{\underline{\theta}}{\operatorname{argmax}} \left\{ \ln p(\underline{y} | \underline{\theta}) + \ln p(\underline{\theta}) \right\}$$

$$\text{GAUSSIAN CASE} \Rightarrow J_{MAP}(\underline{w}, \underline{\theta}) = \frac{N \cdot (\text{MSE})}{N} + \lambda \|\underline{w}\|_2^2$$

Ridge Regression

Subs. (2) & (3)  $\rightarrow$  (1):  $\underbrace{1^{\text{st}} \text{ term}}$

$$\hat{\underline{w}} = \hat{\underline{\theta}} = \arg \max_{\underline{w}} \left\{ \sum_{i=1}^N \ln N(y_i | w_0 + \underline{w}^\top \underline{x}_i, \sigma^2) + \sum_{j=1}^D \ln N(w_j | 0, \tau^2) \right\}$$

$$\hat{\underline{w}} = \arg \max_{\underline{w}} \left\{ -J_R(\underline{w}, \sigma) \right\}$$

$\Rightarrow J_R$  IS OBJECTIVE Fcn. (IN)

$$J_R(w, \sigma) = \sum_{i=1}^N \left[ y_i - (w_0 + \underline{w}^\top \underline{x}_i) \right]^2 + \lambda \|\underline{w}\|_2^2$$

$\lambda = \frac{\sigma^2}{\tau^2}$  TRAINING SAMPLE OUTPUT  
 $J_{\text{MLE}}$   $\hat{f}(\underline{x}_i)$  new (prior term)  
 REGULARIZER

2 RIDGE REGRESSION OBJECTIVE Fcn.

ASSUMED: NORMAL DENSITIES  $p(y | \underline{x})$

$$p(\underline{w})$$

$\not\perp \text{ INDEP. OF } w_j$ .

POINTS IN  $\underline{x}$  ARE i.d.

### parameter estimate

could be similar as

### Bayesian Regression

1. MODEL IS  $p(y | \underline{x}, \underline{\theta}) = p(y | \underline{x}, \underline{w}, \underline{\theta}')$

Ex:  $= p(y | \underline{w}^\top \underline{x})$  (LINEAR CASE)  $\uparrow$   
 any other unknowns.

$= p(y | \underline{w}^\top \underline{\phi}(x))$

## 2. PARAMETER POSTERIOR

FROM EQ. (1):

$$\begin{aligned} p(\underline{\theta} | \underline{\omega}) &= p(\underline{w} | \underline{y}, \underline{X}) \xrightarrow{\text{Likelihood}} \\ &= \frac{p(\underline{y} | \underline{w}, \underline{X}) p(\underline{w} | \underline{X})}{K} \xrightarrow{\text{prior}} \\ K &= \int p(\underline{y} | \underline{w}, \underline{X}) p(\underline{w} | \underline{X}) d\underline{w} \end{aligned}$$

## 3. POSTERIOR PREDICTIVE

FROM EQ. (2):

$$\begin{aligned} p(y | \underline{x}, \underline{\omega}) &= \int p(y | \underline{x}, \underline{w}, \underline{\omega}) p(\underline{w} | \underline{x}, \underline{\omega}) d\underline{w} \\ &= \int p(y | \underline{x}, \underline{w}) p(\underline{w} | \underline{\omega}) d\underline{w}. \end{aligned}$$

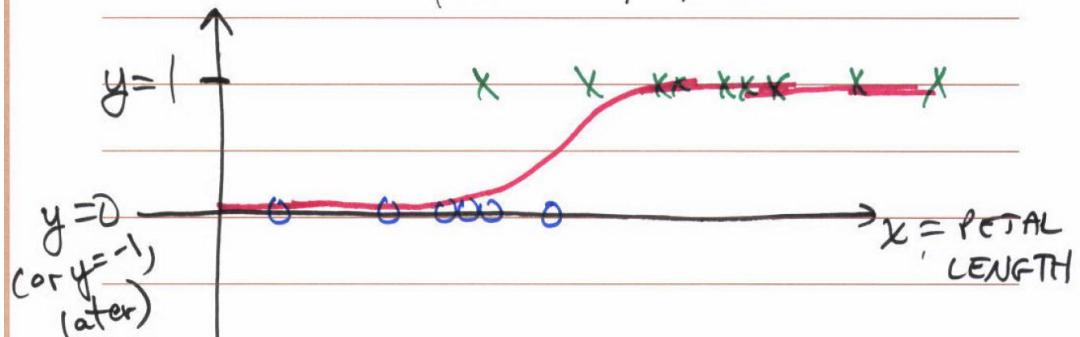
## Logistic Regression

basic function

## LOGISTIC REGRESSION

$$\hat{f}(x) = p(y=1 | x, \omega)$$

$$= p(\text{setosa} | x, \omega)$$



$$\hat{f}(x) = \text{sigm}\{\underline{w}^T \underline{x}\} \quad [\text{Murphy notation}]$$

$$= \frac{e^{\underline{w}^T \underline{x}}}{1 + e^{\underline{w}^T \underline{x}}}$$

$$= \Theta(\underline{w}^T \underline{x}) \quad [\text{AML notation}]$$

= "logistic" or "sigmoid" function  
(of  $\underline{w}^T \underline{x}$ )

model and simple form

MODEL

$$P(y | \underline{x}, \underline{w}) = \text{Ber}(y | \text{sigm}(\underline{w}^T \underline{x}))$$

$$= \mu^{\mathbb{I}(y=1)} (1-\mu)^{\mathbb{I}(y=0)} \quad [M 8.2]$$

in which  $\mu = \text{sigm}(\underline{w}^T \underline{x})$

CHANGE OUTPUT ( $y$ ) REPRESENTATION:

$$\text{LET } \tilde{y} = 2y - 1 \Rightarrow \tilde{y} \in \{-1, +1\}$$

$$p(\tilde{y} | \underline{x}, \underline{w}) = \mu^{\mathbb{I}(\tilde{y}=1)} (1-\mu)^{\mathbb{I}(\tilde{y}=-1)}$$

$$p(\tilde{y} | \underline{x}, \underline{w}) = [\text{sigm}(\tilde{y} \underline{w}^T \underline{x})]^{\mathbb{I}(\tilde{y}=1)} \cdot [\text{sigm}(-\tilde{y} \underline{w}^T \underline{x})]^{\mathbb{I}(\tilde{y}=-1)}$$

can show:  $\text{sigm}(-s) = 1 - \text{sigm}(s)$

$$p(y | \underline{x}, \underline{w}) = \text{sigm}(\tilde{y} \underline{w}^T \underline{x})$$

objective function for logistic regression

OBJECTIVE FCN.

MAX. LIKELIHOOD

$$\text{LIKELIHOOD} = p(\underline{w} | \underline{w})$$

$$= \prod_{i=1}^N p(\tilde{y}_i | \underline{x}_i, \underline{w})$$

$$= \prod_{i=1}^N \left( \frac{e^{\tilde{y}_i \underline{w}^T \underline{x}_i}}{1 + e^{\tilde{y}_i \underline{w}^T \underline{x}_i}} \right) \cdot \frac{e^{-1}}{e^{-1}}$$

$$= \prod_{i=1}^N \left( \frac{1}{e^{-\tilde{y}_i \underline{w}^T \underline{x}_i} + 1} \right)$$

$$-\ell(\underline{w}) = NLL(\underline{w}) = \sum_{i=1}^N \ln \left[ 1 + e^{-\hat{y}_i \underline{w}^\top \underline{x}} \right]$$

$$= J(\underline{w}, \underline{\theta})$$

$\uparrow$  OBJ. FCN. FOR MLE OF  $\underline{w}$  IN LOGISTIC REGRESSION.  
 $J$  IS CONVEX.

LET:  $-\ell(\underline{w}) = \sum_{i=1}^N E_i, \quad E_i = \ln \left[ 1 + e^{-\hat{y}_i \underline{w}^\top \underline{x}} \right]$

minimize lost(objective function)

CAN SHOW:

$$\nabla_{\underline{w}} E_i = \frac{\hat{y}_i \underline{x}_i}{1 + e^{\hat{y}_i \underline{w}^\top \underline{x}_i}}$$

> STOCHASTIC GRADIENT DESCENT

- (i) INITIALIZING  $\underline{w}(0)$
- (ii) ITERATE OVER DATA POINTS UNTIL CONVERGENCE:

(a) RANDOMLY PICK A DATA POINT  
 (WITH REPLACEMENT)

(b) PERFORM A SINGLE-SAMPLE

UPDATE:

$$\underline{w}(i+1) = \underline{w}(i) - \eta(i) \nabla_{\underline{w}} E_i$$

( $i$  = ITERATION NUMBER)

regularization

$$\rightarrow J(\underline{w}, \sigma) = \underbrace{\text{NLL}(\underline{w})}_{\substack{\text{from} \\ \text{logistic} \\ \text{regr., MLE.}}} + \lambda \|\underline{w}\|_2^2$$

(IF  $p(\underline{w}) = \text{GAUSSIAN}$   
 $= N(\underline{w} | 0, \sigma^2)$ )

### 3. Complexity

feasibility of learning (theory of hypothesis set)

concept of feasibility

#### FEASIBILITY OF LEARNING (part 1)

IDEAS  
 WHAT CAN MAKE A ML PROBLEM FEASIBLE?

- IF TRAINING DATA IS LINEARLY SEPARABLE
- IF WE CAN CONSTRUCT A SUITABLE OBJECTIVE FCN.
- IF WE MAKE ASSUMPTION(S) ON THE PROBLEM OR DATA.

e.g.: # BLACK SQUARES IS SIGNIFICANT  
 OR,  $\exists$  SOME UNDERLYING CONCEPT THAT

to constrain feasibility of learning: choose hypothesis set

general error

(all of hypothesis concern about in sample error and out sample error)

LET  $E_{in}(h) = \text{IN-SAMPLE ERROR}$   
 (ERROR ON TRAINING DATA  $D_{Tr}$ )

$$= \frac{1}{N} \sum_{n=1}^N \mathbb{I}[h(\underline{x}_n) \neq f(\underline{x}_n)]$$

$\uparrow$  hypoth.  
 $\uparrow$  TRUE TARGET  
 (e.g.,  $\hat{f}(\underline{x}_n) = y_n$   
 or  $\hat{g}(\underline{x}_n)$ )

$E_{\text{out}}(h)$  = OUT-OF-SAMPLE ERROR

(ERROR ON UNKNOWNS)

$$= P[h(\underline{x}) \neq f(\underline{x})]$$

$E_{\text{test}}(h)$  = ERROR ON TEST SET  $\mathcal{D}_{\text{Test}}$

$$= \frac{1}{N_{\text{Test}}} \sum_{\underline{x}_i \in \mathcal{D}_{\text{Test}}} [h(\underline{x}_i) \neq f(\underline{x}_i)]$$

WE CAN MEASURE  $E_{\text{in}}$  AND  $E_{\text{Test}}$ ; WE WANT  
TO KNOW  $E_{\text{out}}$ .

Hoeffding inequality

$$\text{EST. } \hat{\mu} = \frac{\# \text{RED MARBLES}}{N} \triangleq \nu.$$

How accurate is our estimate?

HOEFFDING INEQUALITY:

$$P[|\nu - \mu| > \epsilon] \leq 2e^{-2\epsilon^2 N}$$

FOR ANY  $\epsilon > 0$

$\epsilon$  IS OUR "TOLERANCE"

$\nu$  IS THE ONLY RANDOM QUANTITY.

HOEFFDING INEQ.:

$$P[|u-v| > \epsilon] \leq 2e^{-2\epsilon^2 N} \quad \text{FOR ANY } \epsilon > 0.$$



$$P[|E_{out}(h) - E_{in}(h)| > \epsilon] \leq 2e^{-2\epsilon^2 N} \quad \text{FOR ANY } \epsilon > 0$$



NOTE: DATASET  $\mathcal{D}$  (TRAINING, TEST, ETC.), DRAWN  
AT RANDOM ACCORDING TO  $p(x)$ .

here we get the bound of model in hypothesis set (we assume the ideal best model that can predict the dataset perfectly) that show different between our output model and real word model

Training procedure

PROCEDURE FOR HOEFFDING INEQ. TO BE VALID:

1. SPECIFY  $h$ . ( $\Rightarrow$  WITH  $p(x)$ , DETERMINES THE BIN OF MARBLES)
2. DRAW  $\mathcal{D}$
3. GET BOUNDS ON  $E_{out}(h)$ , GIVEN  $E_{in}(h)$

ML PARADIGM  $\Rightarrow$

1. COLLECT DATASET  $\mathcal{D}$
2. CONSTRUCT HYPOTHESIS SET  $\mathcal{H} = \{h_m\}_{m=1}^M$
3. TRAIN TO FIND  $h_g$  (best hypothesis).

4. CALCULATE  $E_{in}(h_g)$  OR  $E_{Test}(h_g)$ .
5. WOULD REALLY WANT TO KNOW:  $E_{out}(h_g)$ .

here change  $h$  to  $h_g$  because we specified a  $h_g$  that is one hypothesis in hypothesis set, a detailed one, that may not perfect, but is the best model we can get, can we want to find bound between it and

$$\begin{aligned} & P[|E_{\text{in}}(h_g) - E_{\text{out}}(h_g)| \geq \epsilon] \\ & \leq P\left[\bigcup_{m=1}^M (|E_{\text{in}}(h_m) - E_{\text{out}}(h_m)| \geq \epsilon)\right] \end{aligned}$$

$$\begin{aligned} & \leq \sum_{m=1}^M P[|E_{\text{in}}(h_m) - E_{\text{out}}(h_m)| \geq \epsilon] \\ & \leq 2e^{-2\epsilon^2 N} \\ & \leq 2Me^{-2\epsilon^2 N}. \end{aligned}$$

(2). i.  $P[|E_{\text{in}}(h_g) - E_{\text{out}}(h_g)| \geq \epsilon] \leq 2Me^{-2\epsilon^2 N}$

$M = |\mathcal{H}| = \# \text{ OF HYPOTHESES IN } \mathcal{H}$ .

→ A LOOSE UPPER BOUND.

2 aspects of learning feasibility

## 2 ASPECTS ~~OF~~ OF LEARNING FEASIBILITY

FEASIBILITY OF  
LEARNING:  
can ~~we~~ we get  
 $E_{\text{out}}(h_g)$  small enough?

Can ~~we~~ we get  
 $|E_{\text{in}}(h_g) - E_{\text{out}}(h_g)|$   
small enough?

Can we make  
 $E_{\text{in}}(h_g)$  small  
enough?

### Generalization Error

GENERALIZATION ERROR [AMC 2.1]

HOEFFDING INEQ:

$$P[|E_{\text{in}}(h_g) - E_{\text{out}}(h_g)| > \epsilon] \leq 2M e^{-2\epsilon^2 N}$$

$H > 0$

RE-ARRANGE:

$$P[|E_{\text{out}}(h_g) - E_{\text{in}}(h_g)| \leq \epsilon] > 1 - \delta$$

$\boxed{B} \quad \delta = 2M e^{-2\epsilon^2 N}$

$$\Rightarrow \epsilon = \sqrt{\frac{1}{2N} \ln \frac{2M}{\delta}}$$

~~B > 0:  $E_{\text{out}} \leq E_{\text{in}} + \epsilon$~~

~~B < 0:  $E_{\text{out}} \geq E_{\text{in}} - \epsilon$~~

(2.1)  $P[E_{\text{out}}(h_g) \leq E_{\text{in}}(h_g) + \sqrt{\frac{1}{2N} \ln \frac{2M}{\delta}}] > 1 - \delta$

(2.1')  $P[E_{\text{out}}(h_g) \geq E_{\text{in}}(h_g) - \sqrt{\frac{1}{2N} \ln \frac{2M}{\delta}}] > 1 - \delta$

$\delta$  IS OUR TOLERANCE.

(2.1) GIVES UPPER BOUND ON  $E_{\text{out}}(h_g)$   
IN TERMS OF OUR TOLERANCE  $\delta$  AND  $N, M$ .

In this measurement,  $M \rightarrow \#$  of hypothesis set is based on parameters, and parameters has infinite choices, so the  $M = \text{infinite}$ , makes the bound useless (bound = infinite)  
need a better method to measure:

### Dichotomies

Def: THE SET OF DICHOTOMIES GENERATED BY  $\mathcal{H}$   
ON  ~~$\mathcal{X}$~~   $\{\underline{x}_n\}_{n=1}^N$ :

$$\mathcal{H}(\underline{x}_1, \underline{x}_2, \dots, \underline{x}_N)$$

$$= \left\{ (h(\underline{x}_1), h(\underline{x}_2), \dots, h(\underline{x}_N)) \mid h \in \mathcal{H} \right\}.$$

(DUPLICATE N-TUPLES COUNT AS SAME MEMBER OF  $\{\cdot\}$ ).

$$|\mathcal{H}(\underline{x}_1, \dots, \underline{x}_N)| = \text{CARDINALITY OF } \mathcal{H}(\underline{x}_1, \dots, \underline{x}_N) \\ = \# \text{ OF DICHOTOMIES.}$$

GIVEN  $N$  pts. — HOW MANY DICHOTOMIES ARE POSSIBLE?  $2^N$ .

grow function

Def: GROWTH FUNCTION FOR  $\mathcal{H}$  IS:

$$m_{\mathcal{H}}(N) = \max_{\underline{x}_1, \dots, \underline{x}_N \in \mathcal{X}} |\mathcal{H}(\underline{x}_1, \dots, \underline{x}_N)|$$

I.E., FIND THE ~~SET~~ SET OF  $N$  POINTS THAT MAXIMIZES THE # DICHOTOMIES THAT  $\mathcal{H}$  CAN REALIZE.  $m_{\mathcal{H}}(N) =$  THIS # DICHOTOMIES.

$$m_{\mathcal{H}}(N) \leq 2^N \text{ ALWAYS.}$$

breaking point

## BREAK POINT K

IF THERE IS NO SET OF  $k$  (DISTINCT) POINTS THAT CAN BE SHATTERED BY  $\mathcal{H}$ , THEN  $k$  IS A BREAK POINT FOR  $\mathcal{H}$ , AND  $m_{\mathcal{H}}(k) < 2^k$ .

$\mathcal{H}_L$  BREAK POINTS ARE: AND  $k \geq 3$ ,

IF  $k$  IS A BREAK POINT FOR  $\mathcal{H}$ , THEN:

$$m_{\mathcal{H}}(N) \leq \sum_{i=0}^{k-1} \binom{N}{i} \quad \forall N$$

$$\text{NOTE: } \binom{N}{i} = \frac{N!}{i!(N-i)!} = O(N^{N-(N-i)}) \\ = O(N^i).$$

POLYNOMIAL IN  $N$ , OF DEGREE  $k-1$ .

## VC Dimension

(all about discussion in sample error and out sample error)

## VC DIMENSION

(VAPNIK-CHERVENENKIS)

VC DIM. IS A MEASURE OF THE "FLEXIBILITY"  
OR "COMPLEXITY" OF THE HYPOTHESIS SET.

DEF: THE VC DIMENSION OF  $\mathcal{H}$ ,  $d_{vc}(\mathcal{H})$ ,  
IS THE LARGEST VALUE OF  $N$  FOR WHICH  
 $m_{\mathcal{H}}(N) = 2^N$ . IF  $m_{\mathcal{H}}(N) = 2^N \forall N$ , THEN  
 $d_{vc}(\mathcal{H}) = \infty$ .

1. GIVEN  $d_{vc}(\mathcal{H})$ , THEN  $k = d_{vc} + 1$

(OR ANY  $k \geq d_{vc} + 1$ ) IS A BREAK POINT  
OF  $\mathcal{H}$ .

2.  $d_{vc}(\mathcal{H}) = \max N$  THAT  $\mathcal{H}$  CAN SHATTER.

use VC-dimension to calculate error bound

if we know VC-dimension  $\rightarrow$  can estimate # of hypothesis set based on # of data

$m_{\mathcal{H}}(N) \leq \text{polyn. in } N \text{ OF DEGREE } d_{vc}$ .

ONE CAN SHOW:

$$m_{\mathcal{H}}(N) \leq N^{d_{vc}} + 1$$

Theorem 2.5: VC GENERALIZATION BOUND

FOR ANY TOLERANCE  $\delta > 0$ ,

$$E_{out}(h_g) \leq E_{in}(h_g) + \sqrt{\frac{8}{N} \ln \frac{4m_{\mathcal{H}}(2N)}{\delta}}$$

WITH PROBABILITY  $\geq 1 - \delta$ .

$$\epsilon_{eff}$$

VC generalization and understanding

error for test set (because hypothesis set is only the test set, one)

## 2. USING A TEST SET TO BOUND $E_{\text{OUT}}(h_g)$ :

IF  $N_{\text{Test}}$  HAS NOT BEEN USED TO PICK  $h_g$  OR  
IF, THEN:

$$E_{\text{OUT}}(h_g) \leq E_{\text{Test}}(h_g) + \sqrt{\frac{1}{2N} \ln \frac{2M}{\delta}}$$

WITH PROBABILITY  $1 - \delta$ .

$\rightarrow$  USE  $M = 1$ .

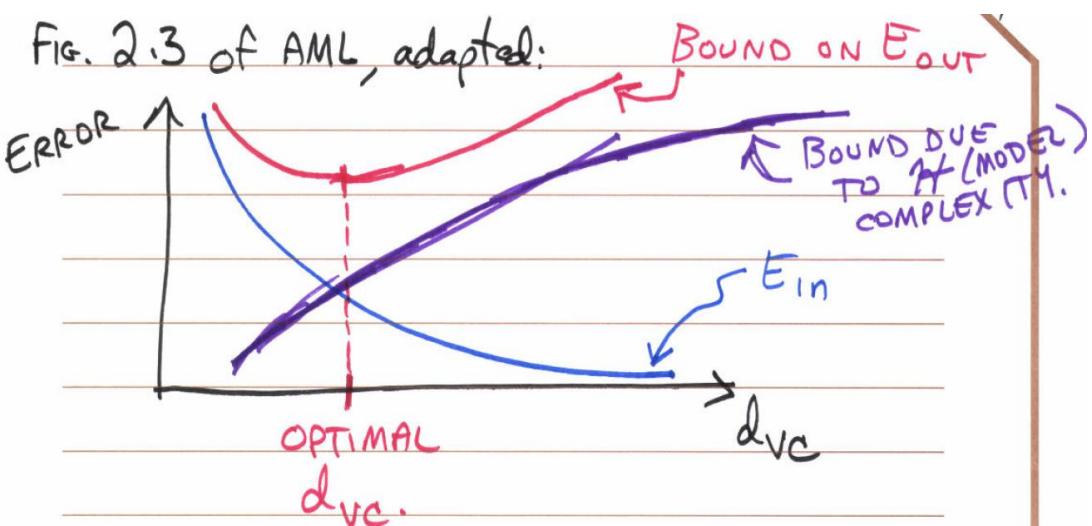
error for training set (hypothesis set is large(all of hypothesis))

## 3. Thm 2.5 AGAIN: (using $\epsilon_{VC}$ )

$$E_{\text{out}}(h_g) \leq E_{\text{in}}(h_g) + \sqrt{\frac{8}{N} \ln \frac{4[(2N)^{\epsilon_{VC}} + 1]}{\delta}}$$

↓ ↑      ↑ ↓  
 $\epsilon_{VC}$       typ. ↑      typ. ↓

BOUND DUE TO  
 MODEL COMPLEXITY  
 (FOR A GIVEN N, S).



## Bias vs Variance

concept: error measurement

$$\text{WE HAD } E_{in}(h) = \frac{1}{N} \sum_{n=1}^N [h(x_n) \neq f(x_n)]$$

= IN-SAMPLE ERROR RATE.

$$E_{out}(h) = P[h(x) \neq f(x)]$$

CAN INSTEAD USE:

$$E_{in}(h) = \frac{1}{N} \sum_{n=1}^N [h(x_n) - f(x_n)]^2$$

$$E_{out}(h) = \mathbb{E}_{\underline{x}} \{ [h(\underline{x}) - f(\underline{x})]^2 \}$$

(E IS w.r.t. p( $\underline{x}$ )).

Bias vs. Variance: it's a trade off when considering model complexity

(different bias and variance when model complexity is different)

NOTE: Below,  $h_g$  is BEST HYPOTHESIS AS  
CHOSEN BY  $\hat{\theta}$  (USING LEARNING ALGORITHM  
 $A.$ )

$$E_{out}(h_g^{(\hat{\theta})}) = \mathbb{E}_{\underline{x}} \left\{ [h_g^{(\hat{\theta})}(\underline{x}) - f(\underline{x})]^2 \right\}$$

↑ target fn

TAKE  $E_D$  OF BOTH SIDES

w.r.t. all datasets  $D$  of size  $N$ .

$$E_D \{ E_{out} (h_g^{(D)}) \} = E_D \{ \text{RHS} \}$$

DEFINE:  $\bar{h}_g(x) = E_D \{ h_g^{(D)}(x) \}$

$$\approx \frac{1}{K} \sum_{k=1}^K \underbrace{h_{gk}(x)}$$

BEST HYPOTH.

BASED ON

$D$ .

$E(D)\{E_{out}(h_g)\}$  means built many  $h_g$ (best model based on current dataset) and find average of these models to get the final best model(based on different dataset)

$$\begin{aligned} & E_D \{ E_{out} (h_g^{(D)}) \} \\ &= E_x \left\{ E_D \left\{ [h_g^{(D)}(x) - \bar{h}_g(x)]^2 \right\} \right\} \\ &\quad \underbrace{\text{var}(x)}_{\text{variance}} + \underbrace{[\bar{h}_g(x) - f(x)]^2}_{\text{bias}(x)} \\ &\quad \text{(sometimes called bias}^2(x)) \\ &= E_x \{ \text{var}(x) \} + E_x \{ \text{bias}(x) \} \end{aligned}$$

bias and variance:

bias: means difference between predict result and target result

variance: means change between different models (like for a specific predict result in different model based on different dataset)

$$E_D \{ E_{out} (h_g^{(D)}) \} = \text{bias} + \text{var.}$$

## learning curve

Y BIAS-VAR. VIEWPOINT -

$$\mathbb{E}_{\mathcal{D}} \{ E_{\text{out}}(h_g^{(\mathcal{D})}) \} = \mathbb{E}_x \{ \text{bias}(x) + \text{var}(x) \}$$

$$= \text{bias} + \text{var}$$

$$\text{bias}(x) = [\bar{h}_g(x) - f(x)]^2$$

$$\text{var}(x) = \mathbb{E}_{\mathcal{D}} \{ [\bar{h}_g^{(\mathcal{D})}(x) - \bar{h}_g(x)]^2 \}$$

Y VC VIEWPOINT -

$$E_{\text{out}}(h_g^{(\mathcal{D})}) \leq E_{\text{in}}(h_g^{(\mathcal{D})}) + \epsilon(N, H, \delta)$$

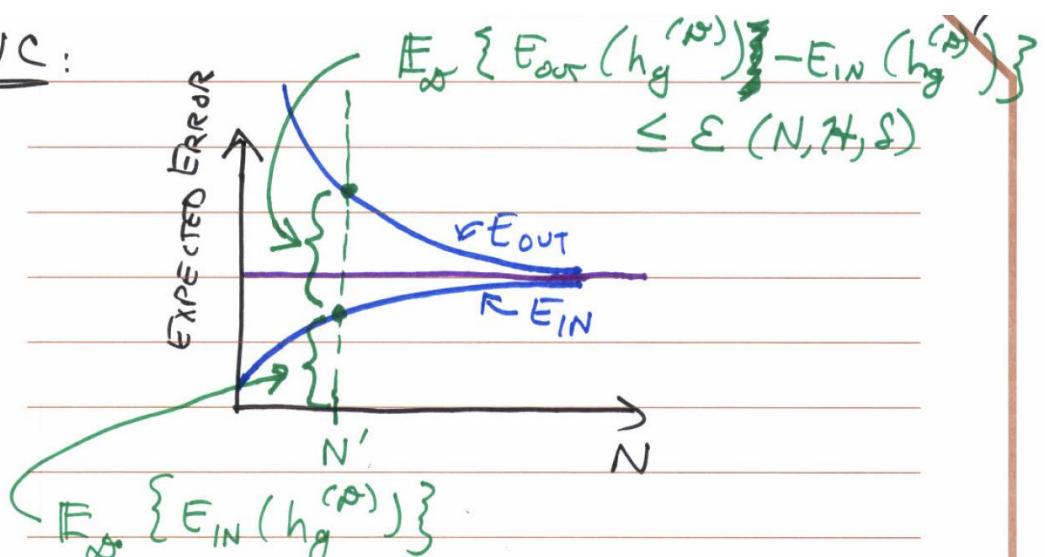
WITH PROBABILITY  $\geq 1 - \delta$ .

$$\mathbb{E}_{\mathcal{D}} \{ E_{\text{out}}(h_g^{(\mathcal{D})}) \} \leq \mathbb{E}_{\mathcal{D}} \{ E_{\text{in}}(h_g^{(\mathcal{D})}) \}$$

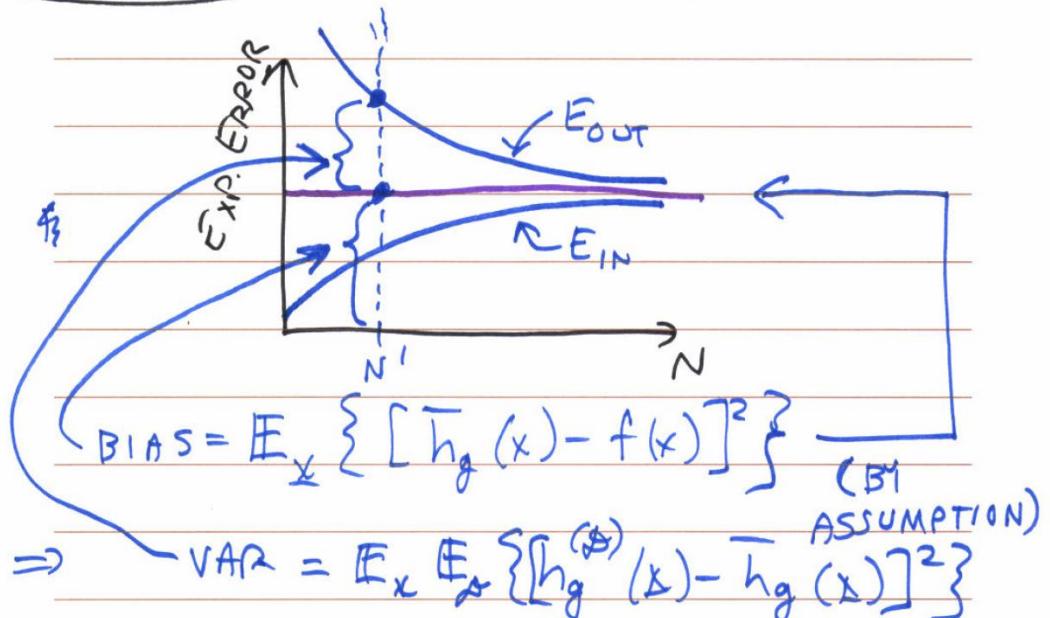
$$+ \epsilon(N, H, \delta)$$

$$\epsilon(N, H, \delta) \leq \sqrt{\frac{8}{N} \ln \frac{4[(2N)^{d_{VC}} + 1]}{\delta}}$$

VC:



## BIAIS-VAR



Id

## 4. Fundation of Model and Training procedure

### Whole Training Method

method1:

1. COLLECT DATA AND CONSTRUCT  $\mathcal{D}$ .

1.5 DIVIDE  $\mathcal{D}$   $\xrightarrow{\mathcal{D}}$   $\mathcal{D}_{\text{Test}}$

SET  $\mathcal{D}_{\text{Test}}$  ASIDE (NO SNOOPING!)

2. DO PRELIMINARY DATA ANALYSIS.

- VISUALIZE (OR LOOK AT) DATA,
- PLOT SOME HISTOGRAMS, LEARN FROM THEM..

### 3. PREPROCESSING

~~CHARACTERIZE~~

~~NORMALIZE~~ NORMALIZE

:

### 4. FEATURE EXTRACTION.

- ASSESS WHETHER EXTRACTED  
FEATURES ARE USEFUL FOR  
CLASSIFICATION.

### 5. DO SOME PRELIMINARY TRIALS ON $\mathcal{D}$ .

- VARYING DIMENSION (#FEATURES)  
- TRY A FEW HYPOTHESIS SETS AND  
LEARNING ALGORITHMS.

### 6. SET UP AND RUN MODEL SELECTION AND LEARNING ALGORITHMS.

- DECIDE ON  $\mathcal{H}$

- FOR CROSS-VALIDATION ON  $\mathcal{D}'$  TO  
CHOOSE PARAMETERS, TRAIN, AND FIND  $h_g$ ,  
(USING  $\mathcal{D}' \setminus \mathcal{D}_{Tr}$ )  
 $\setminus \mathcal{D}_{Val}$

7. BEFORE COMPUTING  $E_{\text{Test}}(h_g)$ , APPLY SAME PREPROC., FEAT. EXTR., AS WAS DONE FOR FINAL SYSTEM ON  $\mathcal{D}'$ . (USING ONLY INFORMATION FROM  $\mathcal{D}'$ .)

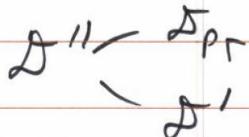
possible pitfall

POSSIBLE PITFALL:  $(E_{\text{Val}} \text{ or } E_{\text{Tr}})$   
 $\mathcal{D}_{\text{Val}}$  (or  $\mathcal{D}_{\text{Tr}}$ ) MIGHT NOT GENERALIZE TO  $E_{\text{out}}$ .  
- INCREASE  $N$  IF POSSIBLE  
- MORE (CORRECTLY DONE) CROSS VAL.

## method 2 -- add a pre-train set

step1--> 1.5

1.6 (a) DRAW  $\mathcal{D}_{\text{PT}}^R$  (PRE-TRAINING)  
(WITHOUT REPLACEMENT) FROM  $\mathcal{D}'$ :



(b) USE  $\mathcal{D}_{\text{PT}}^R$  TO LOOK A DATA,  
CONDUCT INITIAL TESTS, ETC.

(e.g., STEPS (2)-(5)).

(c) DISCARD  $\mathcal{D}_{\text{PT}}^R$  AF I.

(6) ~~PROCEED WITH (6)~~ PROCEED WITH (6). (already removed  $\mathcal{D}_{\text{Test}}$  though).

step 2-->3-->4-->5-->6 (preprocessing, feature selection, )

## 7. EVALUATE ITS PERFORMANCE.

(a) CALCULATE  $E_{in}(h_g)$  USING  $\Delta_{Tr}$  OR  $\Delta_{Val}$

CALCULATE  $E_{VC}$  USING  $d_{VC}$  OR  $H$

(ALSO  $N, \delta$ ).

GET "ERROR BAR" ON  $E_{out}(h_g)$ .

### method 3 -- use prior knowledge to design model

(1) USE PRIOR KNOWLEDGE OF PROBLEM TO:

- DECIDE ON FEATURES, PRE-PROC, ETC

- CONSTRUCT  $H$  (CONSIDERING  $d_{VC}(H)$ ,  
 $N_{Tr}, N_{Val}$ , ETC.)

(2) DRAW  $\Delta'$ ;  $\Delta_{Test}$

(3) SET ASIDE  $\Delta_{Test}$

(4) USE  $\Delta'$  FOR MODEL SELECTION, TRAINING,  
CHOOSING  $h_g$ .

(5) ONCE  $h_g$  IS FINAL, 7(a) AND 7(b)  
CAN BE USED.

### method 4

IV. CONSTRUCT YOUR OWN ~~METHOD~~ AS LONG AS  
YOU DON'T VIOLATE ASSUMPTIONS FOR  
GENERALIZATION ERROR ~~THAT~~ THAT YOU  
COMPUTE AND RELY ON.

### KEY ASSUMPTIONS

$$(i) E_{\text{out}}(h_g) \leq E_{\mathcal{D}_0}(h_g) + \sqrt{\frac{8}{N} \ln \frac{4[(2N)^d]^{VC+1}}{S}}$$

ASSUMES:

BASED ON  $H_0$ .

1.  $\mathcal{D}_0$  AND  $H_0$  MUST BE CONSISTENT.
2. INFO. IN  $\mathcal{D}_0$  CAN'T BE USED TO CONSTRUCT  $H_0$ .
3.  $N = N_{\mathcal{D}_0}$ .

$$(ii) E_{\text{out}}(h_g) \leq E_{\text{Test}}(h_g) + \sqrt{\frac{1}{2N} \ln \frac{2M}{S}}$$

$$M = 1.$$

ASSUME:

1. INFO. IN  $\mathcal{D}_{\text{Test}}$  CAN'T INFLUENCE CHOICE OF  $h_g$ .
2.  $N = N_{\text{Test}}$ .

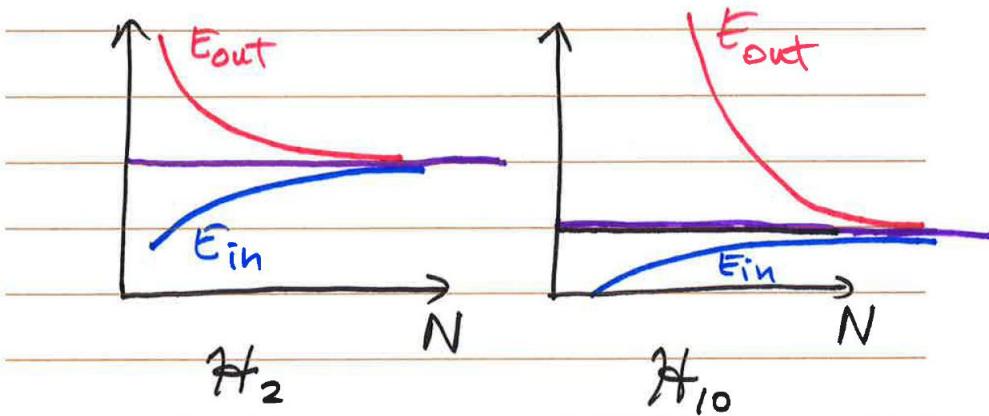
## Overfitting

DEF: OVERTIT IS "AN ANALYSIS WHICH CORRESPONDS TOO CLOSELY OR EXACTLY TO A PARTICULAR SET OF DATA".

[OXFORD DICTIONARY]

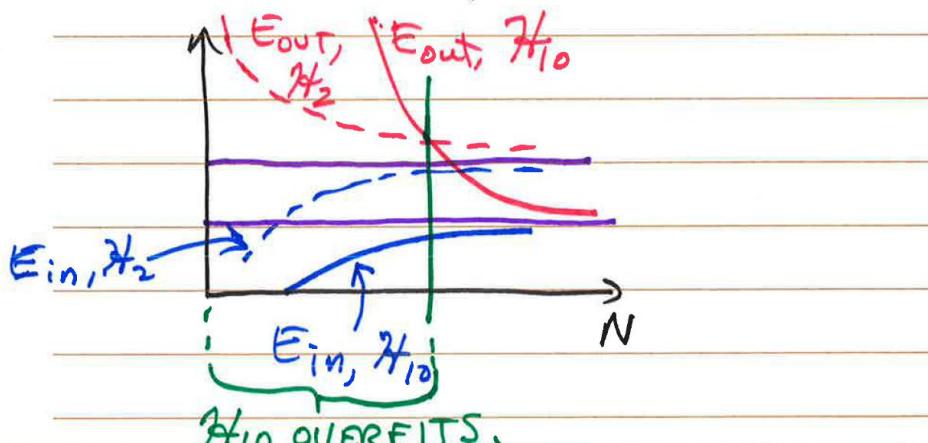
COMMON SYMPTOM OF OVERTITTING: PICKING A HYPOTHESIS WITH LOWER  $E_{\text{in}}$  RESULTS IN A HIGHER  $E_{\text{out}}$ .

## LEARNING CURVES:



TARGET:  $10^{+}$  ORDER POLYN., WITH NOISE.

IF  $E_{in}(h_k) \leq E_{in}(h_j)$ , AND  $E_{out}(h_k) > E_{out}(h_j)$   
 THEN WE CAN SAY  $h_k$  ~~overfit~~ overfits THE  
 DATA (RELATIVE TO  $h_j$ ).



## Regularization

basic concept

LEARNING ALG. FINDS:

$$h_g = \underset{h \in \mathcal{H}}{\operatorname{argmin}} E_{in}(h)$$

$\Omega(\mathcal{H})$  DEPENDS ON  $\mathcal{H}$  BUT NOT ON  $h_g$ .

WITH REGULARIZATION ( $\underline{w} = \underline{w}^{(0)}$  NON-AVAM.)

LET  $f_{\text{obj}}^{(r)}(\underline{h}_{\underline{w}}) = E_{\text{in}}(h_{\underline{w}})$  SUBJECT TO

$$\underline{w}^T \underline{w} \leq C$$

( $C$  IS A PARAMETER).

$$h_g = \underset{\underline{h} \in \mathcal{H}'}{\operatorname{argmin}} E_{\text{in}}(h_{\underline{w}})$$

in which  $\mathcal{H}' = \{ h \mid h \in \mathcal{H} \text{ and } \underline{w}^T \underline{w} \leq C \}$

→ DIFFERENT HYP. SET.

different conditions in Regularization

GRAPHICAL VIEW

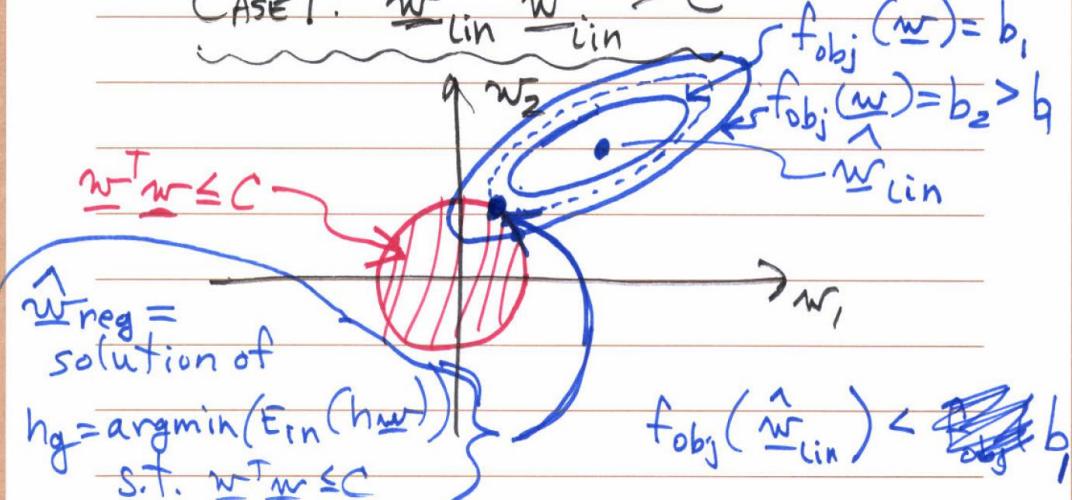
$$f_{\text{obj}}(\underline{w}) \triangleq E_{\text{in}}(h_{\underline{w}}) = \frac{1}{N} \sum_{n=1}^N (\underline{w}^T \underline{x}_n + w_0 - y_n)^2$$

example

$$= \text{min. at } \hat{\underline{w}}_{\text{lin}}$$

USING (i) ABOVE:

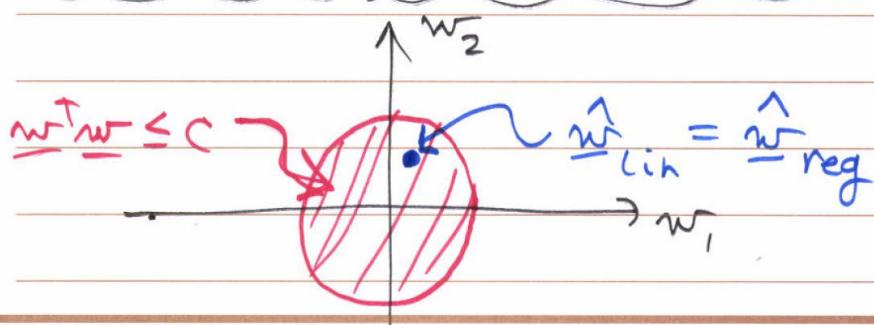
$$\text{CASE 1: } \hat{\underline{w}}_{\text{lin}}^T \hat{\underline{w}}_{\text{lin}} > C$$



→ IN THIS CASE,  $\hat{\underline{w}}_{\text{reg}}^T \hat{\underline{w}}_{\text{reg}} = C$ ,

AND  $\hat{\underline{w}}_{\text{reg}} \neq \hat{\underline{w}}_{\text{lin}}$ .

CASE 2:   $\hat{\underline{w}}_{\text{lin}}^T \hat{\underline{w}}_{\text{lin}} \leq C$ :



→ IN THIS CASE,  $\hat{\underline{w}}_{\text{reg}}^T \hat{\underline{w}}_{\text{reg}} \leq C$ , AND

Regularization in a Lagrange way

USE LAGRANGIAN OPTIMIZATION

(WITH INEQUALITY CONSTRAINT):

$$L(\underline{w}, \lambda_c) = E_{\text{in}}(h_{\underline{w}}) - \lambda_c [C - \underline{w}^T \underline{w}],$$

$$\nabla_{\underline{w}} L = \nabla_{\underline{w}} \left\{ E_{\text{in}}(h_{\underline{w}}) - \lambda_c [C - \underline{w}^T \underline{w}] \right\} \Bigg|_{\substack{\lambda_c \geq 0 \\ \underline{w} = \underline{w}_{\text{reg}}} = 0$$

$$\nabla_{\underline{w}} \left\{ E_{\text{in}}(h_{\underline{w}}) + \lambda_c \underline{w}^T \underline{w} \right\} \Bigg|_{\substack{\underline{w} = \underline{w}_{\text{reg}}} = 0$$

$\Rightarrow \underline{w}_{\text{reg}}$  (LOCALLY) MINIMIZES:

$$E_{\text{aug}}(h_{\underline{w}}) \triangleq E_{\text{in}}(h_{\underline{w}}) + \lambda_c \underline{w}^T \underline{w}, \lambda_c \geq 0.$$

TREAT  $\lambda_c$  AS A PARAMETER.

$$(iii) h_g = \underset{h \in \mathcal{H}}{\operatorname{argmin}} \left\{ E_{\text{in}}(h_g) + \lambda_c \underbrace{\underline{w}^T \underline{w}} \right\}, \lambda_c \geq 0.$$

$\ell_2$  REGULARIZER  
(ALSO CALLED "WEIGHT DECAY").

(i), (ii) HAD C AS PARAMETER.

(iii) HAS  $\lambda_c$  AS PARAMETER.

final function of loss(objective)

SUMMARY: THUS, WE CAN WRITE:

$$E_{\text{aug}}(h_{\underline{w}}) = E_{\text{in}}(h_{\underline{w}}) + \lambda \underline{w}^T \underline{w}, \lambda \geq 0$$

(UNCONSTRAINED OPTIMIZATION PROBLEM)

AND MORE GENERALLY:

$$E_{\text{aug}}(h_{\underline{w}}) = E_{\text{in}}(h_{\underline{w}}) + \frac{\lambda'}{N} \mathcal{L}(h_{\underline{w}}),$$

$\lambda' \triangleq N\lambda \geq 0$

## Model Selection

(based on all feasibility discussed before)

Bayes feature selection (MAP) is a way to make regularization (lasso?)

POSE AS A MAP ESTIMATION PROBLEM.

$$\begin{aligned}
 p(\underline{w} | \underline{x}) &= \frac{1}{K} p(\underline{x} | \underline{w}) p(\underline{w}) \\
 (1) \left\{ \begin{aligned} &= \frac{1}{K} \left[ \prod_{i=1}^N p(y_i | \underline{x}_i, \underline{w}) \right] p(\underline{w}) \\ &= \frac{1}{K} p(\underline{y} | \underline{X}, \underline{w}) p(\underline{w}) \end{aligned} \right.
 \end{aligned}$$

IF LINEAR REGRESSION w/ USUAL ASSUMPTION:

$$p(y | \underline{x}, \underline{w}) = N(y | \underline{w}^\top \underline{x}, \sigma^2)$$

PRIOR TERM:

$$\begin{aligned}
 p(\underline{w}) &= p(\underline{w} | \lambda) \\
 &= \prod_{j=1}^D \text{Lap}(w_j | 0, \frac{1}{\lambda}) \\
 &= \prod_{j=1}^D \frac{\lambda}{2} e^{-\lambda|w_j|} \propto \prod_{j=1}^D e^{-\lambda|w_j|}
 \end{aligned}$$

$$\begin{aligned}
 J(\underline{w}) &= -\ln p(\underline{y} | \underline{X}, \underline{w}) - \ln p(\underline{w} | \lambda) \\
 &= \text{NLL}(\underline{w}) + \cancel{\text{REGULARIZATION}} \lambda \|\underline{w}\|_1,
 \end{aligned}$$

$$\|\underline{w}\|_1 \triangleq \sum_{j=1}^D |w_j|.$$

FOR LINEAR MODEL:

$$\begin{aligned}
 &= \sum_{i=1}^N \frac{1}{2\sigma^2} [y_i - (w_0 + \underline{w}^\top \underline{x}_i)]^2 + \lambda \|\underline{w}\|_1, \\
 J(\underline{w}) &\propto \text{RSS}(\underline{w}) + \lambda' \|\underline{w}\|_1, \quad \lambda' = 2\lambda\sigma^2.
 \end{aligned}$$

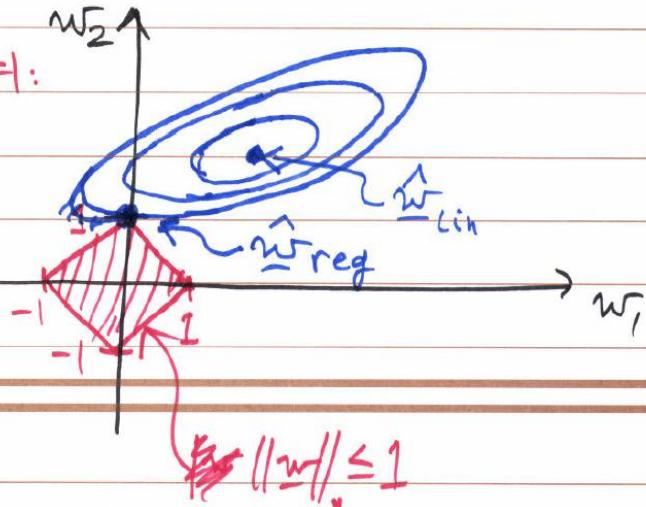
## L1 Regularization(Lasso)

### $\ell_1$ REGULARIZATION AND SPARSITY [M 13.3.1]

→ LOOK AT AS CONSTRAINED MIN. PROBLEMS:

$$\min_{\underline{w}} \text{RSS}(\underline{w}) \text{ s.t. } \|\underline{w}\|_1 \leq B \text{ (LASSO)}$$

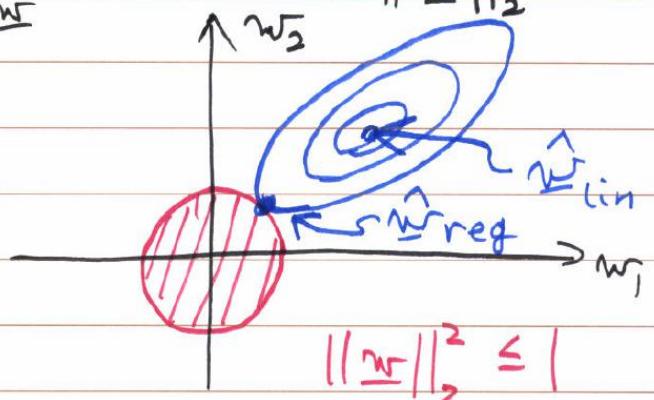
choose  $B=1$ :



$$\|\underline{w}\|_1 \leq 1$$

## L2 Regularization(Ridge)

$$\min_{\underline{w}} \text{RSS}(\underline{w}) \text{ s.t. } \|\underline{w}\|_2^2 \leq B \text{ (RIDGE)}$$



$$\|\underline{w}\|_2^2 \leq 1$$

bayesian variable feature selection

$$\underline{\gamma}_{D \times 1} : \gamma_j = \begin{cases} 1 & \text{FEATURE } j \text{ IS RELEVANT} \\ 0 & \text{otherwise} \end{cases}$$

FIND  $p(\underline{\gamma} | \underline{\alpha})$ .

$$p(\underline{\gamma} | \underline{\alpha}) = \frac{p(\underline{\alpha} | \underline{\gamma}) p(\underline{\gamma})}{p(\underline{\alpha})}$$

PRIOR TERM  $p(\underline{\gamma})$ ?

ASSUME:  $p(\underline{\gamma}) = \prod_{j=1}^D p(\gamma_j)$

COMMON CHOICE:

$$p(\gamma_j) = \text{Ber}(\gamma_j | \pi_0) = \pi_0^{\gamma_j} (1 - \pi_0)^{1 - \gamma_j}$$

$\pi_0$  = PROBABILITY THAT A FEATURE IS

$$\Rightarrow p(\underline{\gamma}) = \pi_0^{\sum \gamma_j} (1 - \pi_0)^{D - \sum \gamma_j}$$

LET  $\|\underline{\gamma}\|_0 \triangleq \sum_{j=1}^D \gamma_j = \# \text{ OF NONZERO ELEMENTS IN } \underline{\gamma}$ .

=  $\ell_0$  PSEUDO-NORM.

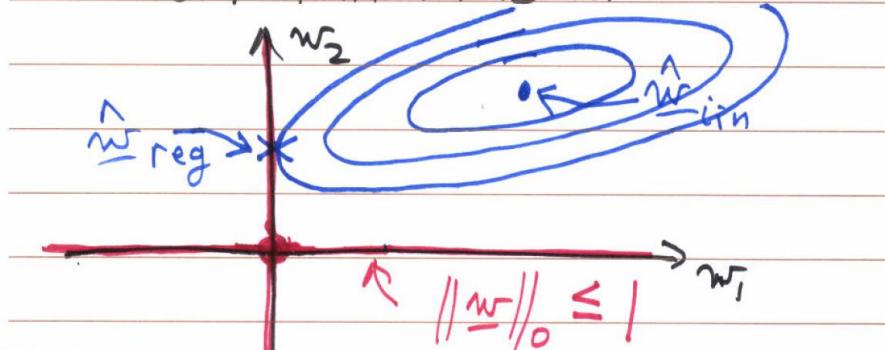
$$p(\underline{\gamma}) = \pi_0^{\|\underline{\gamma}\|_0} (1 - \pi_0)^{D - \|\underline{\gamma}\|_0}$$

## L0 Regularization

THEN:

$$f_{\text{obj}}(\underline{w}) = \|\underline{y} - \underline{X}\underline{w}\|_2^2 + \lambda \|\underline{w}\|_0$$

1. MIN. OVER CONTINUOUS VARIABLES  $\underline{w}$  NOW.
  2. Is  $f_{\text{obj}}$  CONVEX?
- $\rightarrow$  No,  $\|\underline{w}\|_0$  IS NOT CONVEX.  
 $\rightarrow$  SEE MURPHY [3, 2, 3] FOR MORE INFO [N, R, F]



## Validation

case1:

### KEY RELATIONS AND BOUNDS

$$(i) \left. \begin{array}{l} E_{\text{out}}(h_g) \leq E_{\mathcal{D}_0}(h_g) + \epsilon_{\text{eff}} \\ E_{\text{out}}(h_g) \leq E_{\mathcal{D}_0}(h_g) + \epsilon_{\text{vc}} \end{array} \right\} \begin{array}{l} \text{with probability} \\ \leq 1 - \delta \end{array}$$

$$\epsilon_{\text{eff}} = \sqrt{\frac{8}{N} \ln \frac{4m_{\mathcal{D}_0}(2N)}{\delta}}$$

$$\epsilon_{\text{vc}} = \sqrt{\frac{8}{N} \ln \frac{4[(2N)^{\alpha_{\text{vc}}} + 1]}{\delta}}$$

$$\epsilon_{\text{eff}} \leq \epsilon_{\text{vc}}, \quad N = |\mathcal{D}_0|$$

$$\text{e.g.: } \mathcal{D}_0 = \mathcal{D}_{\text{Tr}}, \quad \mathcal{H} = \mathcal{H}_{\text{Tr}}, \quad N = |\mathcal{D}_{\text{Tr}}|,$$

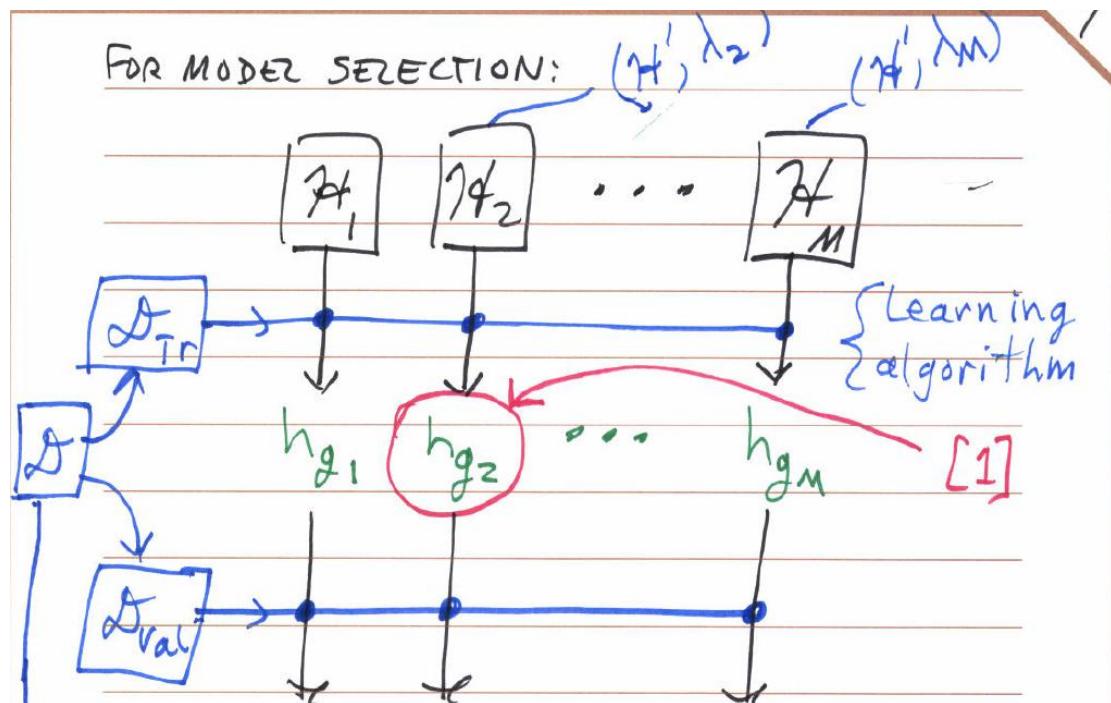
$$d_{vc} = d_{vc}(\mathcal{H}_{\text{Tr}}), \quad m_{\mathcal{H}}(2N) = m_{\mathcal{H}_{\text{Tr}}}(2N).$$

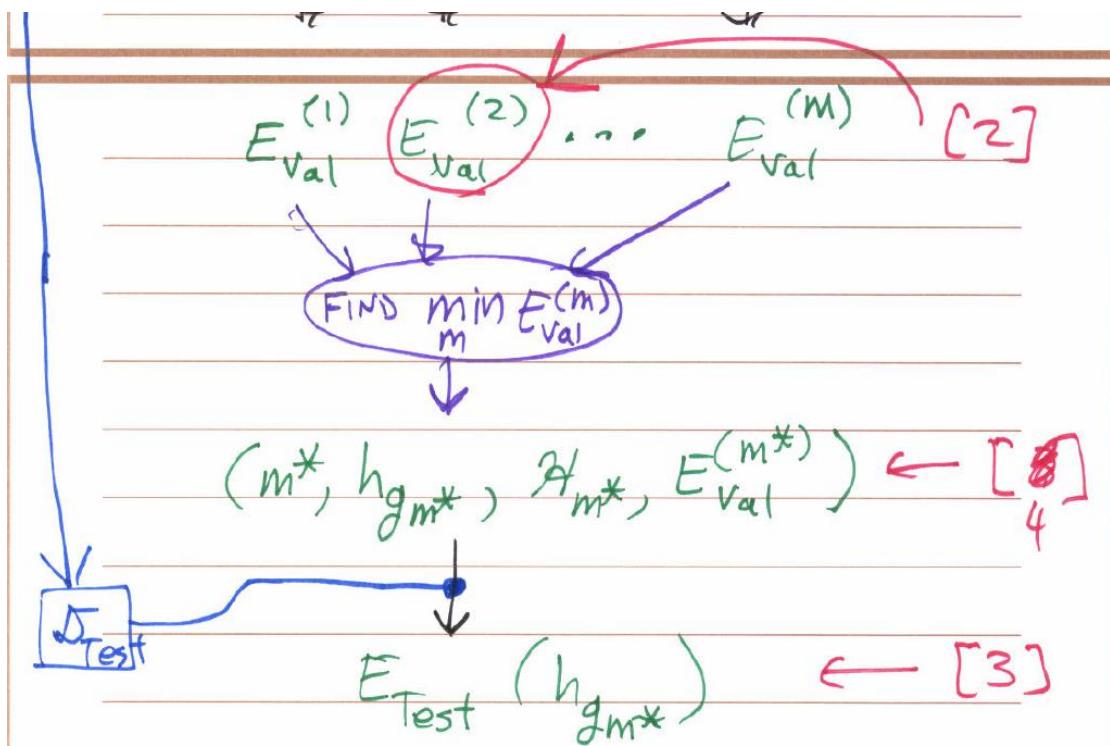
case 2:

$$(ii) \left\{ \begin{array}{l} E_{\text{out}}(h_g) \leq E_{\mathcal{D}_a}(h_g) + \sqrt{\frac{1}{2N} \ln \frac{2M}{\delta}} \\ \text{with pr. } \geq 1-\delta. \\ M = |\mathcal{H}|, \quad N = |\mathcal{D}_a| \end{array} \right.$$

e.g.: if  $\mathcal{D}_a = \mathcal{D}_{\text{Test}}$ , then usu.  $M = 1$ .

validation : for model selection (choose  $\lambda$  in regularization)





4 questions to understand training procedure:

[1] WHAT HYPOTHESIS SET DID  $h_{g_2}$  COME FROM?

A:  $\mathcal{H}_2$ .

THEORETICAL BOUNDS  $E_{out}(h_{g_2})$  USING  $\mathcal{D}_{Tr}$ ?

→ USE (i) WITH  $d_o = d_{Tr}$ ,

$\mathcal{H} = \mathcal{H}_2$ ,  $N = |\mathcal{D}_{Tr}|$

$d_{VC} = d_{VC}(\mathcal{H}_2)$ ,  $m_{\mathcal{H}_2}(2N) = m_{\mathcal{H}_2}(2N)$

[2] ~~eggs~~ WHAT HYP. SET DO WE USE TO GET  
 $E_{out}(h_{g_2})$  BOUND, BASED ON  $\mathcal{D}_{Val}$ ?

$$\mathcal{H} = \{h_{g_2}\}$$

→ USE (ii) WITH  $d_o = d_{Val}$ ,  $M = 1$ .

[3] How est.  $E_{\text{out}}(h_{g_m^*})$  from  $\mathcal{D}_{\text{Test}}$ ?

$$\mathcal{H} = \{h_{g_m^*}\}$$

→ Use (ii), with  $\mathcal{D}_a = \mathcal{D}_{\text{Test}}$ ,  $M=1$ .

[4] How est.  $E_{\text{out}}(h_{g_m^*})$  from  $\mathcal{D}_{\text{Val}}$ ?

$$\mathcal{H}'' = \{h_{g_1}, h_{g_2}, \dots, h_{g_M}\}.$$

→ Use (ii), with  $M = |\mathcal{H}''| = M$ ,

$$\mathcal{D}_a = \mathcal{D}_{\text{Val}}.$$

⇒ → Use (i), with  $m_{\mathcal{H}}(2N) = m_{\mathcal{H}''}(2N)$ ,

$$\text{or } d_{\text{vc}}(\mathcal{H}) = d_{\text{vc}}(\mathcal{H}''), \quad \mathcal{D}_0 = \mathcal{D}_{\text{Val}}.$$

## 5. Nonlinear Method

Adaptive basis function models

$$\hat{f}(\underline{x}) = w_0 + \sum_{m=1}^M w_m \phi_m(\underline{x})$$

IN WHICH  $\phi_m(\underline{x})$  IS LEARNED FROM THE DATA.

IF THE  $\phi_m(\underline{x})$  ARE PARAMETRIC, THEN:

$$\phi_m(\underline{x}) = \phi(\underline{x}; \underline{\nu}_m)$$

↑  
PARAMETERS OF  $\phi_m$ , TO BE

LEARNED FROM THE DATA.

## Decision Tree and Random Forest

### Decision Tree (CART)

concept

MODEL:

$$\begin{aligned}\hat{f}(\underline{x}) &= \sum_{m=1}^M w_m \mathbb{I}(\underline{x} \in R_m) \\ &= \sum_{m=1}^M w_m \phi(\underline{x}; \underline{\nu}_m)\end{aligned}$$

IN WHICH  $R_m = m^{\frac{1}{n}}$  REGION,

$\Sigma_m$  = PARAMETERS OF  $\phi_m$  (LEARNED FROM DATA).

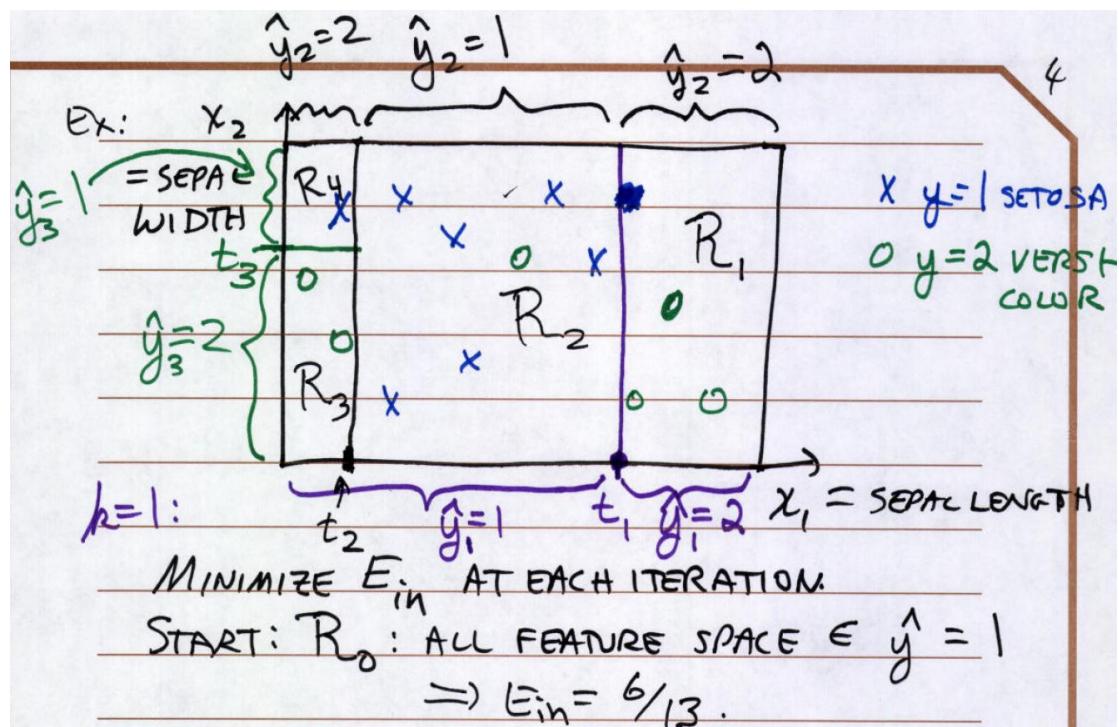
$\Rightarrow \hat{f}(x)$  is a piecewise constant approx.  
to  $f(x)$ .

CART: FORMS A TREE, AND A SET OF REGIONS  
 $R_m$  IN FEATURE SPACE.

TREE REGIONS  $R_m$  COME FROM

RECURSIVE SPLITTING OF A REGION INTO  
2, WITH SPLIT PERFORMED BY THRESHOLDING  
ONE COORDINATE VARIABLE (ONE FEATURE).

training procedure



$k=1$ : CHOOSE  $x_1$  OR  $x_2$  TO THRESHOLD

PICK THRESHOLD VALUE  $t_1$ ,

CHOOSE ~~AS~~ REGION LABELS.

$$\Rightarrow E_{in} = \cancel{3/13} \quad 3/13$$

$k=2$ : CHOOSE  $R_i$  TO SPLIT

CHOOSE  $x_1$  OR  $x_2$  TO THRESHOLD

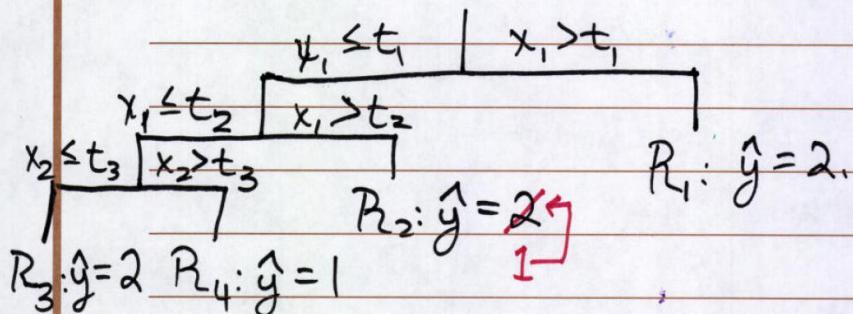
" THRESH. VALUE  $t_2$ .

CHOOSE REGION LABELS (2 NEW REGIONS)

$$\Rightarrow E_{in} = 2/13$$

$k=3$ : SEE ABOVE.  $\Rightarrow E_{in} = 1/13$ .

TREE FOR ABOVE:



3 A VARIETY OF HALTING CONDITIONS, SUCH AS:

- MAX. DEPTH OF TREE.

- MIN. REDUCTION IN COST FN. TO SPLIT A REGION.

- MIN. # OF ~~DATA PTS.~~ IN A FINAL REGION

[ref: Murphy].

NOTE: OPTIMAL FINAL TREE BALANCES COMPLEXITY OF TREE WITH N AND COMPLEXITY OF TARGET FCN.

theoretical

MORE RIGOROUSLY:

AT EACH ITERATION (EA. NODE OF TREE), WE DIVIDE ONE REGION  $R_m$  INTO TWO, BY THRES HOLDING ONE FEATURE  $x_j$ ; THUS:

AT  $k^{\text{th}}$  ITERATION:

$$\min_{m_1, j, t_k} \left\{ f_{\text{obj}}^{(k)} (w_{m_1}, w_{m_2}, \mathcal{D}; j, t_k, m) \right\}$$

in which:  $f_{\text{obj}}^{(k)} = \text{cost}_k \left\{ (\underline{x}_i, y_i) \in \mathcal{D} \right\}_{\text{after split}}$

For cost fcns. THAT ARE ADDITIVE BY REGION,  
of  $R_m$

THAT IS:

$$\text{cost} \left\{ (\underline{x}_i, y_i) \in \mathcal{D} \right\} = \alpha \sum_{m=1}^M \text{cost} \left\{ (\underline{x}_i, y_i) \in R_m \right\}$$

WE CAN INSTEAD USE THE ~~INCREMENTAL~~ INCREMENTAL CHANGE

IN COST:

$$\hat{f}_{\text{obj}}^{(k)} = \left[ \text{cost} \left\{ (\underline{x}_i, y_i) \in R_{m_1} \right\} + \text{cost} \left\{ (\underline{x}_i, y_i) \in R_{m_2} \right\} - \text{cost} \left\{ (\underline{x}_i, y_i) \in R_m \right\} \right]$$

IN WHICH:

$$R_{m_1}: R_m \cap \{x_j \leq t_k\}$$

$$R_{m_2}: R_m \cap \{x_j > t_k\}$$

FOR REGRESSION, COST FCN. IS TYPICALLY:

$$\text{cost} \{(x_i, y_i) \in R_m\} = \sum_{x_i \in R_m} (y_i - w_m)^2$$

FOR GIVEN  $R_m$ , THIS IS MINIMIZED BY:

$$w_m = w_m^* = \bar{y}_{R_m} \triangleq \frac{1}{N_{R_m}} \sum_{x_i \in R_m} y_i$$

NOTE: TO SAVE ON COMPUTATION, CART TYPICALLY CYCLES THROUGH ALL REGIONS  $R_m$ ,  $m=1, 2, \dots$ , SPLITTING EACH INTO 2 IF THE HALTING CONDITION ISN'T MET, INSTEAD OF FINDING THE BEST REGION TO SPLIT AT EACH ITERATION.

CART for multiclass

FOR CLASSIFICATION, USE A DIFFERENT COST FCN, E.G.:

$$\begin{aligned} \text{cost} \{(x_i, y_i) \in R_m\} \\ = \frac{1}{N_{R_m}} \sum_{x_i \in R_m} \mathbb{I}[y_i \neq \hat{y}(R_m)] \end{aligned}$$

CLASS  
ASSIGNMENT  
IN  $R_m$

CART WITH THESE COST FCNS. WORKS FOR  $C > 2$  CLASSES ALSO.

BECAUSE CART IS A GREEDY ALGORITHM, GROWING THE TREE UNTIL THE OPTIMAL STOPPING POINT TYPICALLY DOESN'T YIELD THE BEST RESULTS. USUALLY IT IS RUN PAST THIS POINT, TO YIELD A TREE THAT OVERFITS. THEN TREE IS PRUNED.

"WEAKEST LINK PRUNING":

- COLLAPSE THE INTERNAL NODE THAT

GIVES THE SMALLEST INCREASE IN COST FCN.; ITERATE.

- USE CROSS-VALIDATION TO HALT WHEN MIN. VALIDATION ERROR IS REACHED, (WITHIN  $1\sigma$ ).

### CART SUMMARY [M 16.2.4]

#### PROS

- ALGORITHM IS FAIRLY SIMPLE
- INCORPORATES LINEAR AND NONLINEAR BOUNDARIES ON ITS OWN.
- CAN INCLUDE SOME AUTOMATIC FEATURE SELECTION.

#### CONS

- PREDICTIVE ACCURACY OFTEN ISN'T AS GOOD AS WITH SOME OTHER MODELS.
- CAN BE UNSTABLE TO SLIGHT CHANGES IN DATA.

## Random Forest

(a) DRAW MANY DATASETS FROM  $\mathcal{D}_{\text{tr}}$ , WITH REPLACEMENT.

→ EACH GIVES RISE TO A TREE  $T_m$  AND EST.  $\hat{f}_m(\underline{x})$ .

AT EACH POINT  $\underline{x}$ ,

COULD TAKE AVERAGE RESULT (REGRESSION):

$$\hat{f}(\underline{x}) = \frac{1}{M} \sum_{m=1}^M \hat{f}_m(\underline{x})$$

OR COULD TAKE A VOTE (CLASSIFICATION).

→ THIS BY ITSELF IS CALLED BAGGING

FOR "BOOTSTRAP AGGREGATING").

But:

- DATASETS ARE (HIGHLY) CORRELATED
- REDUCES VAR SOME, BUT NOT A LOT.

a better way is use only a few features (best from a subset of features)

(b) BEFORE SPLITTING EACH REGION  $R_m$ ,

SELECT A RANDOM SUBSET OF  $d$  FEATURES ( $d < D$  or  $d \ll D$ ),

THEN SELECT BEST FEATURE OUT OF THE SUBSET TO THRESHOLD,

⇒ CORRELATION BETWEEN TREES IS TYPICALLY MUCH SMALLER

⇒ REDUCES VARIANCE BY A LOT MORE.

#### Algorithm

both data points and features are selected random in each iteration

1. For  $b = 1$  to  $B$  [Fig. 15.1].

(a) DRAW A SAMPLE DATASET  $\mathcal{D}^*$  AT RANDOM,  
WITH REPLACEMENT, OF SIZE  $N^*$ , FROM  
 $\mathcal{D}_{T_n}$ . TYPICALLY,  $N^* = N_{T_n}$ .

(b) GROW A RANDOM-FOREST TREE  $T_b$ ,  
USING  $\mathcal{D}^*$ :

CYCLE THROUGH EACH REGION  $R_m$ ;

FOR EACH  $R_m$ :

(i) SELECT  $d$  FEATURES AT RANDOM  
(~~from all  $D$  features~~)

• COMMON VALUES:

$$d \approx \sqrt{D}, \text{ or } d = \lfloor \sqrt{D} \rfloor$$

(ii) USE CART APPROACH TO SPLIT  
 $R_m$  BY FINDING OPTIMAL  
 $j, t_k, w$

(iii) SPLIT THE TREE NODE INTO TWO  
DAUGHTER NODES.

(iv) ITERATE UNTIL A HALTING  
CONDITION IS REACHED

• SEE CART HALTING CONDITIONS

2. OUTPUT THE SET OF TREES  $\{T_b, b=1, 2, \dots, B\}$

### 3. To USE RESULTING SET FOR PREDICTION:

REGRESSION -

$$\hat{f}(x) = \frac{1}{B} \sum_{b=1}^B \hat{f}_b(x)$$

PREDICTION FROM TREE  $T_b$

$$= \frac{1}{B} \sum_{b=1}^B \sum_{m=1}^M w_m^{(b)} I(x \in R_m^{(b)})$$

$$= \frac{1}{B} \sum_{b=1}^B \sum_{m=1}^M w_m^{(b)} \phi(x, \Sigma_m^{(b)})$$

to predict result

CLASSIFICATION -

CLASS PREDICTION:

IF  $\hat{y}^{(b)}(x)$  IS CLASS ASSIGNMENT FROM  
TREE  $T_b$ , THEN

$$\hat{y}(x) = \arg \max_c \sum_{b=1}^B I[\hat{y}^{(b)}(x) = c]$$

( $\Rightarrow$  TAKE VOTE,  $\hat{y}$  = WINNING CLASS)

CAN ALSO ESTIMATE CLASS POSTERIOR

PROBABILITIES  $p(\hat{y} = c | x, \Delta)$ , BY:

LET  $p_c^{(b)}(x)$  = FREQ. OF OCCURRENCE OF  
DATA POINTS  $y_i = c$  IN  $R_m^{(b)}$  THAT  
CONTAINS  $x$ , FROM TREE  $T_b$ .

AT EACH POINT  $x$ , TAKE AVE:

$$p(\hat{y} = c | x, \Delta) \approx \frac{1}{B} \sum_{b=1}^B p_c^{(b)}(x).$$

Gini index

Gini Index is a measure of **node purity** or impurity. It is a measure of how often a randomly chosen variable will be misclassified.

$$\text{Gini} = 1 - \sum_{i=0}^{c-1} [p_t]^2$$

c is the number of classes

### information gain

Information gain decides which **feature should be used to split** the data.

$$\text{Information Gain} = \text{Entropy}(\text{parent}) - [\text{Average entropy}(\text{children})]$$

$$\text{Entropy} = \sum_i -P_i (\log_2 P_i)$$

$P_i$  is probability of class i

### Boosting

IS ALSO AN ADAPTIVE BASIS FCN. MODEL:

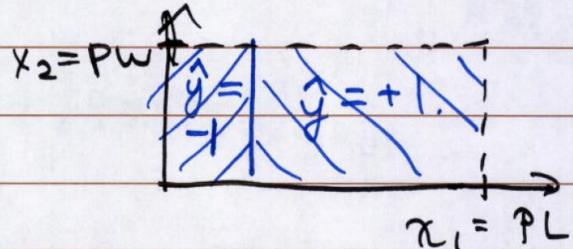
$$(1) \quad \hat{f}(\underline{x}) = w_0 + \sum_{m=1}^M w_m \phi(\underline{x}; \underline{\gamma}_m)$$

EACH  $\phi(\underline{x}, \underline{\gamma}_m)$ :

- IS A SIMPLE CLASSIFIER THAT CAN  
CLASSIFY THE ENTIRE FEATURE SPACE.

- IS A "WEAK LEARNER" THAT IS ONLY  
REQUIRED TO DO BETTER THAN CHANCE

e.g., WEAK LEARNER IS TYPICALLY A  
 "DECISION STUMP"— A 1-STAGE CART  
 RESULTING IN 1 NODE AND 2 LEAVES  
 (OR VARIANTS WITH  $> 2$  LEAVES).



### NOTATION

FOR 2-CLASS PROBLEM, LET CLASS LABELS  
 $\text{BE } \pm 1.$

$$(2) \hat{f}(x) = f_0 + \sum_{m=1}^M \beta_m \phi_m(x)$$

"WEIGHT" OR  
 IMPORTANCE OF  
 $m^{\text{th}}$  CLASSIFIER       $\beta_m \in \{-1, +1\}$   
 $= f_0 + \sum_{m=1}^M \beta_m \phi(x; \gamma_m)$

$$(\beta_m \in \mathbb{R})$$

INTERMEDIATE EXPRESSIONS FOR  $\hat{f}$ :

$$(3) \hat{f}_m(x) = f_0 + \sum_{m'=1}^m \beta_{m'} \phi_{m'}(x)$$

DATA POINTS:  $(x_i, \tilde{y}_i), \tilde{y}_i \in \{-1, +1\}$ .

TO FIND EACH  $\phi_m(x)$ ,

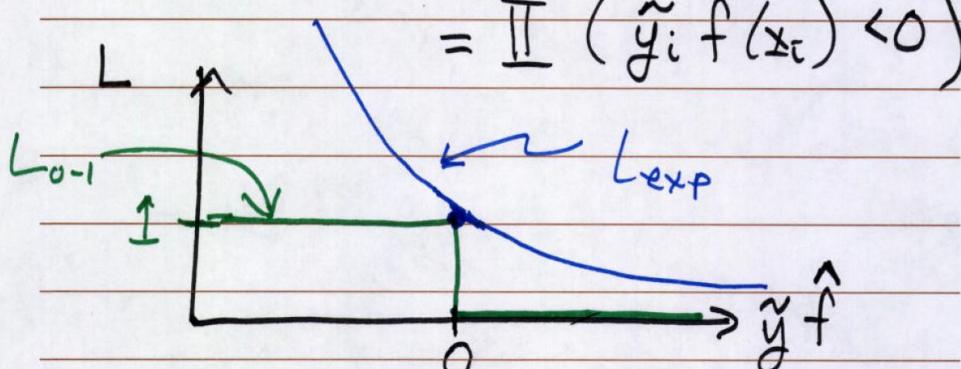
$$f_{\text{obj}} \left[ \{\beta_m, m=1, \dots, M\}, \{\gamma_m, m=1, \dots, M\}, \phi(x) \right]$$

$$= \frac{1}{N} \sum_{i=1}^N L(\hat{y}_i, \hat{f}(x_i))$$

↑  
LOSS FUNCTION.

$$0-1 \text{ Loss: } L_{0-1} = \mathbb{I}(\hat{y}_i \neq \text{sign}\{\hat{f}(x_i)\})$$

$$= \mathbb{I}(\hat{y}_i \hat{f}(x_i) < 0)$$



~~Ex~~  $f_{\text{obj}}$  using  $L_{0-1}$ : - NOT CONVEX  
- NOT AMENABLE TO GRADIENT TECHNIQUES.

(4) EXPONENTIAL LOSS:  $L_{\text{exp}} = \exp\{-\hat{y}_i \hat{f}(x_i)\}$

• (3 OTHER L FCNS.)

- CONVEX

- DIFFERENTIABLE

- PROVIDES CONFIDENCE MEASURE.

## FORWARD STAGewise ADDITIVE Modeling

1. INITIALIZE  $f_0 = 0$ .

2. FOR  $m=1 \rightarrow M$ :

(i) FIND:

$$(\beta_m, \gamma_m) = \underset{\beta_m, \gamma_m}{\operatorname{argmin}} \sum_{i=1}^N L[\tilde{y}_i, \hat{f}_{m-1}(x_i) + \beta_m \phi(x_i, \gamma_m)]$$

$$(ii) \hat{f}_m(x) = \hat{f}_{m-1}(x) + \beta_m \phi_m(x, \gamma_m)$$

3. FINAL CLASSIFIER IS:

$$\hat{y}(x) = \operatorname{sign} \{ \hat{f}_M(x) \}$$

WANT TO FIND:

$$\hat{f}(x) = \underset{f(x)}{\operatorname{argmin}} \sum_{i=1}^N L_{\text{exp}} \left[ \tilde{y}_i, f(x_i) \right]$$

$$\rightarrow \underset{f_0, \beta_m, \gamma_m, m=1, 2, \dots, M}{\operatorname{argmin}} \sum_{i=1}^N L_{\text{exp}} \left[ \tilde{y}_i, f_0 + \sum_{m=1}^M \beta_m \phi(x_i, \gamma_m) \right]$$

EASIER TO MIN. EACH TERM, SEQUENTIALLY:

AT  $m^{\text{th}}$  ITERATION:

$$\underset{\beta_m, \gamma_m}{\operatorname{argmin}} \sum_{i=1}^N L_{\text{exp}} \left[ \tilde{y}_i, \hat{f}_{m-1}(x_i) + \beta_m \phi(x_i, \gamma_m) \right]$$

1. INITIALIZED  $f_0 = 0$ .

2. FOR  $m=1 \rightarrow M$ :

(i) FIND:

$$(\beta_m, \gamma_m) = \underset{\beta_m, \gamma_m}{\operatorname{argmin}} \sum_{i=1}^N L[\tilde{y}_i, \hat{f}_{m-1}(x_i) + \beta_m \phi(x_i, \gamma_m)]$$

$$(ii) \hat{f}_m(x) = \hat{f}_{m-1}(x) + \beta_m \phi_m(x, \gamma_m)$$

3. FINAL CLASSIFIER IS:

$$\hat{y}(x) = \operatorname{sign} \{ \hat{f}_M(x) \}$$

### Adaboosting (Adaptive Boosting)

weak learning: decision stump (one step decision tree)

ADABOOST

→ USE  $L = L_{\text{exp}}$

FROM ABOVE:

2.(i) FIND  $(\beta_m, \gamma_m)$

$$= \underset{\beta_m, \gamma_m}{\operatorname{argmin}} \sum_{i=1}^N L_{\text{exp}}[\tilde{y}_i, \hat{f}_{m-1}(x_i) + \beta_m \phi(x_i, \gamma_m)]$$

$L_m$

$$(5) \quad L_m = \sum_i \exp \left\{ -\tilde{y}_i \left[ \hat{f}_{m-1}(x_i) + \beta_m' \phi(x_i, \underline{x}_m) \right] \right\}$$

$w_{i,m}$  (const. of  $\beta_m, \underline{x}_m$ )

$$(6) \quad \text{LET } w_{i,m} \triangleq \exp \left\{ -\tilde{y}_i \hat{f}_{m-1}(x_i) \right\}$$

$$(7) \quad L_m = \sum_{i=1}^N w_{i,m} \exp \left[ -\tilde{y}_i (\beta_m' \phi(x_i, \underline{x}_m)) \right]$$

Sum over all datapts.      weight on  $i^{\text{th}}$  datapt.,  
 at  $m^{\text{th}}$  iteration.       $\beta_m' > 0.$   
 behaves like regular exp. loss

$L_m$  CAN BE MINIMIZED ALGEBRAICALLY.  
 [see Murphy].

CAN RE-ARRANGE  $L_m$  ABOVE (see Murphy),  
 to get EQNS. FOR ADABOOST ALGORITHM.

$\Rightarrow$

(\*)  $\phi(\underline{x}_i, \underline{y}_m)$  IS CHOSEN TO MINIMIZE A  
SUM OF WEIGHTS OF MISCLASSIFIED DATA  
POINTS.

$$(**) \beta_m = \frac{1}{2} \log \frac{1 - \text{err}_m}{\text{err}_m}, \text{ with:}$$

$$(***) \text{err}_m = \frac{\sum_{i=1}^N w_{i,m} \mathbb{I}[\tilde{y}_i \neq \phi(\underline{x}_i, \underline{y}_m)]}{\sum_{i=1}^N w_{i,m}}$$

= SAMPLE-WEIGHTED ERROR RATE  
(AT  $m^{\text{th}}$  ITERATION)

Algorithm

1. INITIALIZE  $w_i = \frac{1}{N} \quad \forall i$ .

2. FOR  $m = 1$  TO  $M$ :

(i) TRAIN CLASSIFIER  $\phi_m(\underline{x})$  [1-NODE CART]  
ON WEIGHTED DATASET  $\mathcal{D}_m$  (WEIGHTS  $w_{i,m}$ )  
PER (\*).

(ii) COMPUTE [FROM (\*\*\*)]:

$$\text{err}_m = \frac{\sum_{i=1}^N w_i \mathbb{I}[\tilde{y}_i \neq \phi_m(\underline{x}_i)]}{\sum_{i=1}^N w_i}$$

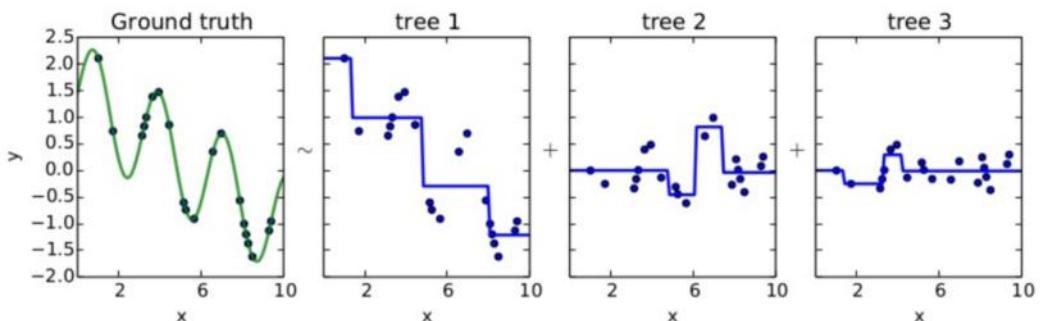
(iii) COMPUTE  
 $\alpha_m = \log \left[ \frac{1 - \text{err}_m}{\text{err}_m} \right] = 2 \beta_m$  in (\*\*).

(iv) UPDATE  $w_i \leftarrow w_i \exp[\alpha_m \mathbb{I}(\hat{y}_i \neq \phi_m(x_i))]$   
 $H_i$ .

3. RETURN  $\hat{f}(x) = \sum_{m=1}^M \alpha_m \phi_m(x)$   
AND  $\hat{y}(x) = \text{sign} \{ \hat{f}(x) \}$ .

## Gradient Boosting

Gradient Boosting method tries to fit the new predictor to the **residual errors** made by the previous predictor. (still use all the data, but each time data will change to it's residual)



## XGBoosting (eXtreme Gradient Boosting) (Learn)

gradient boosted decision trees designed for speed and performance

## 6. Semi-Supervised Learning

TRAINING SET:

LABELLED INSTANCES  $\mathcal{D}_L = \{(x_i, y_i)\}_{i=1}^l$

AND

UNLABELLED INSTANCES  $\mathcal{D}_U = \{x_j\}_{j=l+1}^{l+u}$

Assume unlabeled:  $p(x)$   
 consists, tends to labeled:  $p(x|y)$ , or  $p(y|x)$   

$$p(x) = \sum_y p(x|y)p(y)$$

Two major types of SSL:

### INDUCTIVE SSL

LEARNS  $\hat{y} = \hat{f}(x)$  over all feature space  $\mathcal{X}$ .

### TRANSDUCTIVE SSL

LEARNS  $\hat{y}_i = \hat{f}(x_i) \quad \forall x_i \in \mathcal{D}_U$ .

SSL Models/ Algorithm

### Self-Training

#### Algorithm 2.4. Self-training. (WRAPPER)

Input: labeled data  $\{(x_i, y_i)\}_{i=1}^l$ , unlabeled data  $\{x_j\}_{j=l+1}^{l+u}$ .

1. Initially, let  $L = \{(x_i, y_i)\}_{i=1}^l$  and  $U = \{x_j\}_{j=l+1}^{l+u}$ .

2. Repeat:

3. Train  $\hat{f}$  from  $L$  using supervised learning.

4. Apply  $\hat{f}$  to the unlabeled instances in  $U$ .

5. Remove a subset  $S$  from  $U$ ; add  $\{(x, \hat{f}(x)) | x \in S\}$  to  $L$ .

$\hookrightarrow$  ~~DATA POINTS WITH HIGHEST~~  
 DATA POINTS WITH HIGHEST  
 CONFIDENCE OF  $\hat{f}(x)$  PRE-  
 DITION.

SPECIFIC ASSUMPTION: DATA POINTS WITH HIGHEST  
 CONFIDENCE OF  $\hat{f}(x)$  PREDICTION TEND TO BE  
 EX: Algorithm 2.7. Propagating 1-Nearest-Neighbor. CORRECT.

### Propagating 1-Nearest-Neighbour

### Ex: Algorithm 2.7. Propagating 1-Nearest-Neighbor.

KORREKU.

Input: labeled data  $\{(\mathbf{x}_i, y_i)\}_{i=1}^l$ , unlabeled data  $\{\mathbf{x}_j\}_{j=l+1}^{l+u}$ , distance function  $d()$ .

1. Initially, let  $L = \{(\mathbf{x}_i, y_i)\}_{i=1}^l$  and  $U = \{\mathbf{x}_j\}_{j=l+1}^{l+u}$ .

2. Repeat until  $U$  is empty:

3. Select  $\mathbf{x} = \operatorname{argmin}_{\mathbf{x} \in U} \min_{\mathbf{x}' \in L} d(\mathbf{x}, \mathbf{x}')$ .

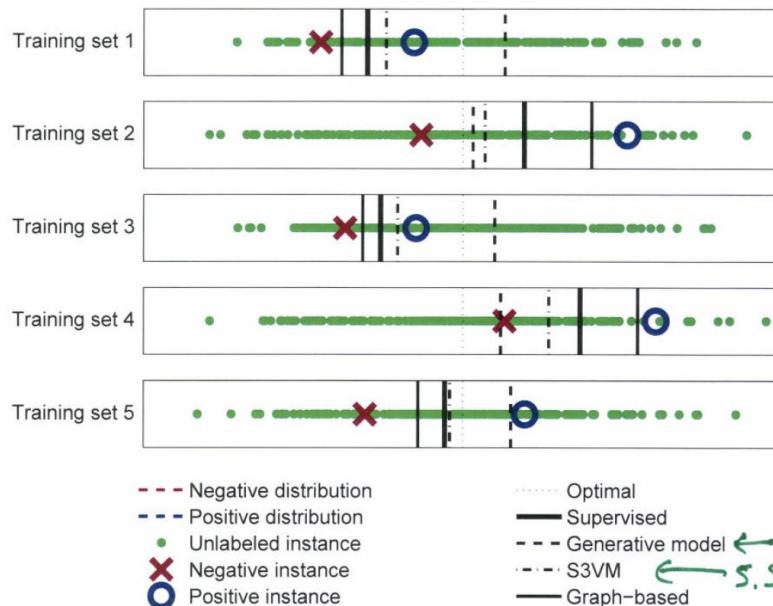
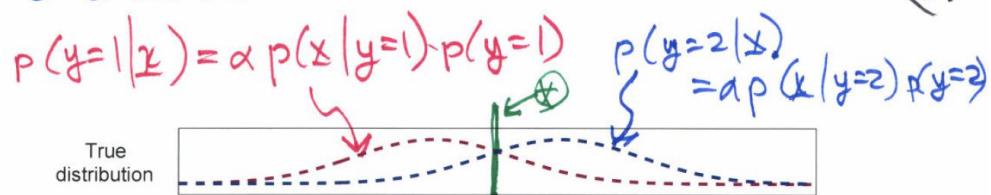
4. Set  $f(\mathbf{x})$  to the label of  $\mathbf{x}$ 's nearest instance in  $L$ . Break ties randomly.

5. Remove  $\mathbf{x}$  from  $U$ ; add  $(\mathbf{x}, f(\mathbf{x}))$  to  $L$ .

FIND THE UNLABELED DATA POINT  $\underline{\mathbf{x}}_j$   
THAT IS CLOSEST TO A LABELED  
DATA POINT  $\underline{\mathbf{x}}'_i$ .  
 $\Rightarrow \underline{\mathbf{x}}_j$  IS S.

## Mixture Model and Parameter Classification

$C=2$  classes. 1D.



SUPPOSE WE MODEL  $p(\underline{x} | y)$  AS A PARTICULAR PDF WITH SOME UNKNOWN PARAMETERS, e.g.:

$$p(\underline{x}|y) = N(\underline{x} | \mu_y, \Sigma_y)$$

$\mu_y$  AND  $\Sigma_y$  ARE UNKNOWN PARAMETERS  $\Theta$ .

$\mu_y$  AND  $\Sigma_y$ ,  $y=1, 2, \dots$  ARE UNKNOWN PARAMETERS  $\Theta$ .

POSTERIOR PREDICTIVE  $p(y|\underline{x}) = ?$

$$p(y|\underline{x}) = \frac{p(\underline{x}|y)p(y)}{p(\underline{x})}$$

2 ways: MLE and MAP

LET'S MODEL EACH CLASS AS A SPECIFIED DENSITY WITH UNKNOWN PARAMETERS:

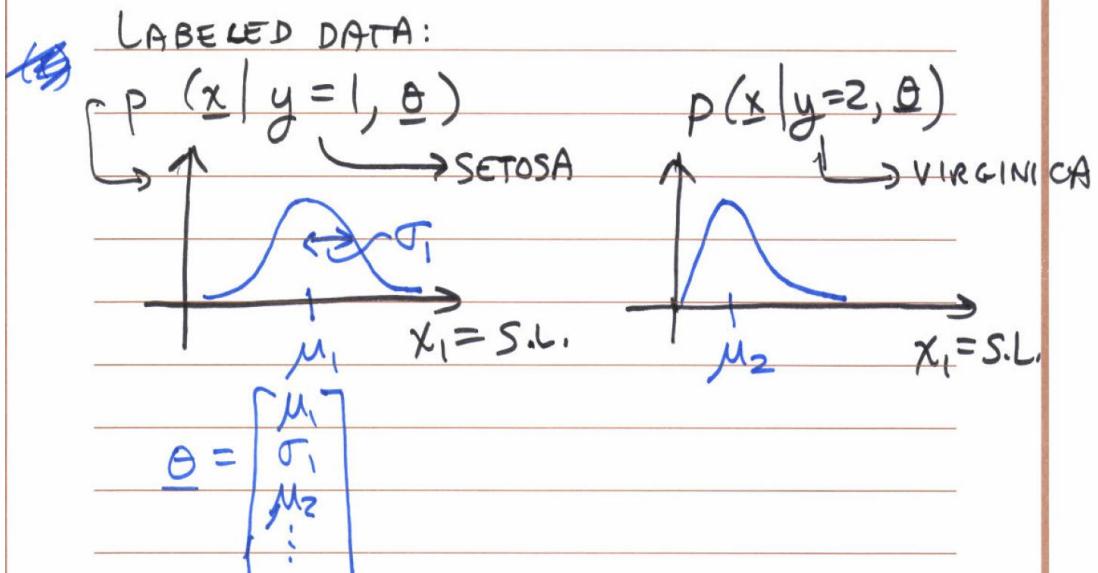
$$p(\underline{x}, y | \Theta) = p(\underline{x} | y, \Theta)p(y | \Theta)$$

(1)

$$= \underbrace{p(\underline{x} | y, \Theta)}_{\text{CLASS-CONDITIONAL DENSITY, CONDITIONED ON } \Theta} \pi_y$$

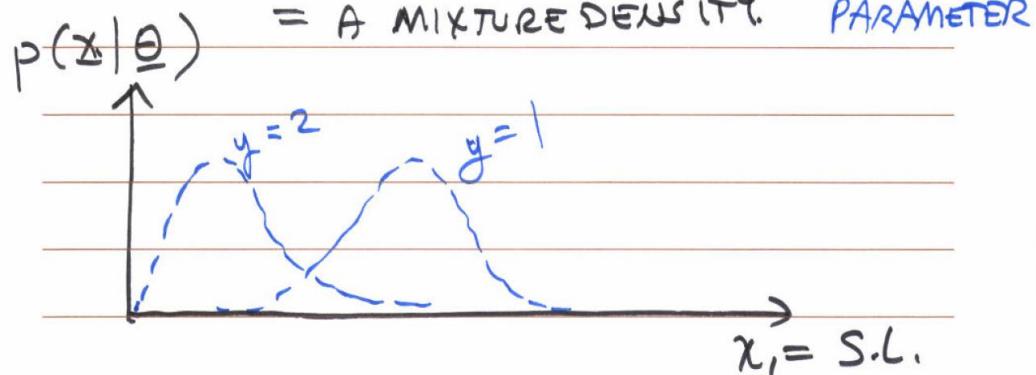
$\pi_y = p(y)$   
 $= \text{PRIOR ON } y$ .  
 CLASS-CONDITIONAL DENSITY, CONDITIONED ON  $\Theta$ .

OUR MODEL:



UNLABELED DATA:

(2)  $p(x|\underline{\theta}) = \sum_{y=1}^C p(x|y, \theta) \underbrace{p(y|\underline{\theta})}_{\text{COMPONENT DENS.}}$   $\underbrace{\pi_y}_{\text{MIXING PARAMETER}}$



MLE

ONE WAY: MLE:

$$\begin{aligned}\hat{\theta} &= \underset{\theta}{\operatorname{argmax}} p(\delta | \theta) = \underset{\theta}{\operatorname{argmax}} \ln p(\delta | \theta) \\ &= \underset{\theta}{\operatorname{argmax}} \sum_{i=1}^l \ln p(x_i, y_i | \theta) \\ &= \underset{\theta}{\operatorname{argmax}} \sum_{i=1}^l \ln \left[ \underbrace{p(x_i | y_i, \theta)}_{\text{KNOWN.}} \underbrace{p(y_i | \theta)}_{\text{PRIOR ON } y.} \right]\end{aligned}$$

EM Algorithm

EM ALGORITHM (GENERAL FORMULATION)  
[FOLLOWS SSL TEXT; ALSO  
MURPHY 11.4]

COMMONLY USED FOR:

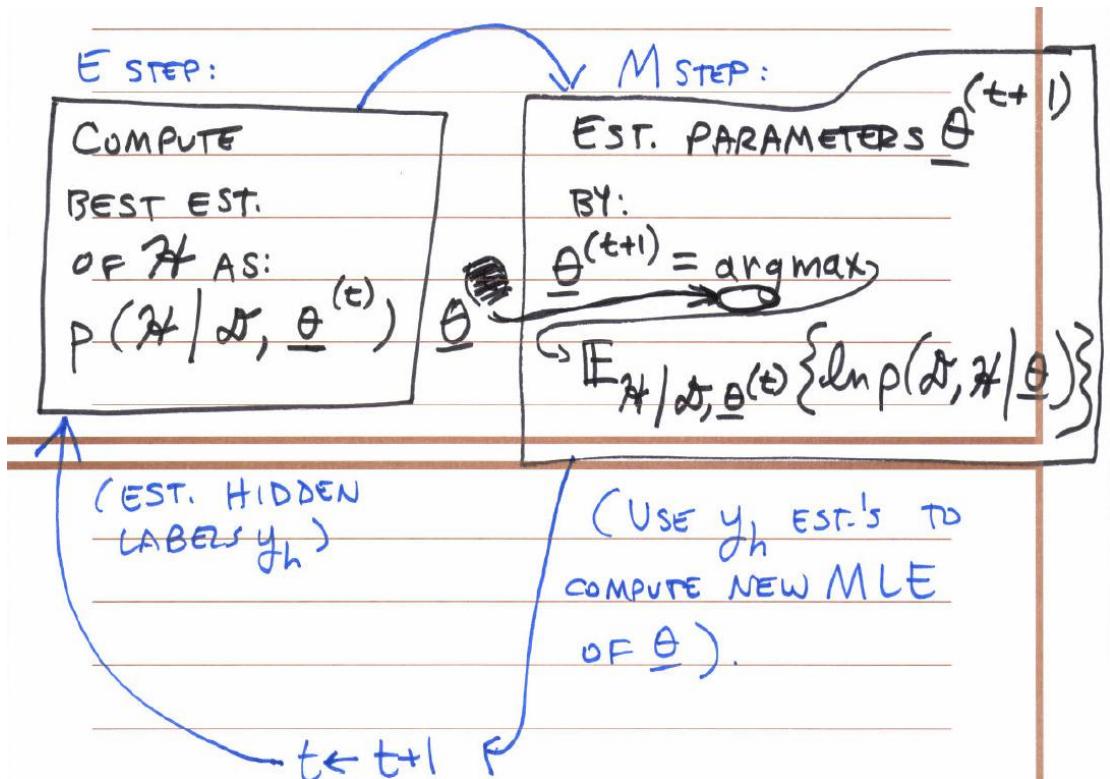
- WORKING WITH MISSING DATA.
- FINDING MLE IN DIFFICULT SITUATIONS.
- ESTIMATING QUANTITIES IN MIXTURE MODELS.

LET  $\delta$  BE ALL THE DATA:

$$\{(x_i, y_i), i=1, \dots, l; x_h, h=l+1, \dots, l+u\}$$

LET  $\gamma$  BE THE HIDDEN LABELS

$$\{y_h, h=l+1, \dots, l+u\}$$



halt when  $P(\underline{d} / \underline{\theta}(t))$  converge

### EM PROPERTIES

1. CAN BE SHOWN THAT  $p(\underline{d} | \underline{\theta})$  INCREASES AT EVERY ITERATION.

2. CONVERGES TO A LOCAL OPTIMUM.

Is  $-\ln p(\underline{d} | \underline{\theta})$  A CONVEX Fcn. OF  $\underline{\theta}$ ?  
 ↗ GENERALLY, NO.

3. RESULT DEPENDS ON STARTING POINT  $\underline{\theta}^{(0)}$ .

COMMON CHOICE:  $\underline{\theta}^{(0)} = \hat{\underline{\theta}}_{\text{MLE}}$  BASED ON  $\underline{p}_i$ .

Application of EM Algorithm

E step

## FOR E STEP

LET  $i$  INDEX THE DATA PTS. IN  $\mathcal{D}_L$ ;  
 $h \ll \ll \ll \ll \ll$  IN  $\mathcal{D}_U$ .

$$\begin{aligned}
 p(\mathbf{z} | \mathbf{x}, \underline{\theta}) &= \prod_{h=l+1}^{l+u} p(y_h | \underline{x}_h, \underline{\theta}) \\
 (3) \quad &= \prod_h p(y_h | \underline{x}_h, \underline{\theta})
 \end{aligned}$$

(KNOW  $\underline{\theta} \Rightarrow$  DON'T NEED OTHER DATA.)

$$\begin{aligned}
 (4) \quad p(y_h = c | \underline{x}_h, \underline{\theta}) &= \frac{p(x_h | y_h = c, \underline{\theta}) p(y_h = c)}{\sum_{y_h=1}^C p(y_h | \underline{\theta}) p(x_h | y_h, \underline{\theta})}
 \end{aligned}$$

LET  $\gamma_{hc} \triangleq p(y_h = c | \underline{x}_h, \underline{\theta})$

DATA PT.  
 INDEX  
 (UNLABELED)  $\uparrow$  CLASS  
 ASSIGNMENT

$\gamma_{hc}$  = RESPONSIBILITY OF  $y_h = c$  LABEL

FOR DATA PT.  $\underline{x}_h$ .

= "SOFT LABEL" FOR DATA PT.  $\underline{x}_h$ .

$$p(y_h = c | \underline{x}_h, \underline{\theta}).$$

M step

### FOR M STEP

$$\max_{\Theta} E_{\mathcal{H}|\mathcal{D}, \Theta^{(t)}} \left\{ \ln p(\mathcal{D}, \mathcal{H} | \Theta) \right\}$$

$$= \max_{\Theta} \left\{ \sum_{\mathcal{H}} p(\mathcal{H} | \mathcal{D}, \Theta^{(t)}) \ln p(\mathcal{D}, \mathcal{H} | \Theta) \right\}$$

$$p(\mathcal{D}, \mathcal{H} | \Theta) = \underbrace{p(\mathcal{H} | \mathcal{D}, \Theta)}_{\text{GIVEN ABOVE.}} \underbrace{p(\mathcal{D} | \Theta)}_{\text{LIKELIHOOD OF ALL KNOWN (GIVEN) DATA.}}$$

EQ. (3), (4).

$$p(\mathcal{D} | \Theta) = \left[ \prod_{i=1}^l p(x_i, y_i | \Theta) \right] \prod_{h=l+1}^{l+u} p(x_h | \Theta)$$

$$p(x_i, y_i | \Theta) = p(x_i | y_i, \Theta) p(y_i | \Theta)$$

MODEL FOR LABELED DATA.

MIXTURE DENSITY (EQ-(2)):

$$p(x_h | \Theta) = \sum_{y_k=1}^C p(x_h | y_k, \Theta) \pi_{y_k} . \quad (*)$$

Conclusion of EM:

works well when model is correct, otherwise may not good

## 7. Unsupervised Learning

### UNSUPERVISED LEARNING (USL)

SAMPLES (DATA POINTS) IN  $\mathcal{D}$  HAVE NO LABELS.

EXAMPLES OF ITS USEFULNESS:

- CLUSTERING

e.g.: RECOMMENDER SYSTEMS (MOVIES, NEWS)

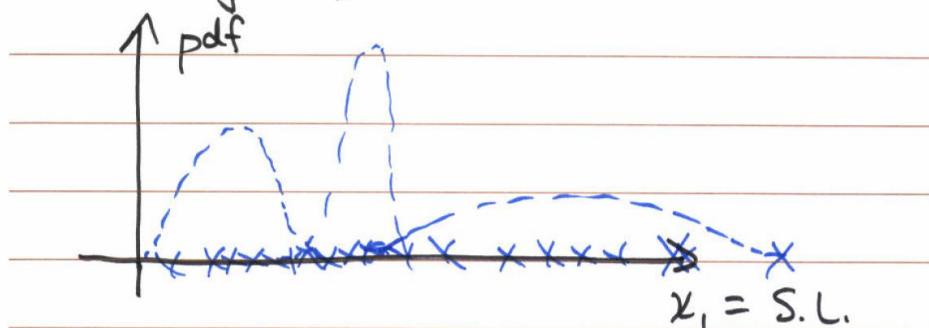
- DIMENSIONALITY REDUCTION (e.g., PCA)

- FILLING IN / DEALING WITH MISSING DATA.

- LEARNING ABOUT CHARACTERISTICS IN THE DATA.

### Density Estimate

MODEL EACH CLUSTER AS A pdf ~~with~~ WITH UNKNOWN PARAMETERS (TYPICAL FOR CONTINUOUS FEATURES  $x_i \in \mathbb{R}$ ).



THUS:

$$P(x|\theta) = \sum_{k=1}^K P(x|z=k, \underline{\theta}) P(z=k|\theta)$$

$z$  = CLUSTER INDEX

$\underline{\pi}_k$  = PARAMETERS  
FOR CLUSTER  $k$ .

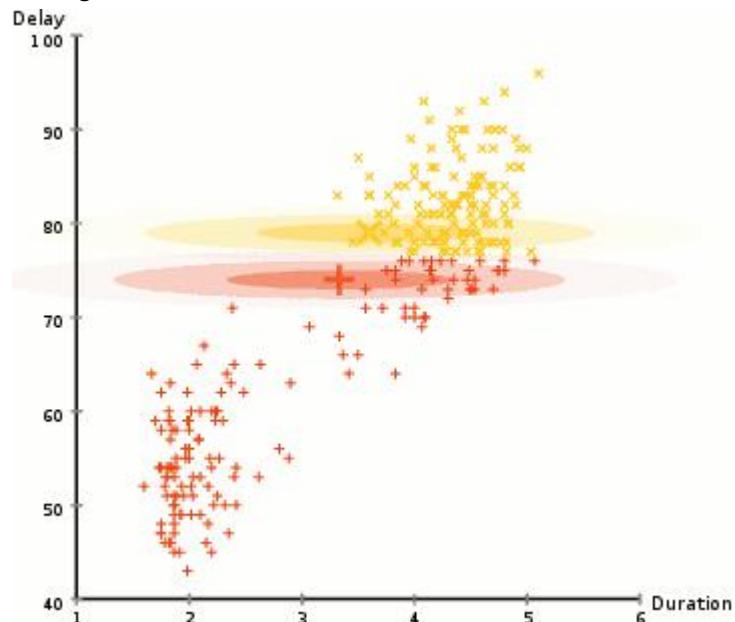
= A MIXTURE MODEL

$$P(x|\theta) = \sum_{k=1}^K \pi_k P(x|z=k, \underline{\theta}_k)$$

$$\text{AND } \sum_{k=1}^K \pi_k = 1.$$

OUR MODEL  
FOR  
CLUSTER  $k$ .

EM Algorithm



## EM FOR CLUSTERING USING MIXTURE MODELS

SAME BASIC ALGORITHM AS EM FOR SSL.

$$\text{LET } \mathcal{D} = \{x_i\}_{i=1}^N$$

$$\mathcal{H} = \{z_i\}_{i=1}^N, z_i \text{ IS CLUSTER LABEL}$$

FOR DATA PT.  $x_i$

$$z_i \in \{1, 2, \dots, K\}.$$

### ALGORITHM

1. INITIALIZE  $t=0$  AND  $\underline{\theta}^{(0)}$

2. ITERATE (INDEX t):

2.1 E STEP: COMPUTE  $p(\mathcal{H} | \mathcal{D}, \underline{\theta}^{(t)})$

2.2. M STEP: FIND:

$$\underline{\theta}^{(t+1)} = \underset{\underline{\theta}}{\operatorname{argmax}} \mathbb{E}_{\mathcal{H} | \mathcal{D}, \underline{\theta}^{(t)}} \left\{ \ln p(\mathcal{D}, \mathcal{H} | \underline{\theta}) \right\}$$

2.3  $t \leftarrow t + 1$

2.4 HALT WHEN  $p(\mathcal{D} | \underline{\theta}^{(t)})$  CONVERGES.

3. OUTPUT  $\hat{\underline{\theta}} = \underline{\theta}^{(t)}$

E Step

### EQUATIONS FOR E STEP

$$p(\mathcal{H} | \mathcal{D}, \underline{\theta}) = \prod_{i=1}^N \underbrace{p(z_i | x_i, \underline{\theta})}_{\text{our model}} \overbrace{\pi_k}^{\text{prior}}$$

$$p(z=k | x, \underline{\theta}_k) = \frac{p(x | z=k, \underline{\theta}_k) p(z=k | \underline{\theta}_k)}{\sum_{k'=1}^K p(x | z=k', \underline{\theta}_{k'}) p(z=k' | \underline{\theta}_{k'})}$$

$$\rightarrow \text{SOFT LABEL } \gamma_{ik}^{(t)} = p(z_i=k | x_i, \underline{\theta}_k^{(t)})$$

M Step

### EQNS. FOR M STEP

$$\begin{aligned} p(\underline{x}, \underline{z} | \underline{\theta}) &= \prod_{i=1}^N p(x_i, z_i | \underline{\theta}) \\ &= \prod_{i=1}^N \underbrace{p(x_i | z_i, \underline{\theta})}_{\text{our model}} \underbrace{p(z_i | \underline{\theta})}_{\pi_k} \\ \rightarrow \underline{\theta}^{(t+1)} &= \underset{\underline{\theta}}{\operatorname{argmax}} \left\{ \sum_{\underline{z}} p(\underline{z} | \underline{x}, \underline{\theta}^{(t)}) \cdot \ln p(\underline{x}, \underline{z} | \underline{\theta}) \right\} \end{aligned}$$

1. ALGORITHM CHARACTERISTICS - SAME AS FOR  
SSL EM.

2. CHOICE OF  $\underline{\theta}^{(0)}$  - IF NO PRIOR KNOWLEDGE,  
CAN RUN ALG. A NUMBER OF TIMES WITH  
DIFFERENT ( $\sim$  RANDOM) CHOICES OF  $\underline{\theta}^{(0)}$ ,  
COMPARE RESULTS USING  $p(\underline{x} | \underline{\theta})$ .

3. DOES THERE EXIST A UNIQUE BEST SOLUTION?  
IF MIXTURE  $p(\underline{x} | \underline{\theta})$  IS IDENTIFIABLE,  
THEN ~~exists~~ IN LIMIT  $N \rightarrow \infty$ , YES.

DEF: A DENSITY  $p(\underline{x} | \underline{\theta})$  IS IDENTIFIABLE  
 IF  $p(\underline{x} | \underline{\theta}_1) = p(\underline{x} | \underline{\theta}_2) \forall \underline{x}$   
 $\Rightarrow \underline{\theta}_1 = \underline{\theta}_2$  (UP TO A PERMUTATION OF  
 MIXTURE COMPONENT INDICES)\*

IN PRACTICE, IF  $x_j \in \mathbb{R}$ , AND HAVE  
 A SUFFICIENT NUMBER OF DATA POINTS,  
 IDENTIFIABILITY IS USUALLY SATISFIED.

There are really 2 key advantages to using GMMs.

Firstly GMMs are a lot more flexible in terms of cluster covariance than K-Means; due to the standard deviation parameter, the clusters can take on any ellipse shape, rather than being restricted to circles. K-Means is actually a special case of GMM in which each cluster's covariance along all dimensions approaches 0.

Secondly, since GMMs use probabilities, they can have multiple clusters per data point. So if a data point is in the middle of two overlapping clusters, we can simply define its class by saying it belongs X-percent to class 1 and Y-percent to class 2. I.e GMMs support mixed membership.

## Clustering

**Connectivity models (Hierarchy Clustering):** As the name suggests, these models are based on the notion that the data points closer in data space exhibit more similarity to each other than the data points lying farther away. These models can follow two approaches. In the first approach, they start with classifying all data points into separate clusters & then aggregating them as the distance decreases. In the second approach, all data points are classified as a single cluster and then partitioned as the distance increases. Also, the choice of distance function is subjective. These models are very easy to interpret but lacks scalability for handling big datasets. Examples of these models are hierarchical clustering algorithm and its variants.

**Centroid models (K means):** These are iterative clustering algorithms in which the notion of similarity is derived by the closeness of a data point to the centroid of the clusters. K-Means clustering algorithm is a popular algorithm that falls into this category. In these models, the no. of clusters required at the end have to be mentioned beforehand, which makes it important to have prior knowledge of the dataset. These models run iteratively to find the local optima.

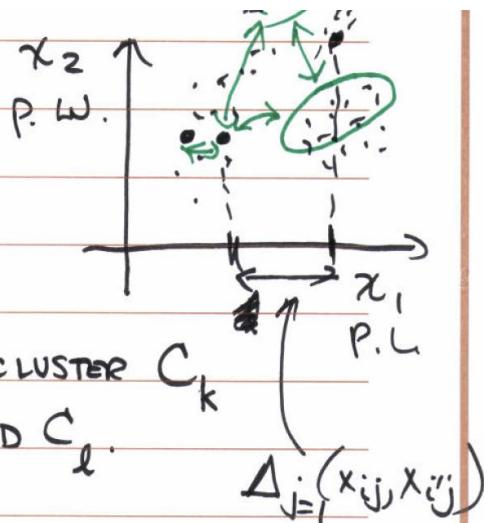
**Distribution models (EM Algorithm):** These clustering models are based on the notion of how probable is it that all data points in the cluster belong to the same distribution (For example: Normal, Gaussian). These models often suffer from overfitting. A popular example of these models is Expectation-maximization algorithm which uses multivariate normal distributions.

**Density Models (DBSCAN):** These models search the data space for areas of varied density of data points in the data space. It isolates various different density regions and assign the data points within these regions in the same cluster. Popular examples of density models are DBSCAN and OPTICS.

## Similarity/ Dissimilarity Measure

NEED MEASURES FOR  
SIMILARITY / DISSIM.  
BETWEEN:

- 2 POINTS  $x_i$  AND  $x_j$
- A POINT  $x_i$  AND A CLUSTER  $C_k$
- TWO CLUSTERS  $C_k$  AND  $C_l$ .



ALSO: MEASURE FOR QUALITY OF A CLUSTERING

Euclidean Distance | Manhattan Distance | Hamming Distance

2	1	0 (# of different features)
---	---	-----------------------------

LET  $\Delta(x_i, x_{i'}) = \Delta_{ii'} = d_{ii'}$   
DENOTE A DISSIMILARITY FUNCTION

CAN USE A DISTANCE FUNCTION, E.G.:

$$\begin{aligned} \Delta(x_i, x_{i'}) &= d_{ii'}(x_i, x_{i'}) \\ &= \sum_{j=1}^D \Delta_j(x_{ij}, x_{i'j}) \end{aligned}$$

$\Delta_j(x_{ij}, x_{i'j})$  CAN BE:

$$\frac{(x_{ij} - x_{i'j})^2}{|x_{ij} - x_{i'j}|} \quad (\text{EUCL. DIST.}^2)$$

(L<sub>1</sub> NORM DIST. OR  
CITY-BLOCK DIST. OR  
MANHATTAN DIST.)

FOR NOMINAL FEATURES (SYMBOLIC, CATEGORICAL, OR  
LABELS):

$\Delta(x_{ij}, x_{i'j}) = \# \text{ OF FEATURES THAT}$   
ARE DIFFERENT

$$= \sum_{j=1}^p \mathbb{I}(x_{ij} \neq x_{i'j})$$

= HAMMING DISTANCE.

CAN LET  $\Delta(x_i, x_{i'}) = \text{THIS OR OTHER } d_{ii'}$ ,  
s.t.  $d_{ii'} \geq 0 \quad \forall i, i'$ , AND  $d_{ii} = 0 \quad \forall i$

OR  $f(d_{ii'})$ ,  $f = \text{ANY MONOTONICALLY}$   
INCREASING FUNCTION.

LET  $s(\underline{x}_i, \underline{x}_{i'}) = s_{ii'}$  DENOTE A SIMILARITY Fcn.

CAN CHOOSE  $s_{ii'} = g[d_{ii'}]$ ,  $g$  = ANY MONOTONICALLY DECREASING FUNCTION.

E.G.:  $s(\underline{x}_i, \underline{x}_{i'}) = (\max_{l,l'} d_{ll'}) - d_{ii'}$

OR  $= \exp\left\{-\frac{d_{ii'}}{\sigma^2}\right\}$

similarity : when near 1 (large) --> two data point are almost same

FOR BINARY FEATURES

(e.g., CONTAINS AN ATTRIBUTE OR NOT) :

$$s(\underline{x}_i, \underline{x}_{i'}) = \frac{\underline{x}_i^\top \underline{x}_{i'}}{\underline{x}_i^\top \underline{x}_i + \underline{x}_{i'}^\top \underline{x}_{i'} - \underline{x}_i^\top \underline{x}_{i'}}$$

= % OF ATTRIBUTES THAT  
ARE SHARED.

CORRELATION COEFFICIENT , E.G. (PEARSON) :

LET  $j$  BE INDEX OVER SPATIAL LOCATION

OR TIME

$$r_{ii'} = \frac{\sum_{j=1}^D (x_{ij} - \bar{x}_i)(x_{i'j} - \bar{x}_{i'})}{\left[ \sum_j (x_{ij} - \bar{x}_i)^2 \sum_j (x_{i'j} - \bar{x}_{i'})^2 \right]^{1/2}}$$

IN WHICH:

$$\bar{x}_i = \frac{1}{D} \sum_{j=1}^D x_{ij}. \text{ NOTE: } -1 \leq r_{ii} \leq 1 \text{ ALWAYS.}$$

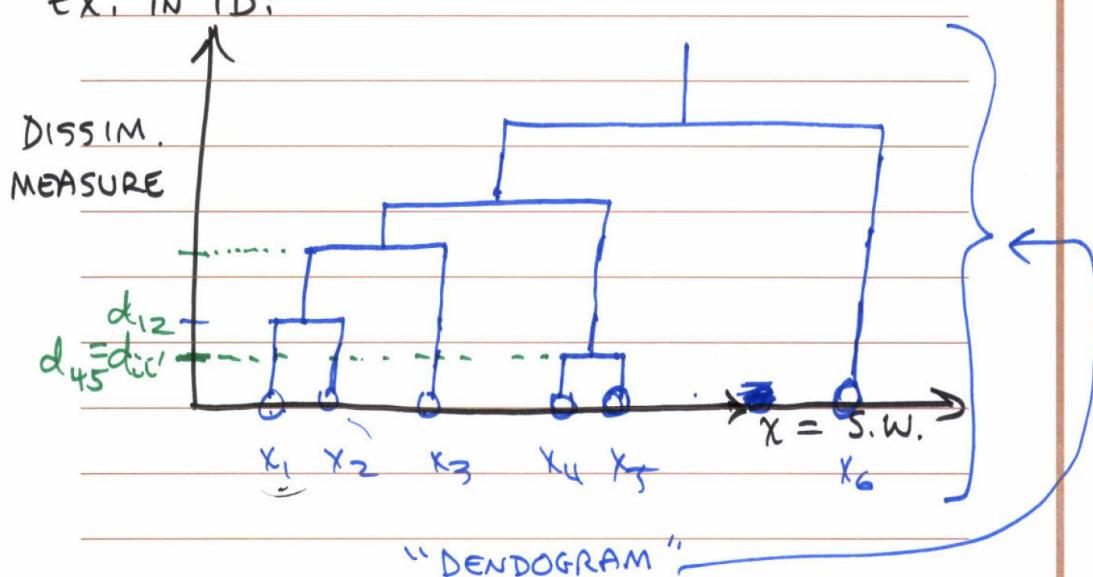
CAN LET  $S_{ii'} = r_{ii'}$ . FOR SIMILARITY.

$$\text{OR: } d_{ii'} = \frac{1 - r_{ii'}}{2}$$

### Hierarchical Clustering

find the nearest one and combine together as a new node. Then expand

EX. IN 1D:



TWO BASIC APPROACHES:

- AGGLOMERATIVE (BOTTOM UP).
- DIVISIVE (TOP DOWN)

→ UNDERLYING ASSUMPTION: IF 2 DATA POINTS ARE IN THE SAME CLUSTER AT ONE LEVEL, THEN THEY ARE IN THE SAME CLUSTER AT ALL HIGHER LEVELS.

## Agglomerative

Bottom up: find the most similar two, combine change from N clusters to few clusters

### AGGLOMERATIVE HIERARCHICAL CLUSTERING

#### PROCEDURE

LET  $\delta_{jk}$  = DISTANCE OR DISSIMILARITY BETWEEN CLUSTERS  $C_j$  AND  $C_k$ ,  
 $\hat{K}$  = CURRENT # OF CLUSTERS.

1. CHOOSE HALTING CONDITION (H.C.)
2. INITIALIZE  $\hat{K} = N$ , CLUSTER  $C_i = \{x_i\}$ ,  
 $i=1, 2, \dots, N$ ; ITERATION  $m = 1$ .
3. REPEAT UNTIL H.C. IS MET:
  4. FIND NEAREST (LEAST DISSIMILAR)

PAIR OF CLUSTERS:

$$j', k' = \underset{j, k}{\operatorname{arg\,min}} \delta_{jk}, \text{ AND } \delta' = \underset{j, k}{\min} \delta_{jk}$$

(RESOLVE ~~TIES~~ TIES RANDOMLY)

5. OPTIONAL OUTPUT  $m, \hat{K}, \delta', j', k'$
6. IF H.C. IS BASED ON  $\delta'$ , TEST FOR IT (HALT IF TRUE) ( $\rightarrow$  e.g.,  $\delta' \geq \delta_{\text{halt}}$ )
7. MERGE CLUSTERS  $C_{j'}$  AND  $C_{k'}$  TO FORM A NEW CLUSTER  $C_l$   
[APPLY MERGE RULE]

8. UPDATE  $\hat{K} \leftarrow \hat{K}-1$ , ITERATION  $m \leftarrow m+1$

9. IF H.C. IS BASED ON  $\hat{K}$ , TEST FOR IT  
(HALT IF TRUE).

10. OUTPUT FINAL CLUSTERS  $C_l$ ,  $l=1, \dots, \hat{K}_{\text{FINAL}}$ ;  
 $\hat{K}_{\text{FINAL}}$ ; DISSIMILARITY  $\delta(m)$ .

IF  $\hat{K}_{\text{FINAL}} = 1$ , THE RESULTING HIERARCHY IS  
A DENDOGRAM.

USEFUL DISTANCE OR DISSIMILARITY MEASURES:

(BETWEEN CLUSTERS  $C_k, C_\ell$ ):

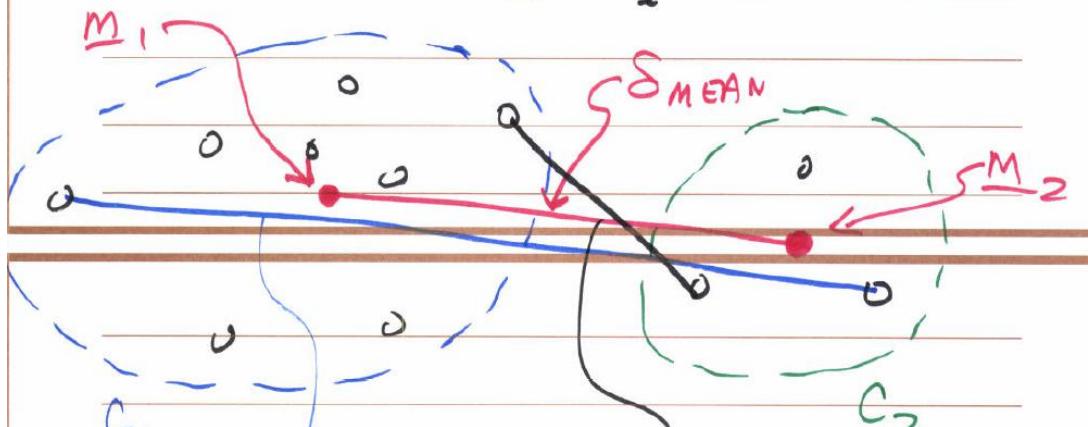
$$\delta_{\text{MEAN}}(C_k, C_\ell) \triangleq \| \bar{m}_k - \bar{m}_\ell \|_2$$

$$\delta_{\text{AVG}}(C_k, C_\ell) \triangleq \frac{1}{N_k N_\ell} \sum_{x \in C_k} \sum_{x' \in C_\ell} \| x - x' \|_2$$

(MEAN OF DISTANCES BETWEEN ALL  
PAIRS OF PTS. ONE FROM EACH CLUSTER).

$$\delta_{\min}(C_k, C_l) = \min_{\substack{x \in C_k \\ x' \in C_l}} \|x - x'\|_2$$

$$\delta_{\max}(C_k, C_l) = \max_{\substack{x \in C_k \\ x' \in C_l}} \|x - x'\|_2$$



Nearest Neighbor

CHARACTERISTICS:

RESULTING GRAPH IS A TREE (NO CLOSED LOOPS)

CONTINUE TO  $K_{\text{FINAL}} = 1 \Rightarrow \underline{\text{SPANNING TREE}}$

USE  $\delta_{\min} \Rightarrow \underline{\text{MINIMAL SPANNING TREE}}$

4

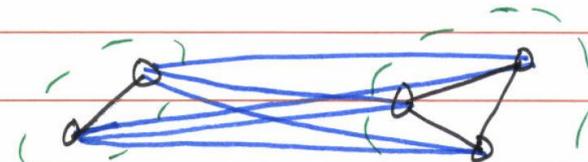
(A SPANNING TREE THAT HAS MIN. TOTAL LENGTH OF ALL EDGES).

Farthest Neighbor

Use  $\delta_{\max}$ .

MERGE RULE: CONNECT ALL NODES IN ONE CLUSTER TO ALL NODES IN OTHER CLUSTER.

$\Rightarrow$  EACH CLUSTER IS REPRESENTED BY A FULLY CONNECTED SUBGRAPH.



CHARACTERISTICS:

DEF DIAMETER OF A CLUSTER: LENGTH OF THE LONGEST EDGE.

DIAMETER OF A CLUSTERING (PARTITIONING): LENGTH OF LONGEST EDGE OVER ALL CLUSTERS.

F.N. ALG., AT EACH STEP, MERGES THE 2 CLUSTERS THAT GIVE THE SMALLEST

INCREASE IN DIAMETER OF THE CLUSTERING.

How to choose K in Clustering

1. FOR AGGLOMERATIVE HIERARCHICAL CLUSTERING, CAN USE THE STEPWISE CRITERION:

$$\text{Ex: } \delta'(m) = \min_{j,k} \delta_{jk}$$

IN WHICH FOR NN/SINGLE LINKAGE:

$$\delta_{jk} = \delta_{\min}(C_j, C_k)$$

cluster quality measurement

2. FOR GRAPHICAL METHODS GENERALLY

(OR ANY CLUSTERING METHOD), CAN USE  
OTHER AD HOC MEASURES

CALINSKI AND HARABASZ INDEX:

$$CH(k) \triangleq \frac{\left( \sum_{k=1}^K N_k \| \underline{m}_k - \underline{m} \|_2^2 \right) / (K-1)}{\left( \sum_{k=1}^K \sum_{\underline{x}_i \in C_k} \| \underline{x}_i - \underline{m}_k \|_2^2 \right) / (N-K)}$$

UN-NORM'D SAMPLE VARIANCE OF CLUSTER  $C_k$

~ SAMPLE VARIANCE OF CLUSTER MEANS

SAMPLE WITHIN CLUSTER VARIANCE

$$\text{in which: } N_k = \# \text{DATA PTS. IN } C_k$$

$$\underline{m}_k \triangleq \frac{1}{N_k} \sum_{\underline{x}_i \in C_k} \underline{x}_i$$

$$\underline{m} \triangleq \frac{1}{N} \sum_{i=1}^N \underline{x}_i.$$

$$\text{LET } \hat{k} = \underset{k}{\operatorname{argmax}} CH(k)$$

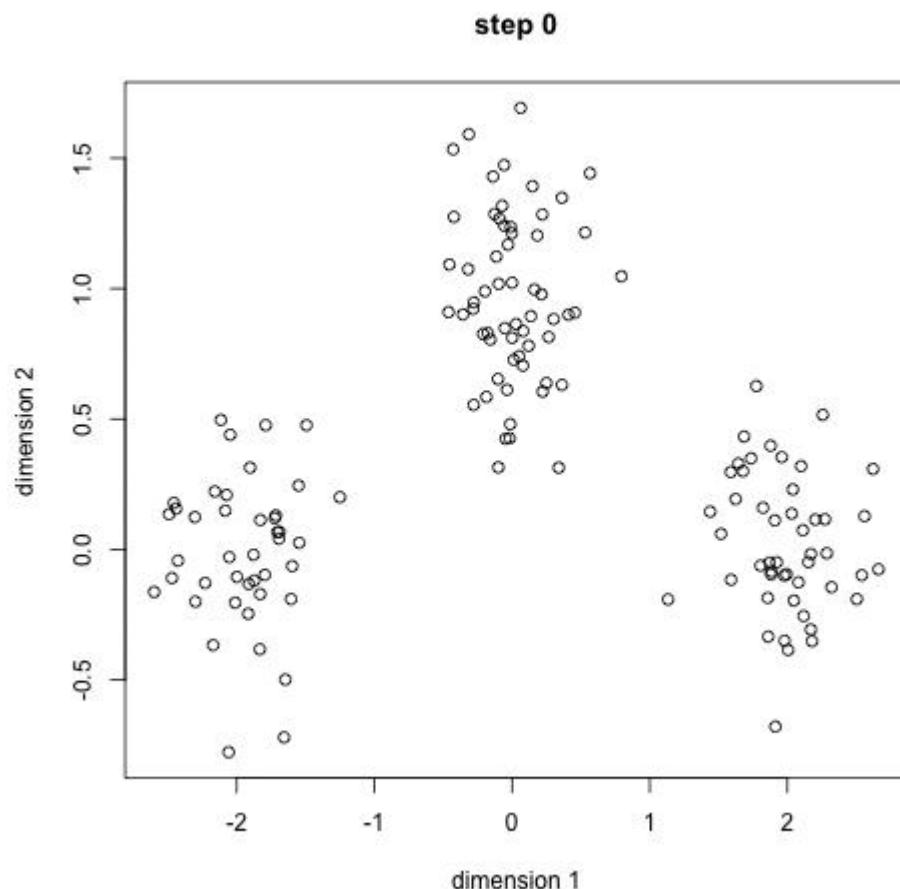
$CH(k)$  WAS FOUND THE BEST IN A SYSTEMATIC

COMPARISON OF 30 AD HOC CLUSTER QUALITY MEASURES.

Divisive

Top down:

K Means Clustering



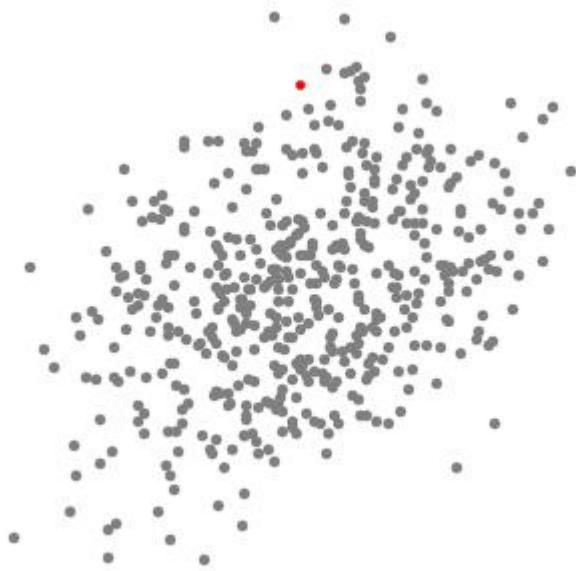
To begin, we first select a number of classes/groups to use and randomly initialize their respective center points. To figure out the number of classes to use, it's good to take a quick look at the data and try to identify any distinct groupings. The center points are vectors of the same length as each data point vector and are the "X's" in the graphic above.

Each data point is classified by computing the distance between that point and each group center, and then classifying the point to be in the group whose center is closest to it.

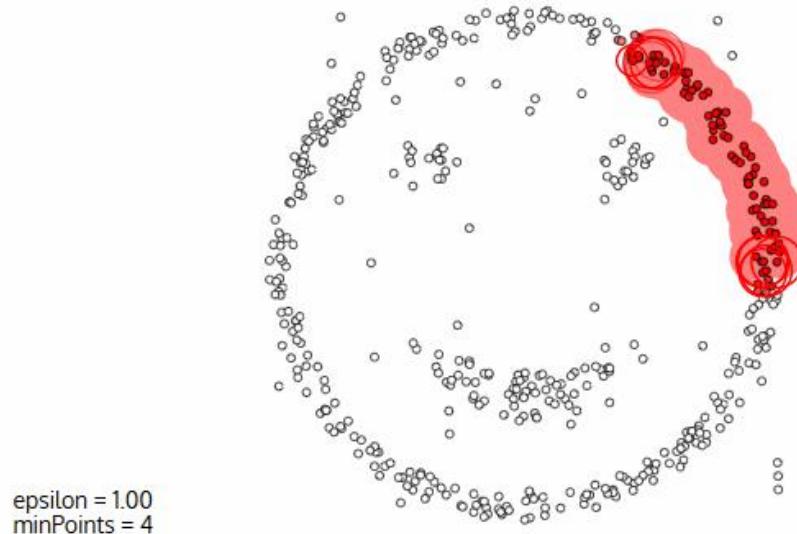
Based on these classified points, we recompute the group center by taking the mean of all the vectors in the group.

Repeat these steps for a set number of iterations or until the group centers don't change much between iterations. You can also opt to randomly initialize the group centers a few times, and then select the run that looks like it provided the best results.

## Mean Shift



## DBSCAN (Density-Based Spatial Clustering of Applications with Noise)



**Restart**



**Pause**

DBSCAN is a density based clustered algorithm similar to mean-shift, but with a couple of notable advantages.

DBSCAN begins with an arbitrary starting data point that has not been visited. The neighborhood of this point is extracted using a distance  $\epsilon$  (All points which are within the  $\epsilon$  distance are neighborhood points).

If there are a sufficient number of points (according to  $\text{minPoints}$ ) within this neighborhood then the clustering process starts and the current data point becomes the first point in the new cluster. Otherwise, the point will be labeled as noise (later this noisy point might become the part of the cluster). In both cases that point is marked as "visited".

For this first point in the new cluster, the points within its  $\epsilon$  distance neighborhood also become part of the same cluster. This procedure of making all points in the  $\epsilon$  neighborhood belong to the same cluster is then repeated for all of the new points that have been just added to the cluster group.

This process of steps 2 and 3 is repeated until all points in the cluster are determined i.e all points within the  $\epsilon$  neighborhood of the cluster have been visited and labelled.

Once we're done with the current cluster, a new unvisited point is retrieved and processed, leading to the discovery of a further cluster or noise. This process repeats until all points are marked as visited. Since at the end of this all points have been visited, each point will have been marked as either belonging to a cluster or being noise.

DBSCAN poses some great advantages over other clustering algorithms. Firstly, it does not require a pre-set number of clusters at all. It also identifies outliers as noises unlike mean-shift which simply throws them into a cluster even if the data point is very different. Additionally, it is able to find arbitrarily sized and arbitrarily shaped clusters quite well.

The main drawback of DBSCAN is that it doesn't perform as well as others when the clusters are of varying density. This is because the setting of the distance threshold  $\epsilon$  and minPoints for identifying the neighborhood points will vary from cluster to cluster when the density varies. This drawback also occurs with very high-dimensional data since again the distance threshold  $\epsilon$  becomes challenging to estimate.

## Clustering Measurement

The term clustering validation is used to design the procedure of evaluating the results of a clustering algorithm.

The *silhouette plot* is one of the many measures for inspecting and validating clustering results.

Recall that the silhouette ( $S_i$ ) measures how similar an object  $i$  is to the other objects in its own cluster versus those in the neighbor cluster.  $S_i$  values range from 1 to -1:

A value of  $S_i$  close to 1 indicates that the object is well clustered. In other words, the object  $i$  is similar to the other objects in its group.

A value of  $S_i$  close to -1 indicates that the object is poorly clustered, and that assignment to some other cluster would probably improve the overall results.

Assume the data have been clustered via any technique, such as k-means, into  $k$  clusters.

For each data point  $i \in C_i$  (data point  $i$  in the cluster  $C_i$ ), let

$$a(i) = \frac{1}{|C_i| - 1} \sum_{j \in C_i, i \neq j} d(i, j)$$

be the mean distance between  $i$  and all other data points in the same cluster, where  $d(i, j)$  is the distance between data points  $i$  and  $j$  in the cluster  $C_i$  (we divide by  $|C_i| - 1$  because we do not include the distance  $d(i, i)$  in the sum). We can interpret  $a(i)$  as a measure of how well  $i$  is assigned to its cluster (the smaller the value, the better the assignment).

We then define the mean dissimilarity of point  $i$  to some cluster  $C$  as the mean of the distance from  $i$  to all points in  $C$  (where  $C \neq C_i$ ).

For each data point  $i \in C_i$ , we now define

$$b(i) = \min_{k \neq i} \frac{1}{|C_k|} \sum_{j \in C_k} d(i, j)$$

to be the *smallest* (hence the `min` operator in the formula) mean distance of  $i$  to all points in any other cluster, of which  $i$  is not a member. The cluster with this smallest mean dissimilarity is said to be the "neighboring cluster" of  $i$  because it is the next best fit cluster for point  $i$ .

We now define a *silhouette* (value) of one data point  $i$

$$s(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}}, \text{ if } |C_i| > 1$$

and

$$s(i) = 0, \text{ if } |C_i| = 1$$