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

EE559 Pattern Recognition

1.Introduction

basic concept

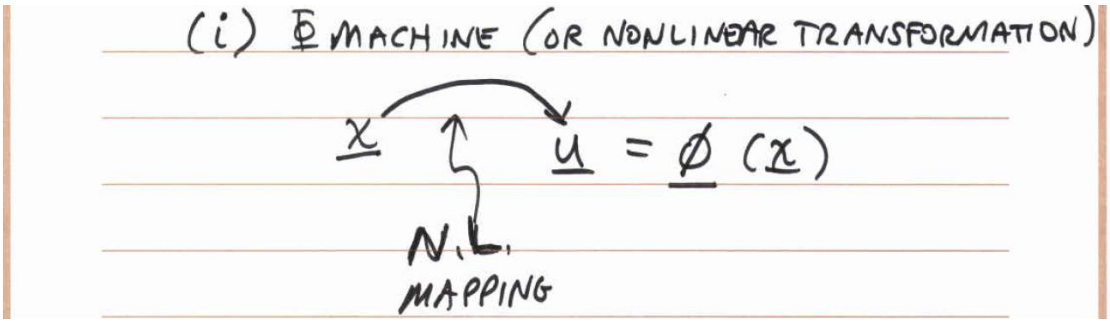
Training/Classification procedure

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

feature space, discriminant function

totally seperated, pairwise seperated

augment space

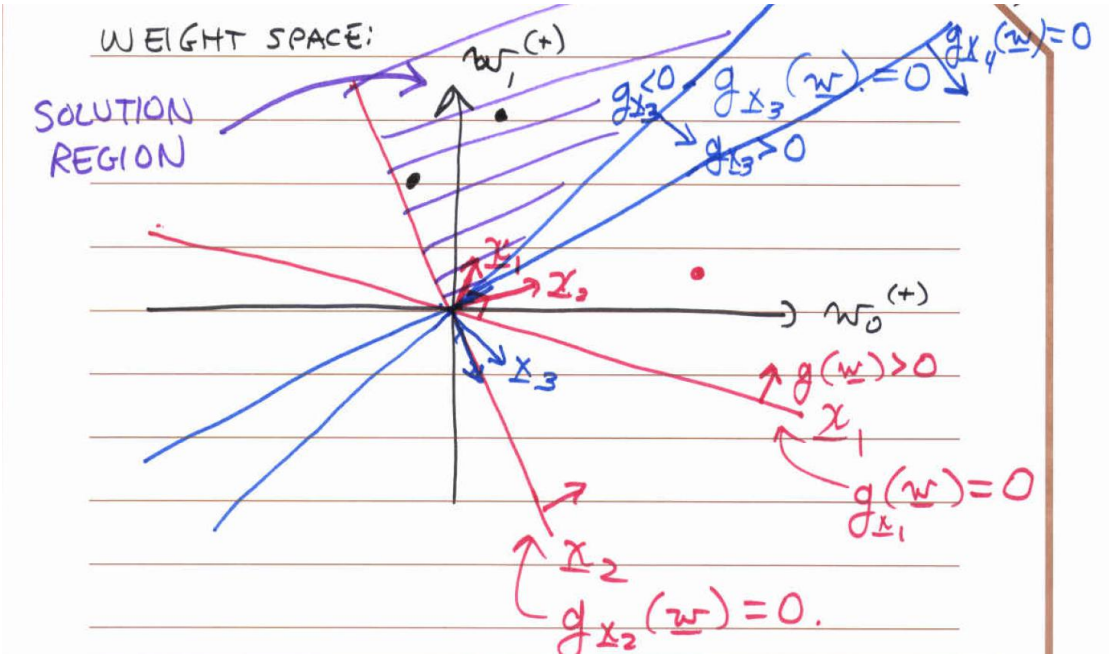


weight space

descriminate function: g(x)

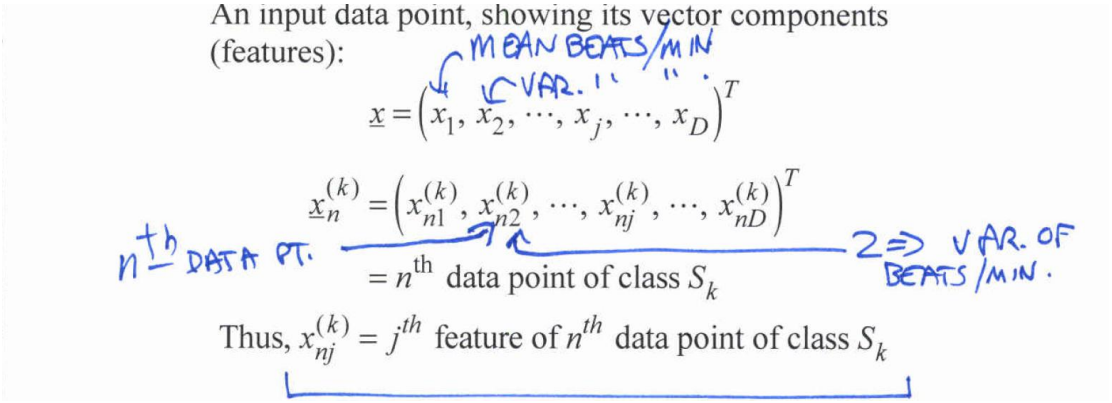
criterion functions: J(w) (cost function is one kind of criterion function)

each line means a pair of weight



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

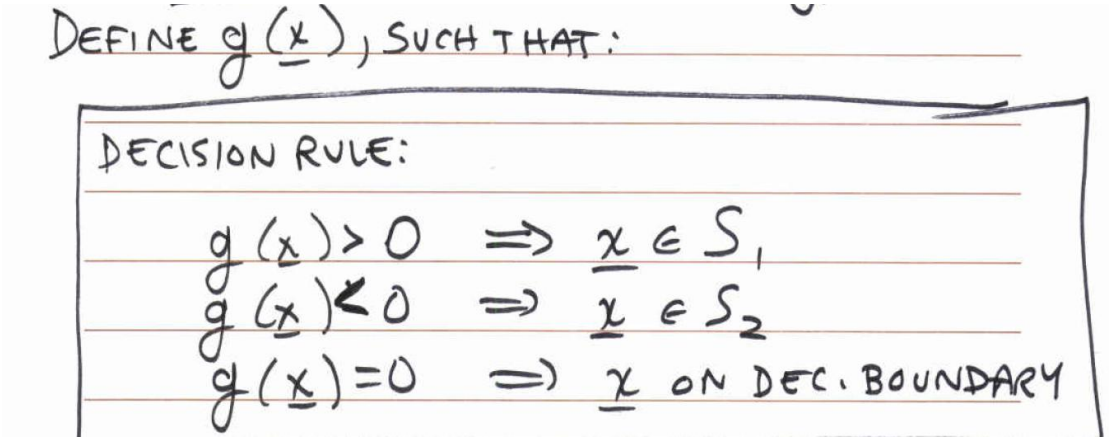
features



Discriminant function:

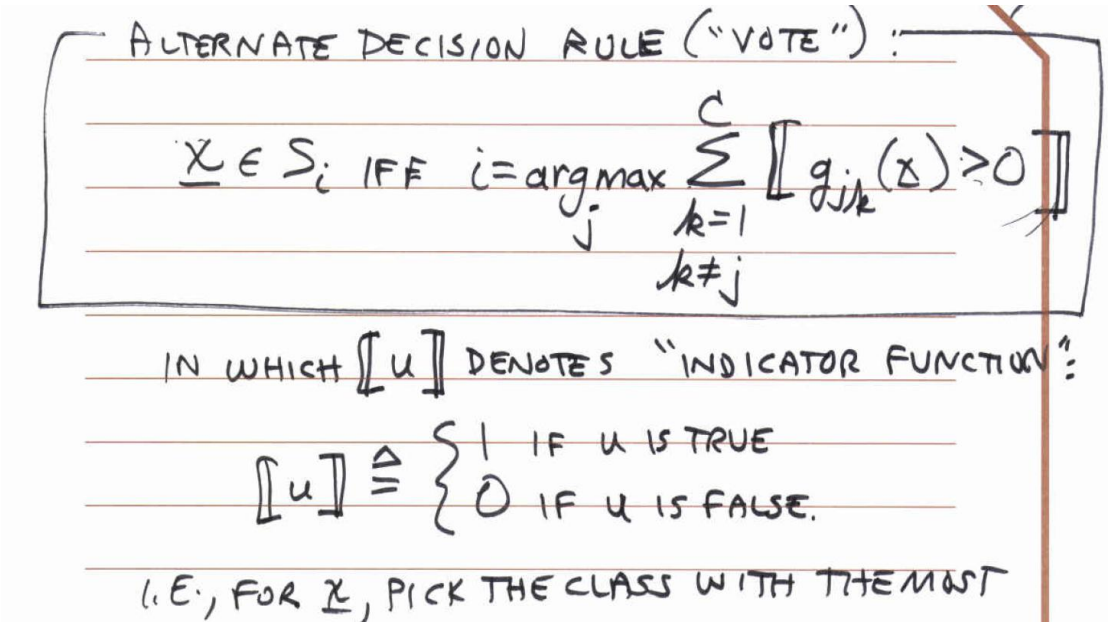
used to discriminate (> or < 0) class (not minimize)

two class:

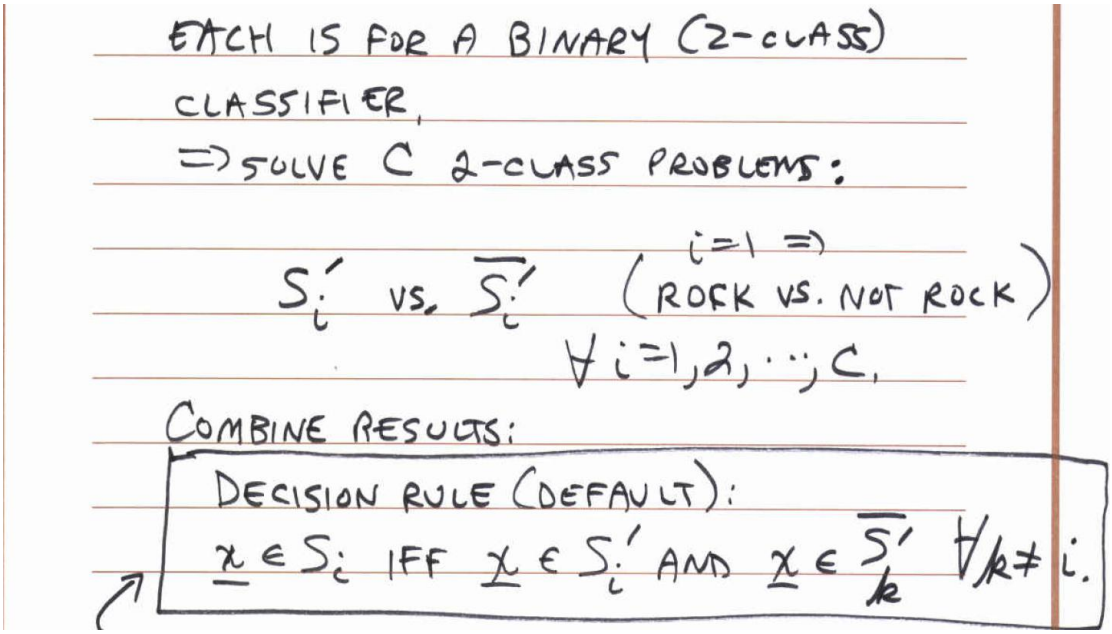


multi class:

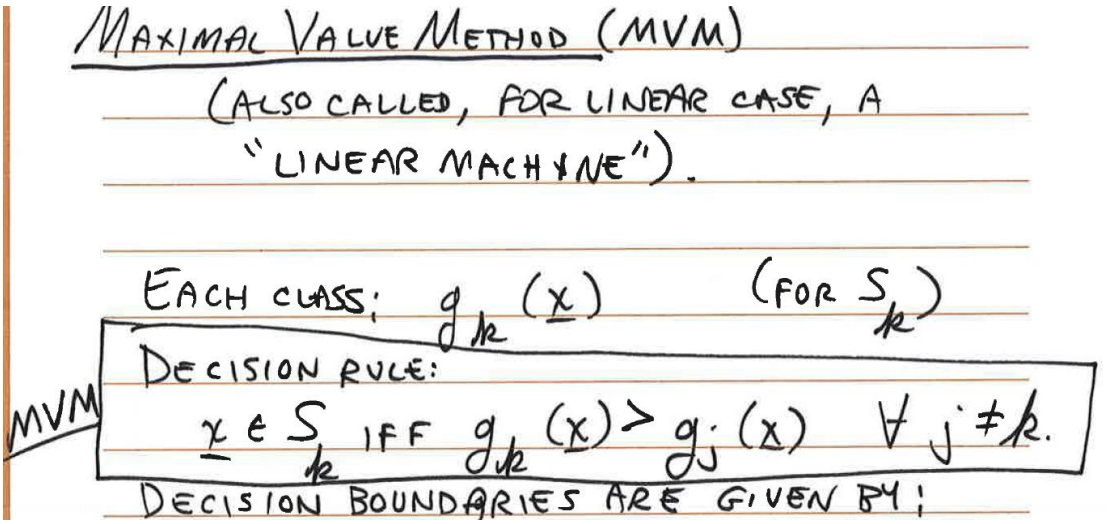
one vs one



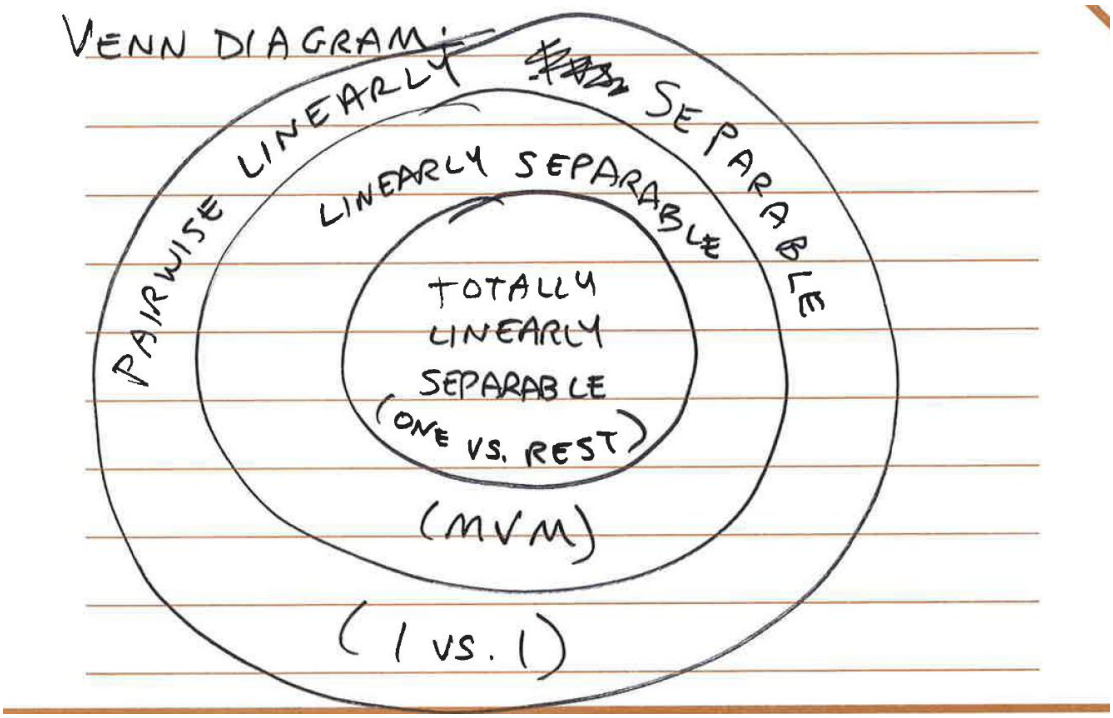
one vs rest



Maximum Value Method



overall:
totally linear seperate: a line can seperate one class with all the rest
linear seperate: a line can seperate one class with other class
pairwise seperate: some lines can seperate one class with other class



Criterion Function
loss function is one way of criterion function
criterion function can 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.

EX: $g(\underline{x}) = \underline{w}^T \underline{x}$ 2-class.

d.o.f. = # WEIGHT VALUES

$= D+1$.

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

e.g.: $\underline{X}\,\underline{w} = \underline{b}$ IS A

CAN BE FORMALIZED MATHEMATICALLY "VC DIMENSION" [VAPNIK-CHERVONENKIS], WHICH QUANTIFIES THE "FLEXIBILITY" OR "CAPACITY" OF A DECISION BOUNDARY.

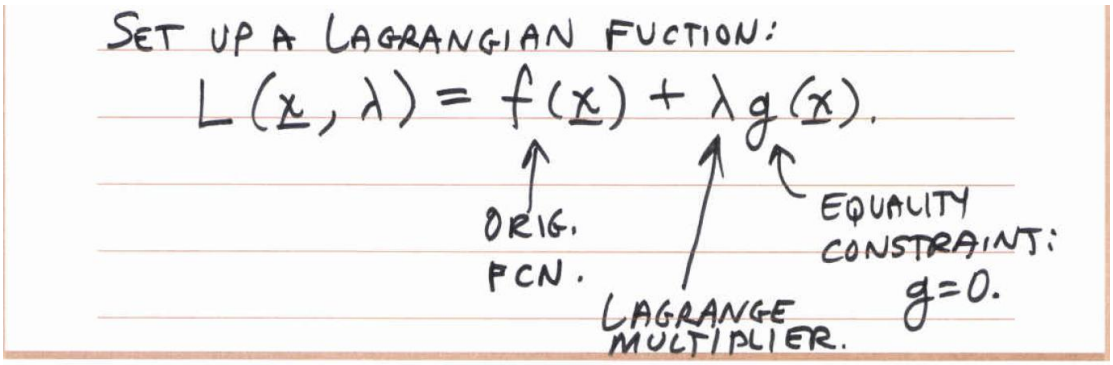
EXAMPLES:

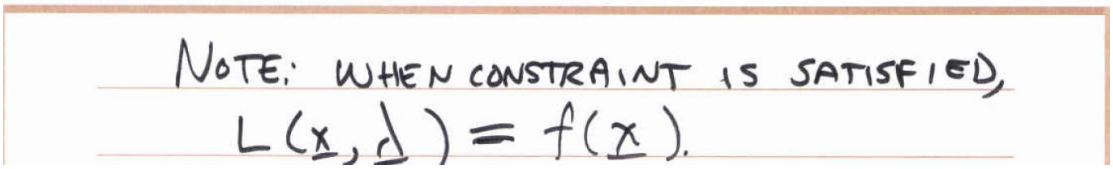
- LINEAR BOUNDARY IN (D+1)-SPACE [D FEATURES]

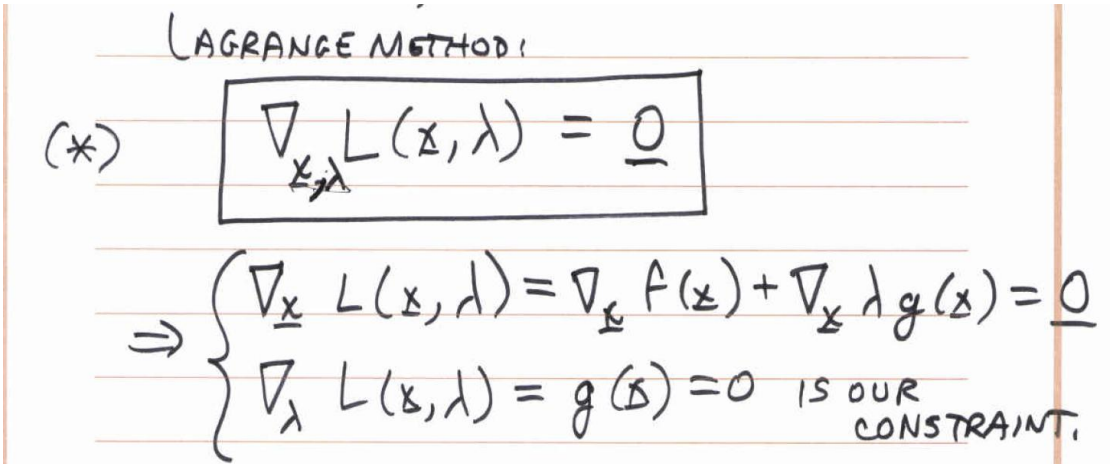
  VCdim = D+1

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

Lagrange Optimization (Basic for SVM)



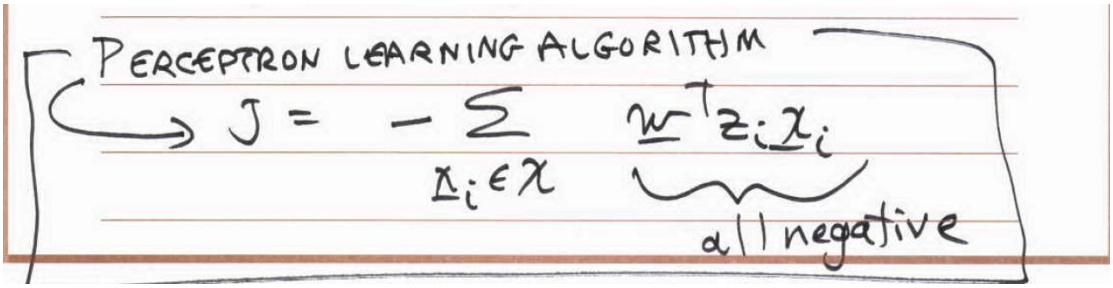




Feature selection

2.Distribution-Free Classification Methods

Perceptron



MINIMIZE J BY GRADIENT DESCENT.

$$\nabla_{\underline{w}} J(\underline{w}) = -\sum_{\underline{x}_n \in \mathcal{X}} z_n \underline{x}_n$$

$$\underline{w}(i+1) = \underline{w}(i) + \eta(i) \sum_{\underline{x}_n \in \mathcal{X}} z_n \underline{x}_n$$

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

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

STOP WHEN $\mathcal{X} = \emptyset$ $[\underline{w}(i+1) = \underline{w}(i)]$

PERCEPTRON LEARNING USING BATCH GRADIENT DESCENT (GD) (OR BATCH UPDATE)

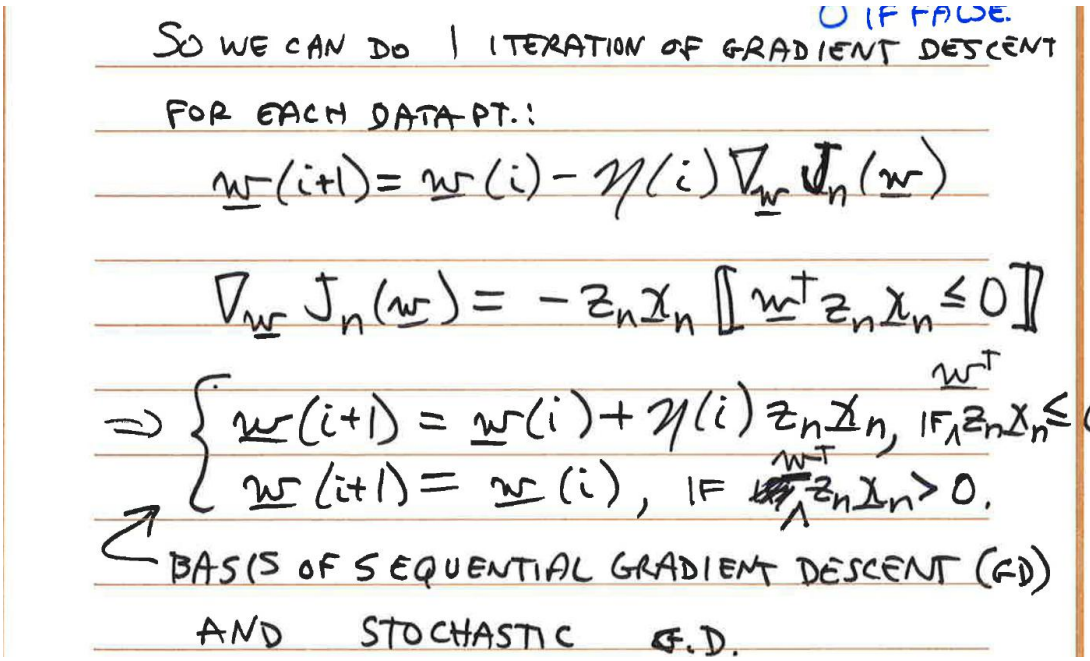
WE CAN INSTEAD UPDATE $\underline{w}$ AFTER EACH DATA POINT.

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

$J_n(\underline{w})$: crit. fcn. based on 1 training pt. $(\underline{x}_n)$

$$J_n(\underline{w}) = -\underline{w}^T z_n \underline{x}_n [\![\underline{w}^T z_n \underline{x}_n \leq 0]\!]$$

$[\![\cdot]\!]$: 1 IF TRUE



batch gradient descent

Iterate until convergence

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

points misclassified by $\mathsf{H}_{\underline{w}(i)}$, and $\eta(i) \ge 0$*.

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 $\{1, 2, \cdots, N\}$

$i = n - 1$

$$\begin{cases} \underline{w}(i+1) = \underline{w}(i) + \eta(i) z_n \underline{x}_n & \text{if } z_n \underline{x}_n \le 0 \\ \underline{w}(i+1) = \underline{w}(i) & \text{if } z_n \underline{x}_n > 0 \end{cases}$$

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)

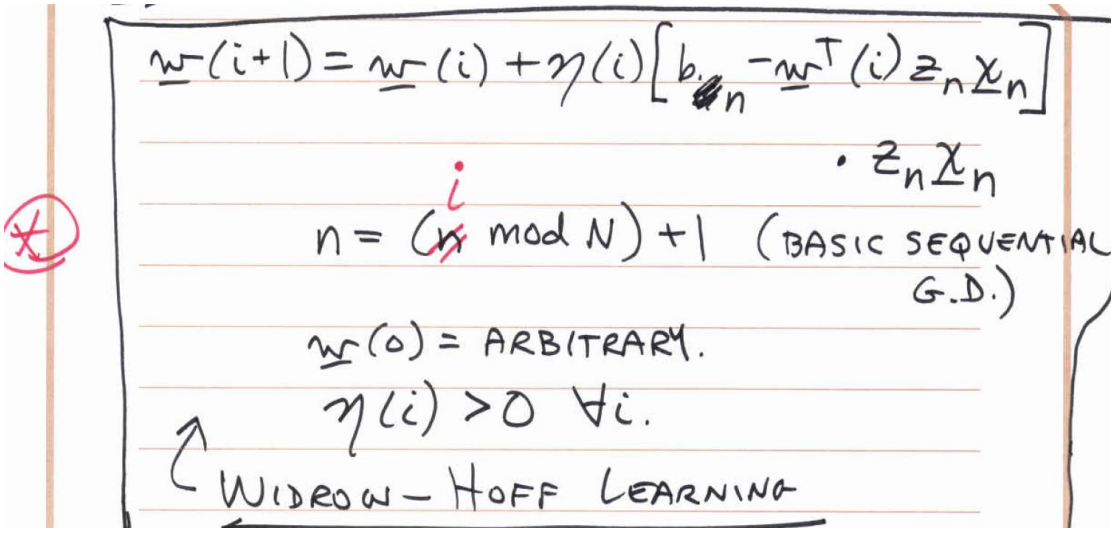
Iterate until convergence:

(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

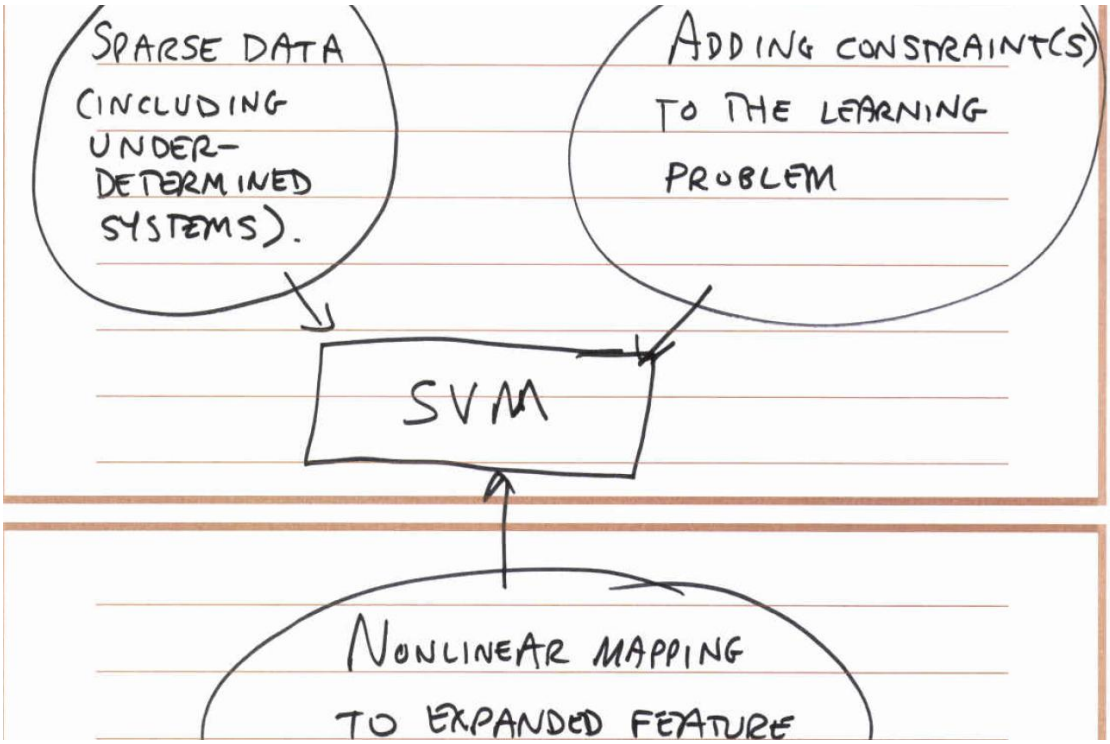
if use MSE (1,0) rather than misclassified data, linear regression



SVM

aim: maximize the margin

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



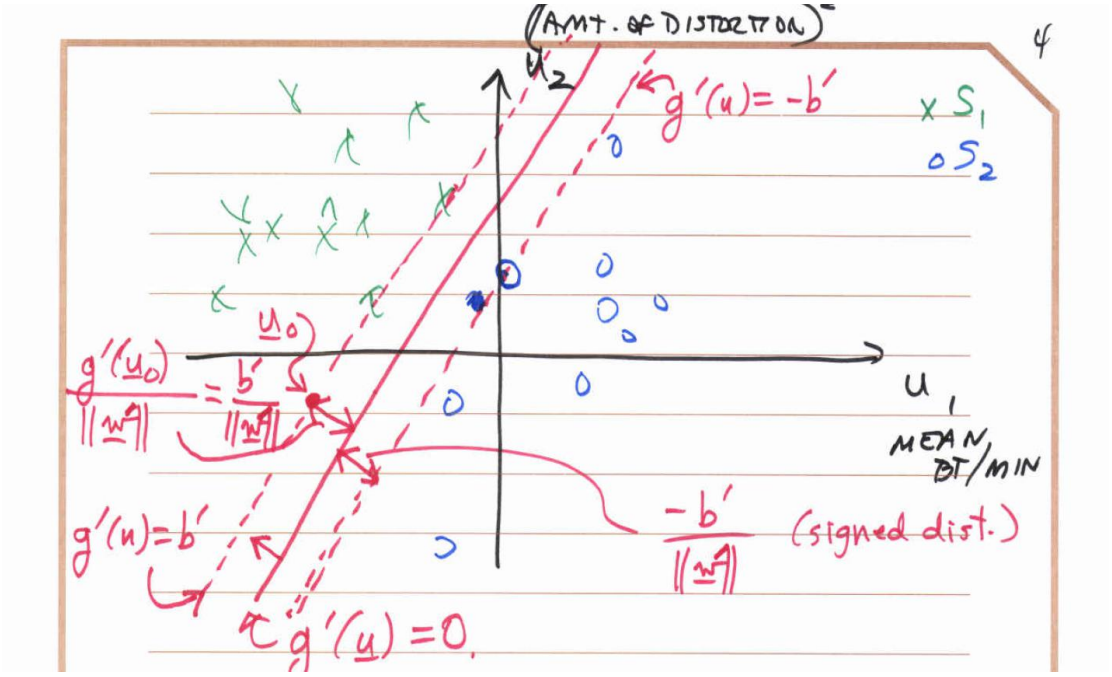
(ii) OPTIMIZED VERSION OF USING A (SAFETY) MARGIN.

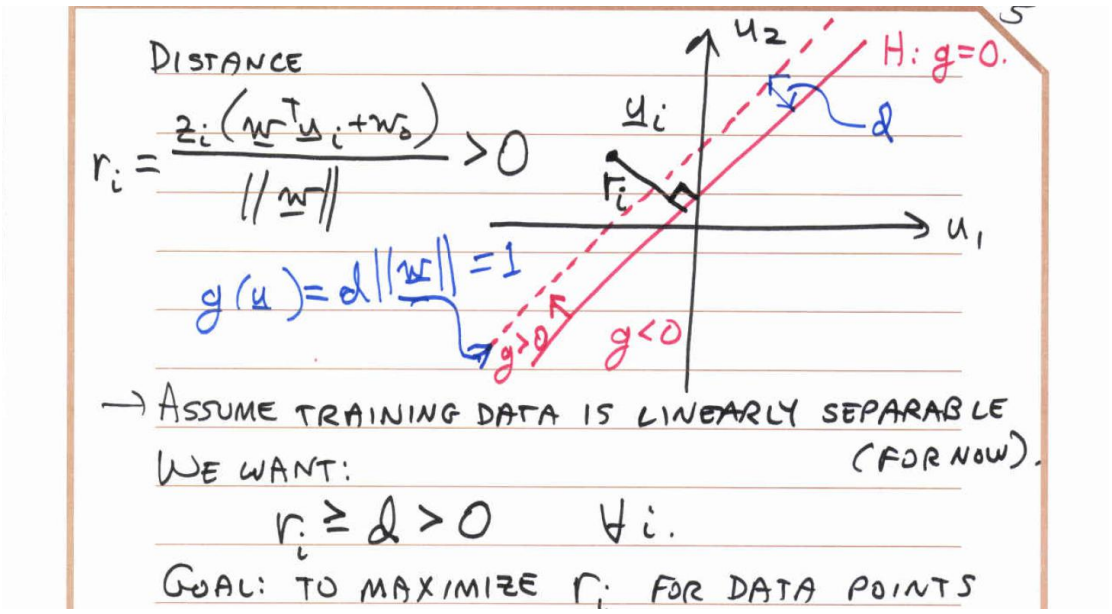
$$g(\underline{x}) = \underline{w}^T \underline{x} + w_0 \geq b > 0 \quad \text{FOR } \underline{x} \in S_1.$$

$$\Downarrow$$

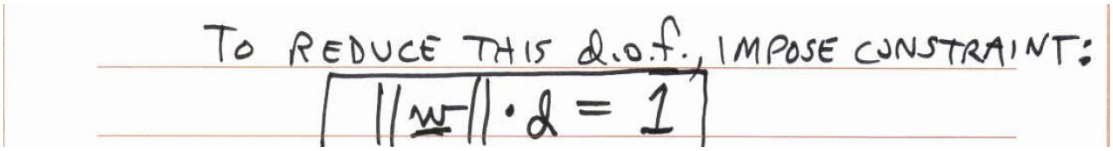
$$g'(\underline{u}) = \underline{w}'^T \underline{u} + w_0' \geq b' > 0 \quad \text{FOR } \underline{u} \in S_1.$$

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

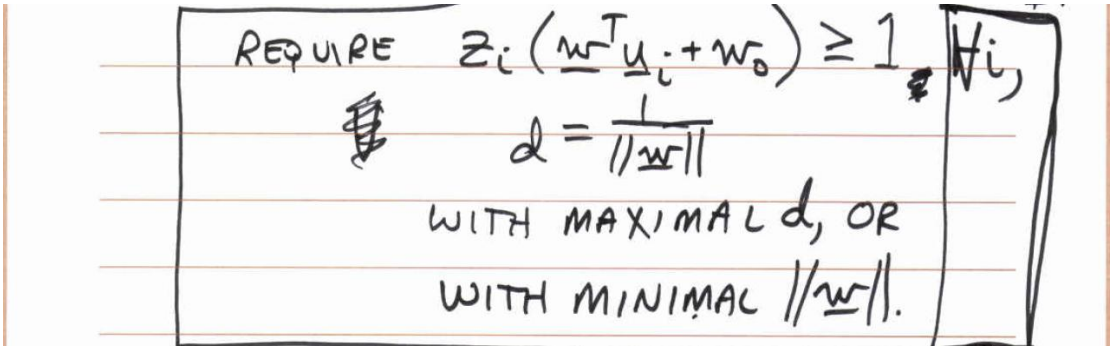


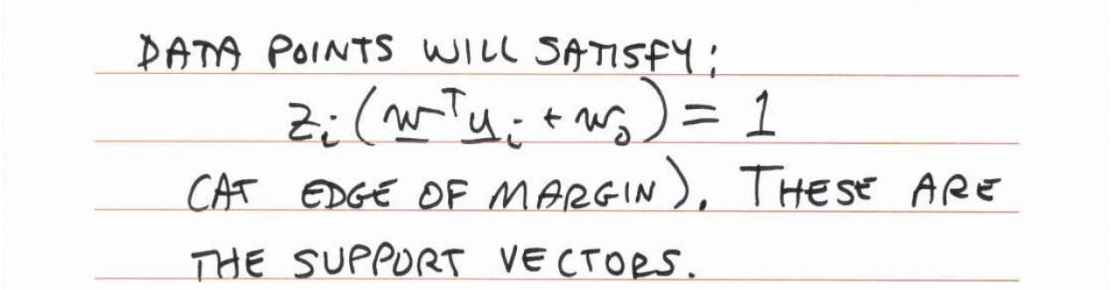


to reduce complexity, add a constrain, what we don't need to define d (minimal distance)



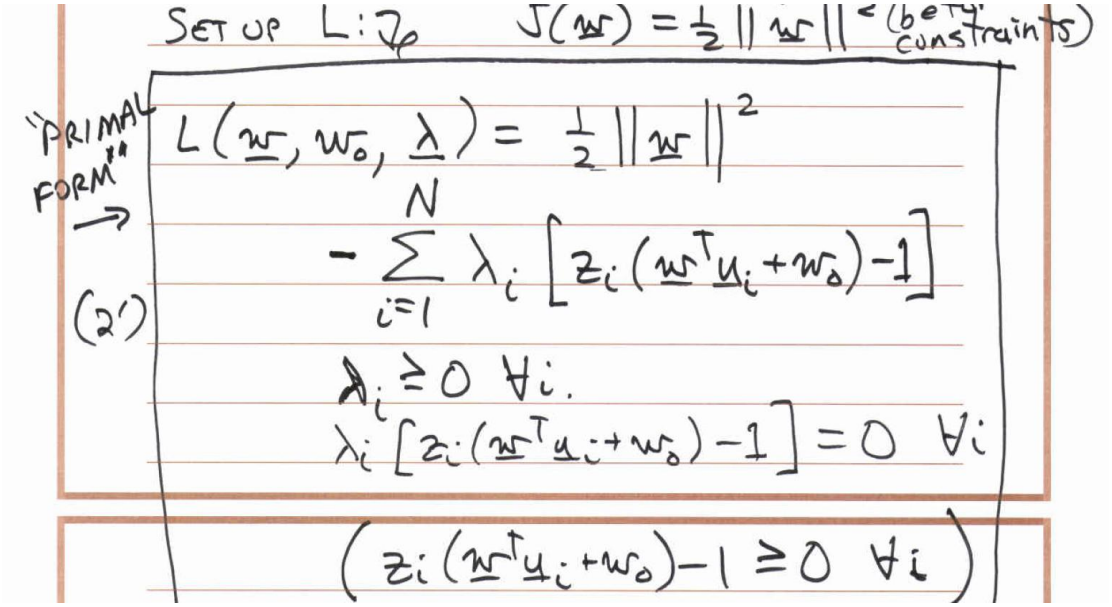
training method:





SVM learning

transfer minimal ||w|| with requirement Zi(WtUi +W0)>=0 into a lagrange equation and find the optimal



min L() w.r.t w,w0 | max L() w.r.t λ

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

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

$\lambda_i \geq 0 \ \forall i$

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

$$\lambda_i\left[z_i(\underline{w}^{*T}\underline{y}_i + w_0) - 1\right] = 0 \ \forall i$$

ONLY HAVE TO OPTIMIZE W.R.T. $\underline{\lambda}$.
YIELDS SOLUTION $\underline{\lambda}^*$.

slack varable

INTRODUCE SLACK VARIABLES ξ_i, $i = 1, 2, \ldots, N$,
SUCH THAT:

$\xi_i = 0$ IF $\underline{y}_i$ IS ON CORRECT SIDE OF MARGIN

$\xi_i =$ NORMALIZED DISTANCE $\underline{y}_i$ TO MARGIN BOUNDARY, IF ON INCORRECT SIDE OF MARGIN BOUNDARY.

$\xi_i = 1$ IF $\underline{y}_i$ IS ON DECISION BOUNDARY #.

Now, CRITERION FCN becomes:

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

NEW TERM.

SLACK-VARIABLE PARAMETER.

CONSTRAINTS:

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

NEW TERM.

primal form

Now, LAGRANGIAN BECOMES:

$$L(\underline{w}, w_0, \underline{\lambda}, \underline{\mu}) = \frac{1}{2}\|\underline{w}\|^2 + C\sum_{i=1}^{N}\xi_i - \sum_{i=1}^{N}\lambda_i\left[z_i(\underline{w}^T\underline{y}_i + w_0) - 1 + \xi_i\right] - \sum_{i=1}^{N}\mu_i\xi_i$$

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

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

dual form

CAN BE EXPRESSED IN DUAL FORM;

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

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

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

nolinear mapping - kernel

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

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

$$K(\underline{x}_i, \underline{x}_j) = \underline{\phi}^T(\underline{x}_i)\, \underline{\phi}(\underline{x}_j)$$

$$= \underline{\text{KERNEL FUNCTION}}.$$

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

COMMON KERNEL FUNCTIONS:

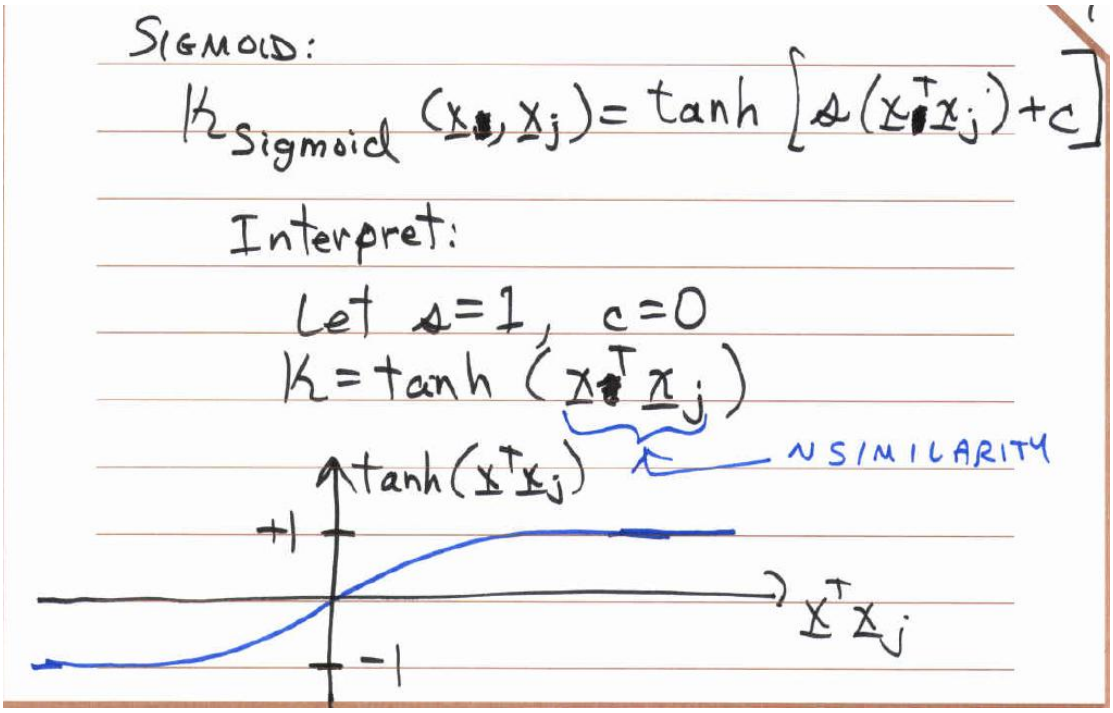
LINEAR: $K_{\text{Linear}}(\underline{x}_i, \underline{x}_j) = \alpha \underline{x}_i^T \underline{x}_j$.

POLYNOMIAL: $K_{\text{Poly}}(\underline{x}_i, \underline{x}_j)$

$$= (\underline{x}_i^T \underline{x}_j + 1)^P, \quad P \in \mathbb{Z}^+$$

RADIAL BASIS FUNCTION:

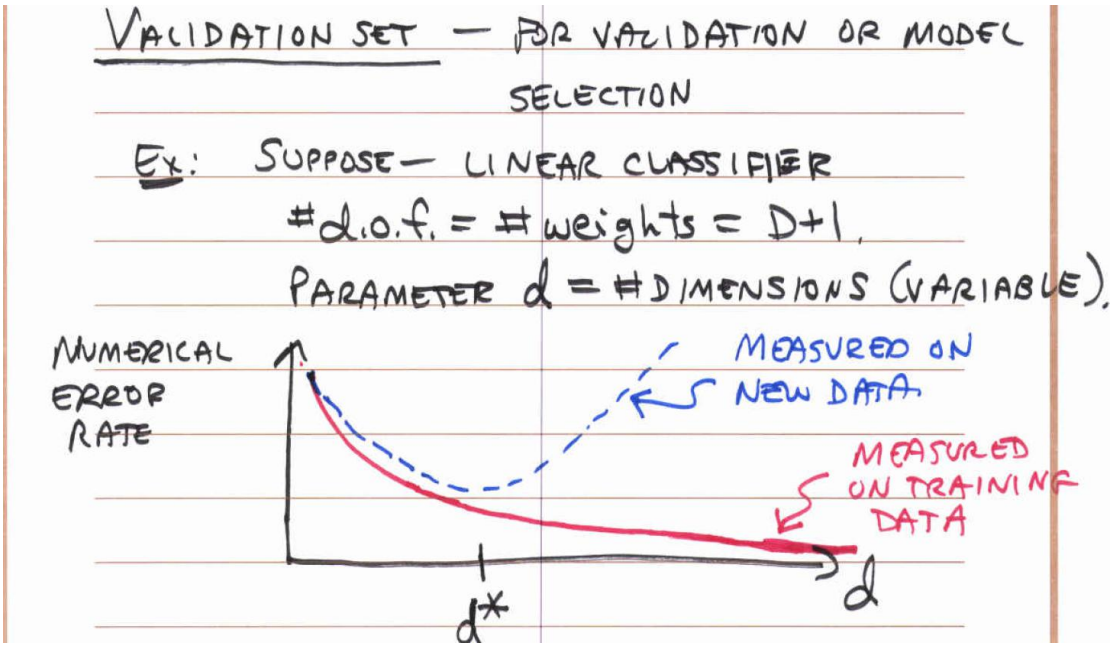
$$K_{\text{RBF}}(\underline{x}_i, \underline{x}_j) = \exp\{-\gamma \|\underline{x}_i - \underline{x}_j\|^2\}, \quad \gamma > 0.$$



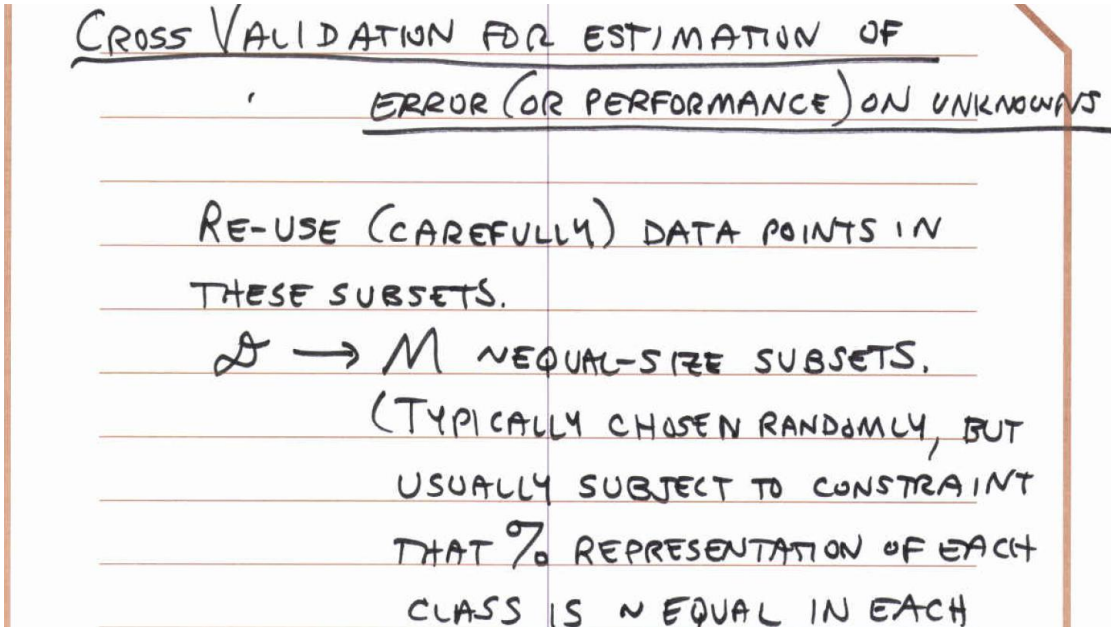
3.Validation and data reduction

validation

training set, test set, validation set



cross validation



cross validation for parameter selection

feature selection & dimension reduction

To solve too many D.o.F

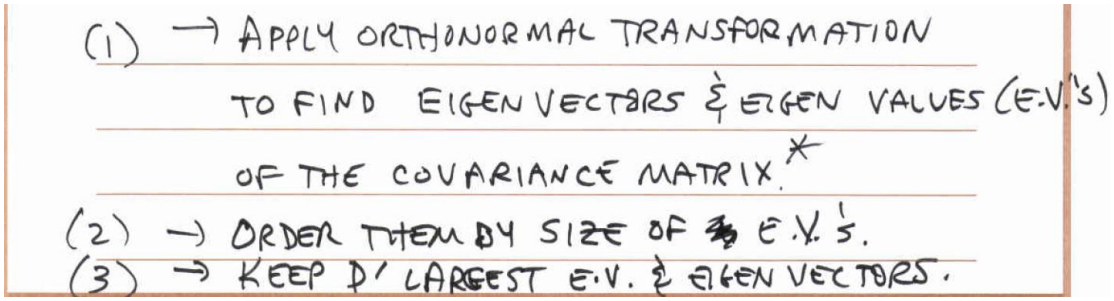
two type:

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)



* USE SAMPLE COVARIANCE MATRIX:

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

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

$$x' = E^T x$$

2.Fisher linear discriminant (LDA, NDA)

for a 2-D two class datasey, 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 seperation

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

DEF: FISHER'S LINEAR DISCRIMINANT IS THE LINEAR FUNCTION $w^T x$ FOR WHICH THE CRITERION FCN.:

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

IS MAXIMIZED.

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

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

$\Rightarrow$ IF $\underline{\underline{S}}_W$ IS NONSINGULAR:

$$\boxed{\underline{w} = \underline{\underline{S}}_W^{-1}\left[\underline{m}_1 - \underline{m}_2\right]} \leftarrow \text{F.L.D. SOLUTION.}$$

NOTE: DIRECTION OF $\underline{w}$ DEFINES THE NEW (1D) FEATURE SPACE.

$\therefore$ WE CAN NORMALIZE $\underline{w}$ 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. $1 \leq D' < D$. (SOME OF THE APPLY F.L.D. ITERATIVELY.)

multiple discriminant analysis

GENERALIZE F.L.D. QUANTITIES:

$\underline{\underline{S}}_i$ = SCATTER MATRIX FOR CLASS S_i

= SAME AS F.L.D.

$\underline{m}_i$ = MEAN OF CLASS S_i

= SAMA 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 DEFIND BY PROJECTION MATRIX $\underline{\underline{W}}$:

$$\underline{\underline{W}} \triangleq \left[\underline{w}_1 \;\vdots\; \underline{w}_2 \;\vdots\; \cdots \;\vdots\; \underline{w}_{C-1} \right]$$

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

THEN:

$$\underline{\tilde{u}}_j = \underline{\underline{W}}^T \underline{x}_j$$

$\underline{\tilde{u}}_j$: TYP. (C-1)-DIM.; $\underline{x}_j$: D-DIM.

4. Statistical Classification

Basic conception

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

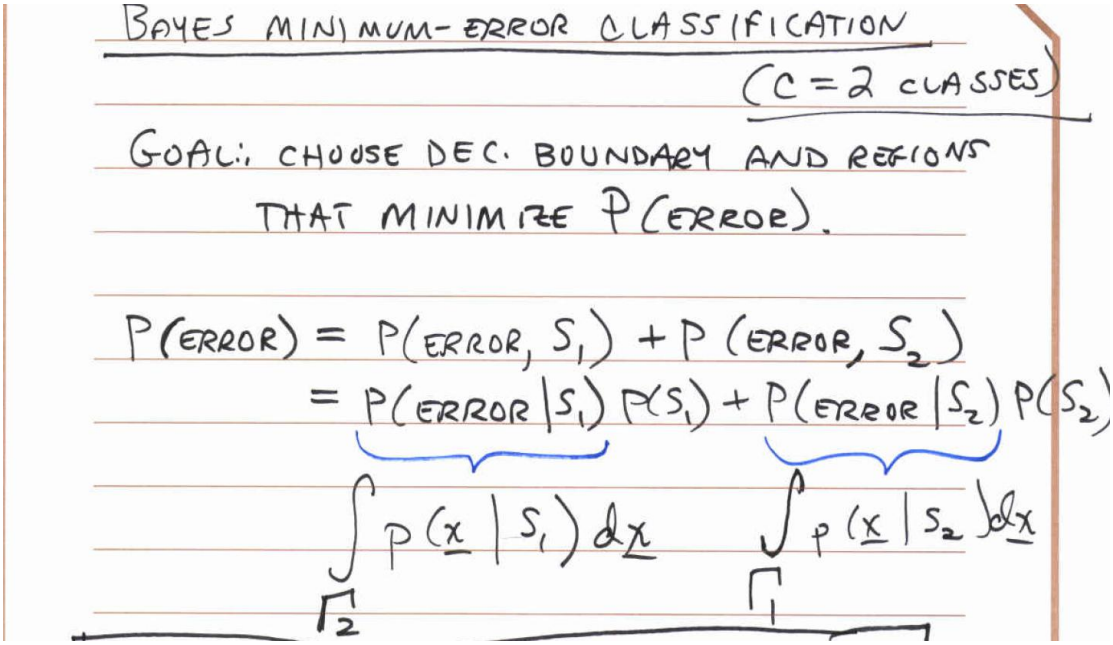
Bayes Minimum-Error classification

naive bayes a basic assumption: each feature is independent
it's like a Maximum A Posteriori in decision makinhg

RELATION BETWEEN MAP AND NAIVE BAYES

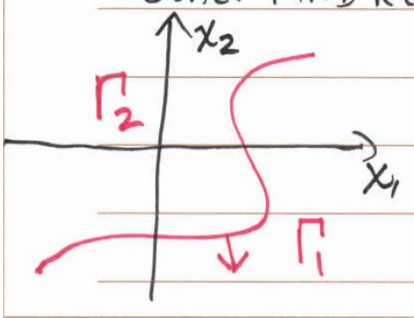
MAP use data point to estimate parameter --> θ , θ is the posterior and hypothesis

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



$$\therefore \quad P(\text{ERROR}) = P_e = P(S_1) \int_{\Gamma_2} p(\underline{x}|S_1)\,d\underline{x} + P(S_2) \int_{\Gamma_1} p(\underline{x}|S_2)\,d\underline{x} \tag{1}$$

GOAL: FIND REGIONS Γ_1, Γ_2 THAT MINIMIZE P_e.



KNOW: Γ_1 AND Γ_2 ARE NONOVERLAPPING.

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

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

$$p(\underline{x}|S_1)P(S_1) \underset{S_2}{\overset{S_1}{\gtrless}} p(\underline{x}|S_2)P(S_2) \tag{2}$$

↑ BAYES DEC. RULE FOR MIN. ERROR

OR: $\ln\left[p(\underline{x}|S_1)P(S_1)\right] \underset{S_2}{\overset{S_1}{\gtrless}} \ln\left[p(\underline{x}|S_2)P(S_2)\right]$

(3) $$P(S_1|\underline{x}) \underset{S_2}{\overset{S_1}{\gtrless}} P(S_2|\underline{x})$$

Σ BAYES DEC. RULE FOR MIN. P_e, IN TERMS POSTERIOR PROBABILITIES

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

OTHER FORMS:

REARRANGE (2): $$\frac{p(\underline{x}|S_1)}{p(\underline{x}|S_2)} \underset{S_2}{\overset{S_1}{\gtrless}} \frac{P(S_2)}{P(S_1)}$$

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

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

⇒ $$l(\underline{x}) \underset{S_2}{\overset{S_1}{\gtrless}} T$$

$$-\ln l(\underline{x}) \triangleq h(\underline{x}) \qquad \ln[l(\underline{x})] \gtrless \ln T$$

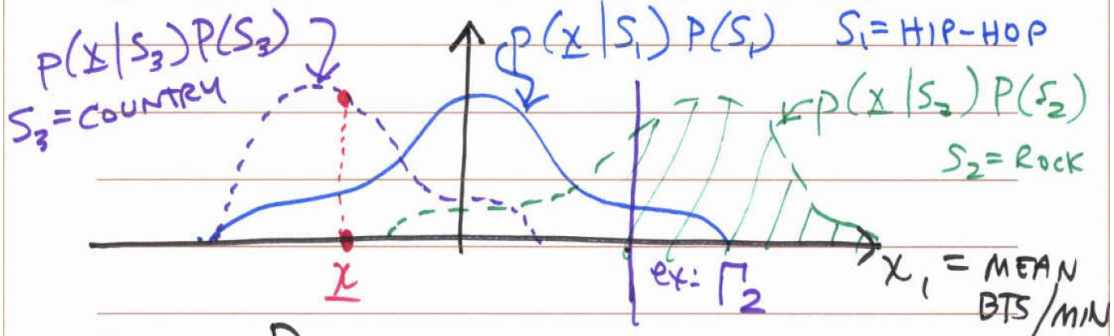
$$-\ln[l(\underline{x})] \lessgtr -\ln T$$

$$h(\underline{x}) = \ln p(\underline{x}|S_2) - \ln p(\underline{x}|S_1)$$

$$h(\underline{x}) \underset{S_2}{\overset{S_1}{\lessgtr}} \ln \frac{P(S_1)}{P(S_2)} \triangleq \tau$$

multiclass bayes

BAYES MIN. ERROR : C > 2 CLASSES



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

$$P_{correct} = \sum_{i=1}^{C} P(S_i) \int_{\Gamma_i} p(\underline{x}|S_i)\, d\underline{x}$$

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

$$\Rightarrow \boxed{p(\underline{x}|S_i)P(S_i) > p(\underline{x}|S_j)P(S_j) \;\; \forall j \neq i \Rightarrow \underline{x} \in S_i}$$

$\Leftarrow$ BAYES MIN. ERROR DEC. RULE, $C > 2$.

THIS IS A MAXIMAL VALUE METHOD.

summary

CAN DEFINE:

$$\boxed{g_i(\underline{x}) = p(\underline{x}|S_i)P(S_i) \quad \text{OR} \quad g_i(\underline{x}) = \ln\left[p(\underline{x}|S_i)P(S_i)\right]}$$

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:

$$p(\underline{x}|S_1)P(S_1) \underset{S_2}{\overset{S_1}{\gtrless}} p(\underline{x}|S_2)P(S_2)$$

2. PROBABILITY OF ERROR:

MULTICLASS:

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

2-CLASS:

$$P_e = \int_{\Gamma_2} p(\underline{x}|S_1)P(S_1)\,d\underline{x} + \int_{\Gamma_1} p(\underline{x}|S_2)P(S_2)\,d\underline{x}$$

finally, compared seveal distribution function, if σ1 = σ2, it's a linear classifier, else, quardrapt

Density Estimate

conditions in density estimate

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

DON'T KNOW $p(\underline{x}|S_i)$,

MAYBE DON'T KNOW $P(S_i)$.

DON'T MAKE ANY ASSUMPTIONS ON $p(\underline{x}|S_i)$

basic sinary

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

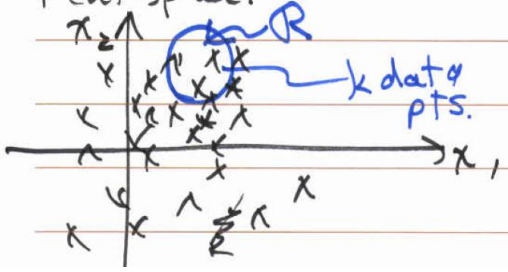
$\underline{x}_1, \underline{x}_2, \cdots, \underline{x}_N$.

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

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

$$\Downarrow \quad P = \frac{\bar{k}}{N}$$

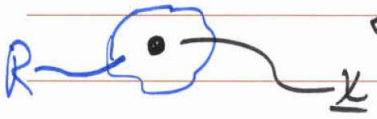
Feat. space:



$\frac{k}{N}$ = REASONABLE EST. OF P.

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

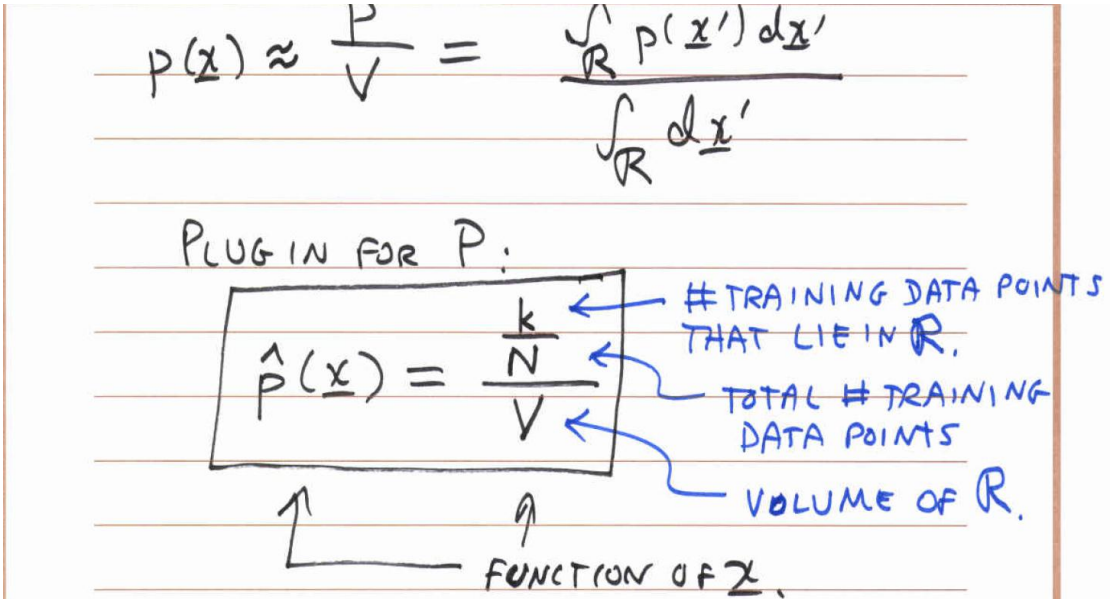
ALSO ASSUME $p(\underline{x})$ IS CONTINUOUS AT $\underline{x}$.

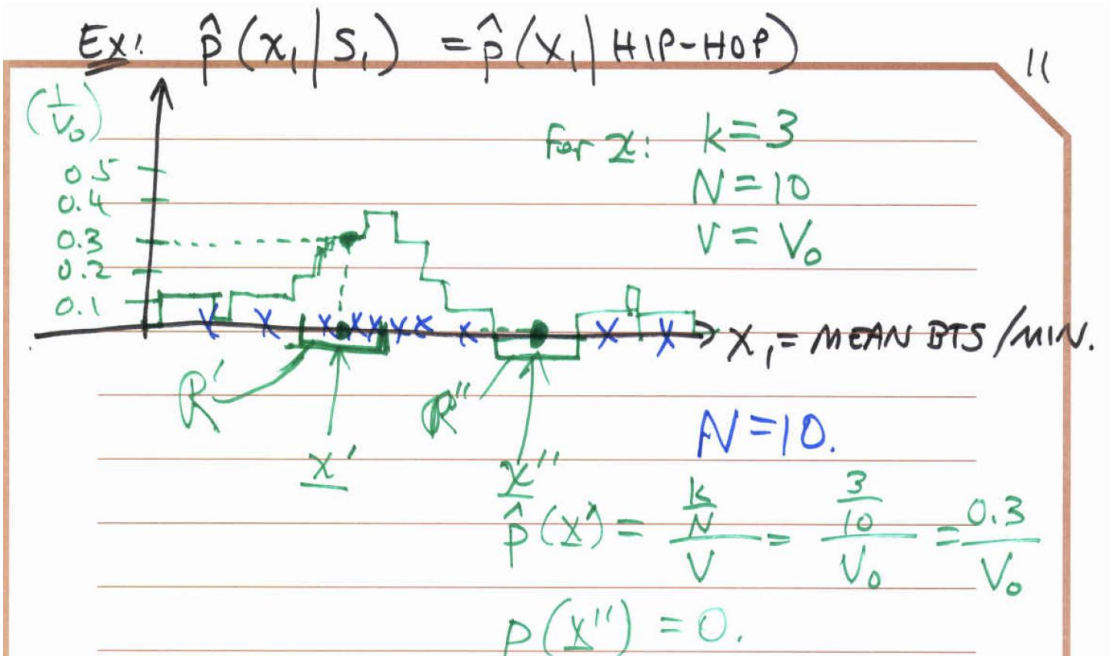


$$P = \int_{\mathcal{R}} p(\underline{x}')\, d\underline{x}'$$

approx. $p(\underline{x}') \approx p(\underline{x})$ = const. in $\mathcal{R}$.

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





2 methods to make density estimation

2 PRIMARY APPROACHES TO ESTIMATING $p(\underline{x})$

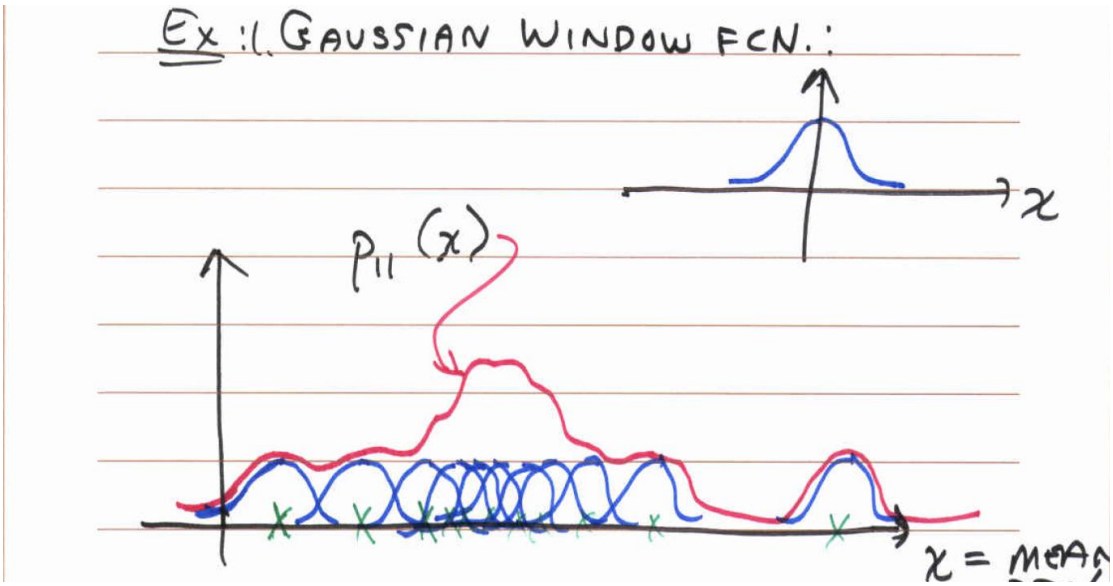
1. SPECIFY $V_n = f_1(n)$
   (PERHAPS ALSO SPECIFY SHAPE OR PROFILE OF R_n).
   e.g.: $V_n = \frac{1}{\sqrt{n}}$
   $\Rightarrow$ PARZEN WINDOWS (PW)
2. SPECIFY $k_n = f_2(n)$
   e.g.: $k_n = \sqrt{n}$.
   $\Rightarrow$ k-NEAREST NEIGHBORS

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

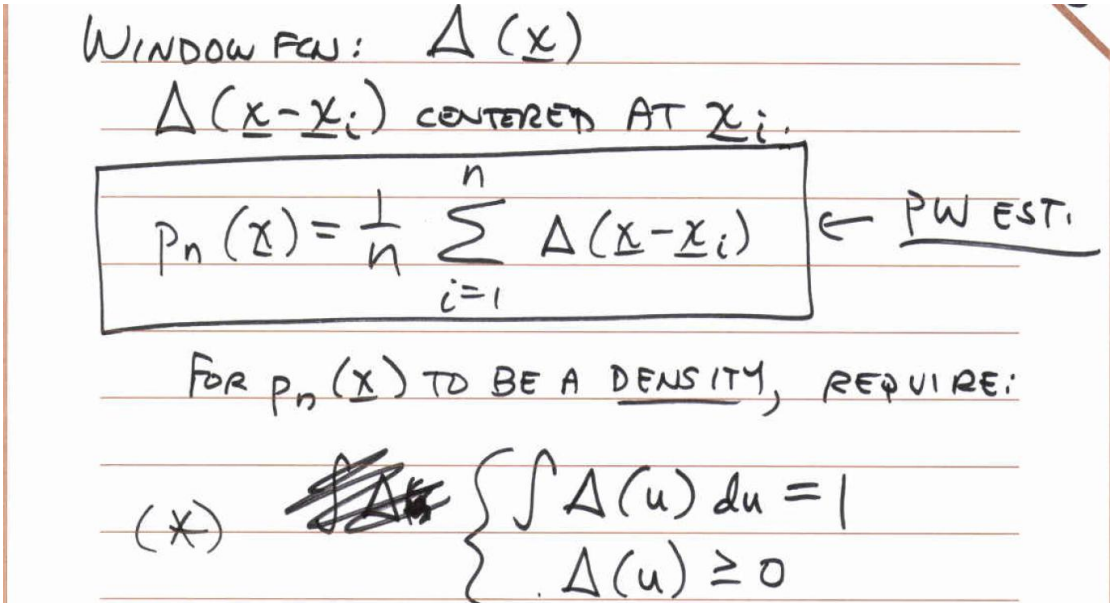
1.Parzen Windows

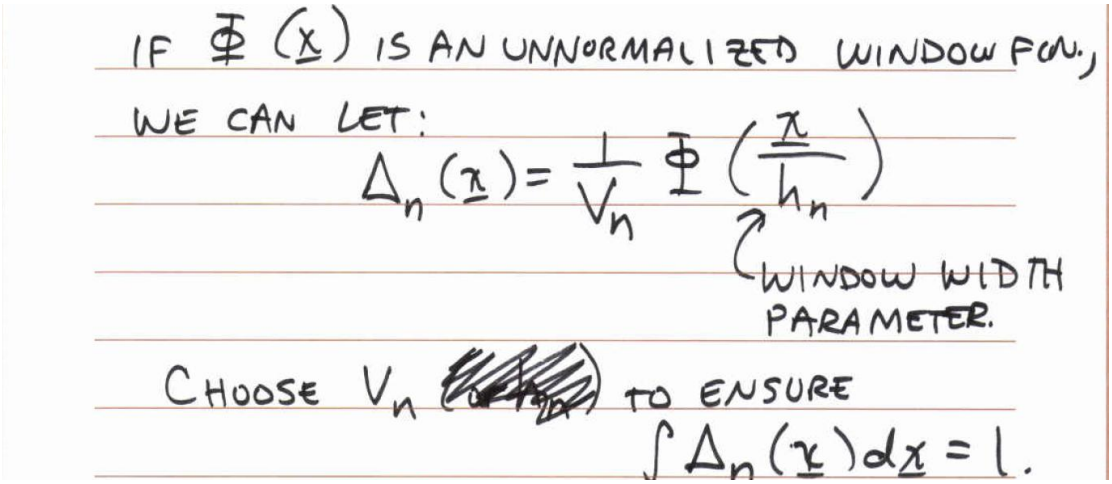
PARZEN WINDOWS ESTIMATION (PW)

1. CREATE A WINDOW FUNCTION ~~$f_1(n)$~~.
2. CENTER THE WINDOW FCN. AT EACH DATA POINT.
3. TAKE SUM OF ALL WINDOW FUNCTIONS, DIVIDED BY n.



window detail





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:

$$p_n(\underline{x}) \approx \frac{\frac{k_n}{n}}{V_n}$$

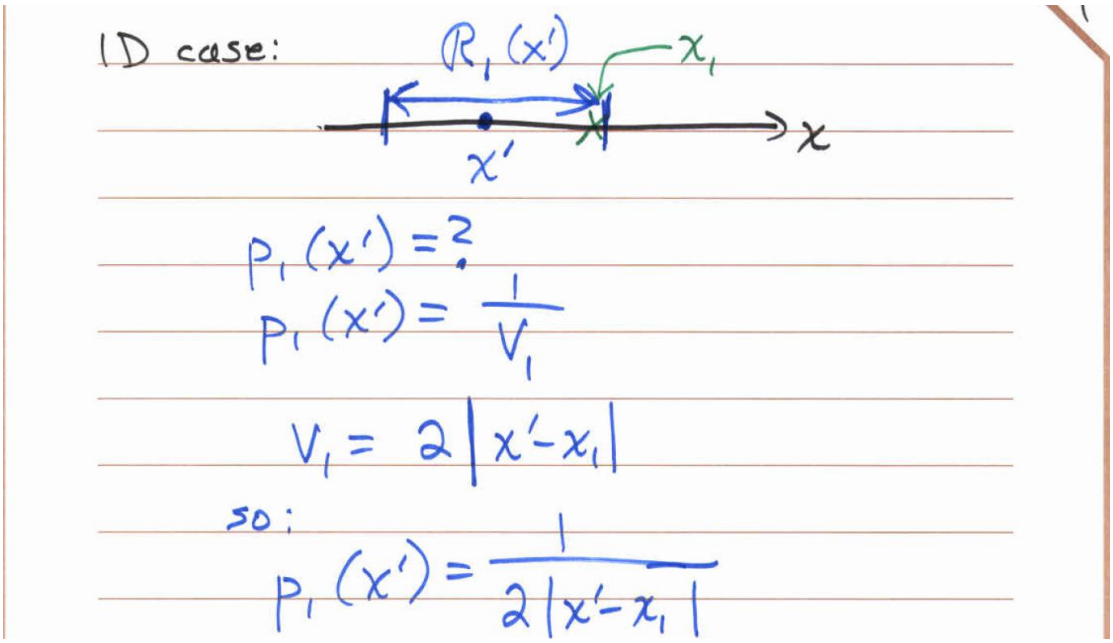
← SPECIFY THIS.
← CALCULATE FROM THE DATA.

EX: COMMON CHOICE: $k_n = \sqrt{n}$.

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

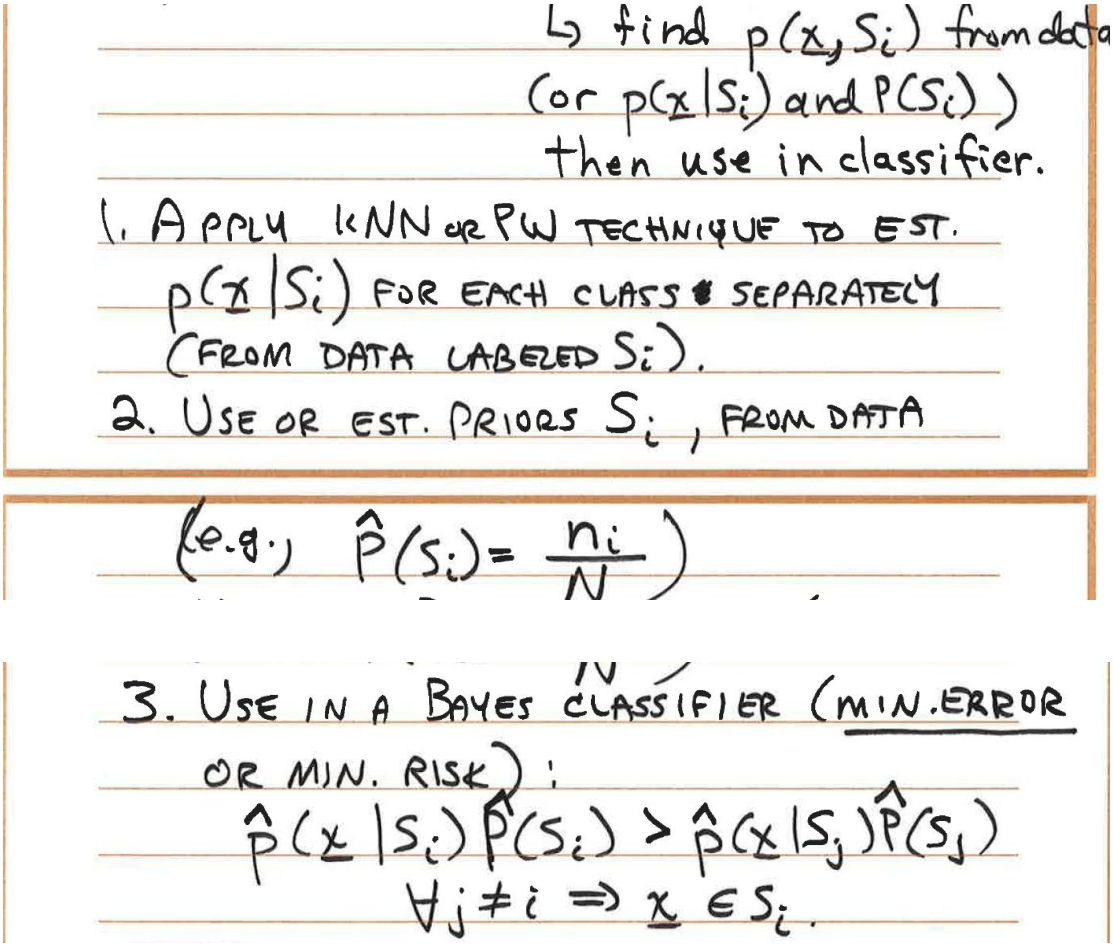
EX: LET $n=1 \Rightarrow k_n = \sqrt{n} = 1$.

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

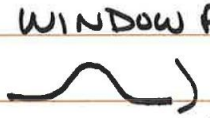


classification based on density estimation

approach1: Generative Model



approach2: Discrimination Model

FOR THE CASES OF k-NN EST. OR PW EST. USING A BINARY-VALUED WINDOW FUNCTION ( INSTEAD OF), CAN DERIVE AN EXPRESSION IN ADVANCE BASED ON EST. OF POSTERIOR PROBABILITIES:

$$P(S_i|\underline{x}) > P(S_j|\underline{x})$$
$$\forall j \neq i \Rightarrow \underline{x} \in S_i.$$

AS FOLLOWS.

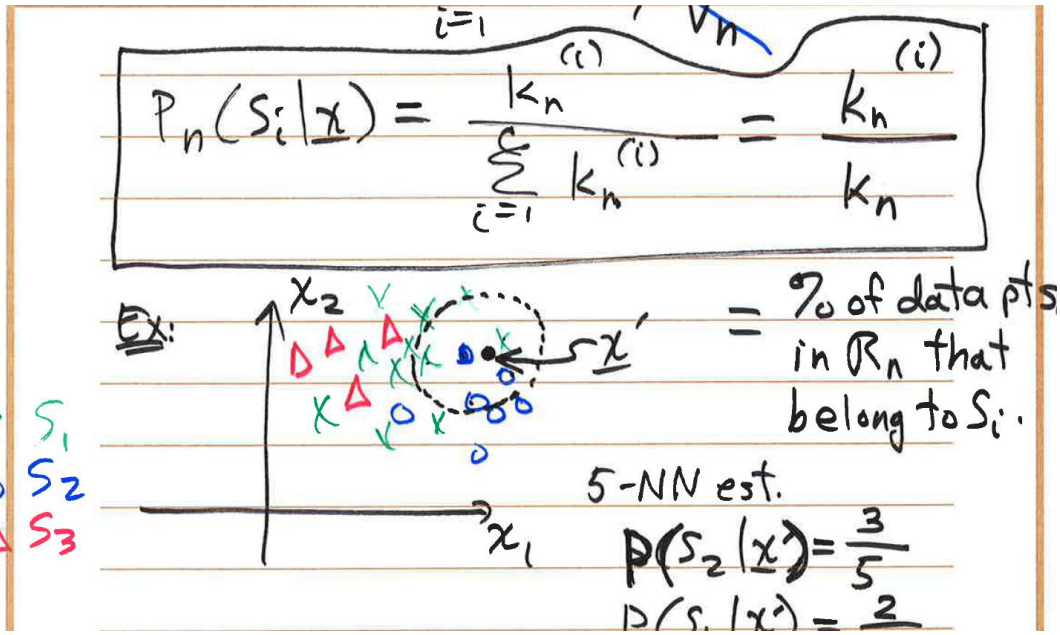
$$p_n(\underline{x}, S_i) = \frac{\frac{k_n^{(i)}}{n}}{V_n}$$

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

n = TOTAL # DATA PTS. OVER ALL CLASSES.

V_n = VOLUME OF R_n

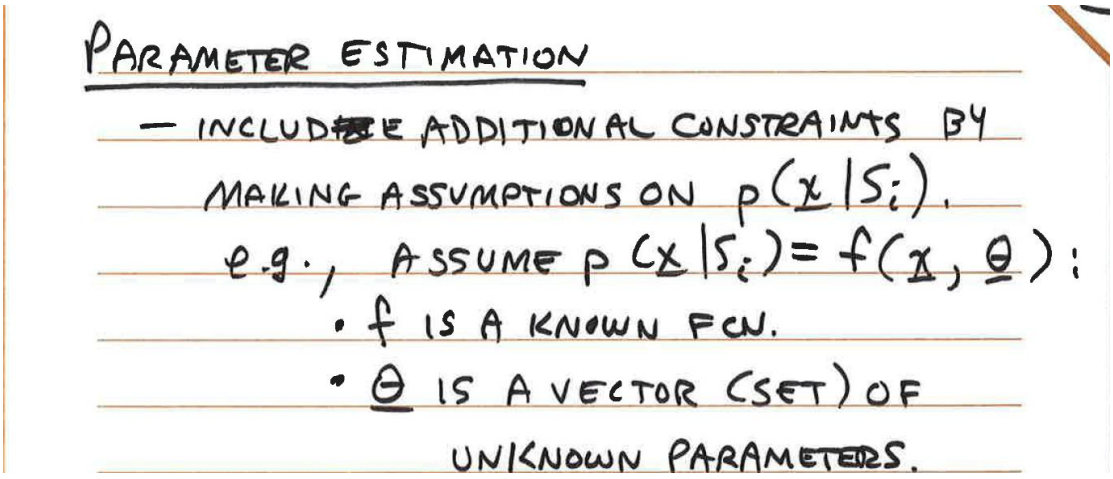
k_n = TOTAL # DATA PTS. IN R_n (will use later).



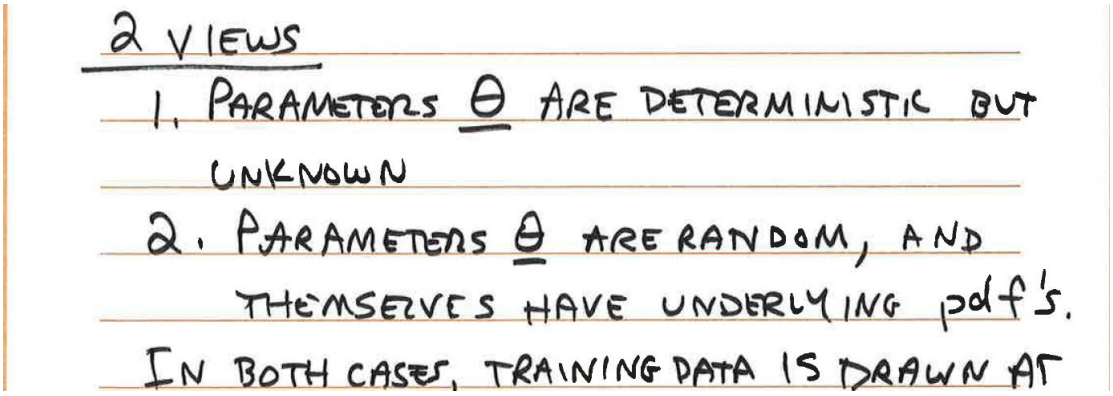
dimension in density estimate: -- curse of dimensionality

Parameter Estimate

two basic assumption



explanation:



Frist View:

VIEW 1 – $\underline{\theta}$ IS DETERMINISTIC

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

LET $\underline{x}_1, \underline{x}_2, \cdots, \underline{x}_n$ BE THE DATA, DRAWN I.I.D. FROM DENSITY $p(\underline{x})$ [OR $p(\underline{x}|S_i)$].

LET $\underline{\mathcal{X}} = \begin{bmatrix} \underline{x}_1 \\ \underline{x}_2 \\ \vdots \\ \underline{x}_n \end{bmatrix}$

ALSO: $\mathcal{D}_i = \{\underline{x}_1, \underline{x}_2, \cdots, \underline{x}_n\}$ = DATA PTS. LABELED S_i.

$$\underline{\hat{\theta}} = G(\underline{x}_1, \underline{x}_2, \cdots, \underline{x}_n)$$
$$= G(\underline{\mathcal{X}})$$
$$= \underline{\hat{\theta}}(\underline{\mathcal{X}})$$

$\underline{x}_i$ ARE DRAWN AT RANDOM $\Rightarrow$ $\underline{\hat{\theta}}$ IS RANDOM

use knowledge of unbias and consistant : θ^ is the estimate of θ

two method used in view 1:

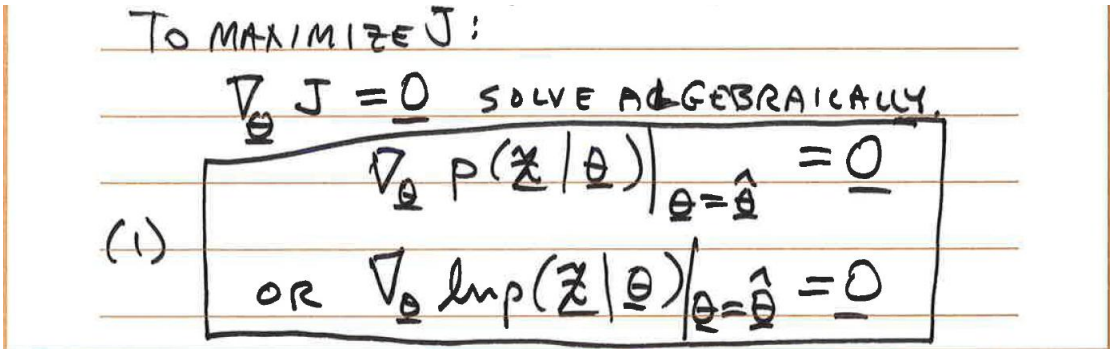
Maximum likelihood Estimate

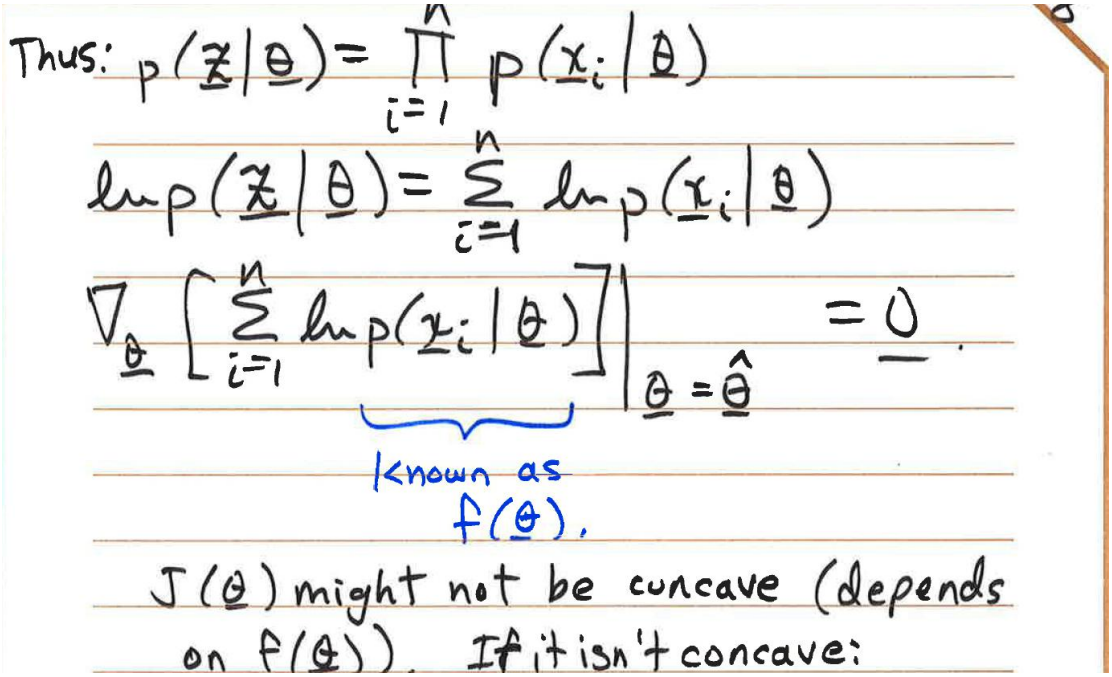
CRITERION FCN: $J_{ML}(\underline{\theta}) = p(\underline{\mathcal{X}}|\underline{\theta})$

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

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

PROBABILITY OF OBTAINING THE DATA POINTS THAT WERE ACTUALLY OBSERVED.





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: $\hat{\underline{\theta}}_{MAP} = \arg\max_{\underline{\theta}} p(\underline{\theta}|\underline{z})$

or $\hat{\underline{\theta}}_{MAP} = \arg\max_{\underline{\theta}} \ln p(\underline{\theta}|\underline{z})$

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

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

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

$\ln p(\underline{z}|\underline{\theta})$: ✓

$\ln p(\underline{\theta})$: 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:

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

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

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

$$\Rightarrow p(\underline{x}|\underline{\mathcal{Z}}) = \int p(\underline{x}|\underline{\theta})\, p(\underline{\theta}|\underline{\mathcal{Z}})\, d\underline{\theta}$$

INSERTING CLASS INDECET AND S_i:

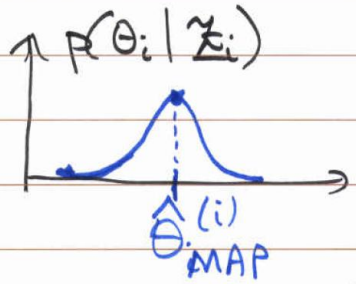
$$(3)\quad p(\underline{x}|S_i, \underline{\mathcal{Z}}_i) = \int p(\underline{x}|\underline{\theta}_i, S_i)\, p(\underline{\theta}_i|\underline{\mathcal{Z}}_i)\, d\underline{\theta}_i$$

KNOWN.

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

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

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



LIKELIHOOD (KNOWN)
PRIOR

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

NORMALIZING FACTOR

FOR BAYES MIN. ERROR CLASSIFIER:

IN (2), USE:

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

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

THEN (3) $\Rightarrow$

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

$\Rightarrow$ SAME AS APPROACH 1.

5.Artifical Neural Network

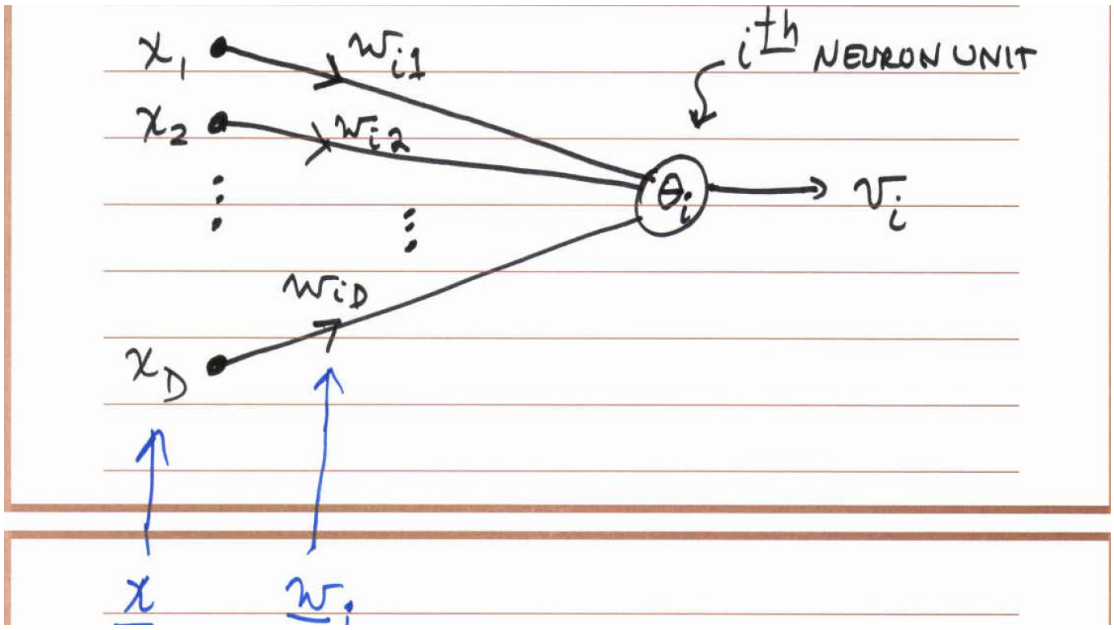
ARTIFICIAL NEURAL NETWORKS (ANN)

[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.

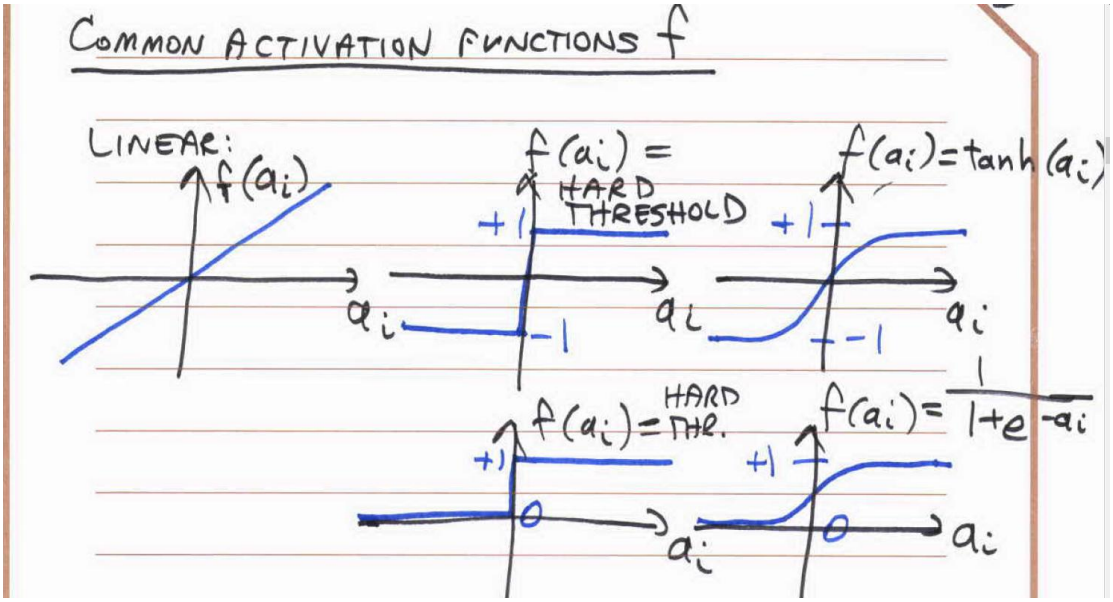
basic elements



$$(1)\quad v_i = f\left(\underline{w}_i^T \underline{x} - \theta_i\right) = f(a_i)$$

f: ACTIVATION FUNCTION. θ_i: THRESHOLD. a_i: (MEMBRANE) POTENTIAL

activation function



Relu Function family

Noisy ReLUs [edit]

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

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

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

Leaky ReLUs [edit]

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

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

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.[12]

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

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

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

and thus has a relation to "maxout" networks.[12]

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.[13]

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

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

neural network classify algorithm

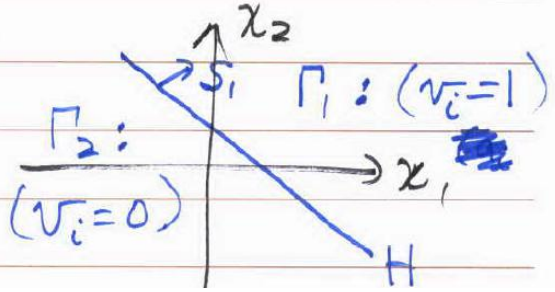
understanding of nerual network: single neuron

CONSIDER f = HARD THRESHOLD:

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

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

$\Rightarrow$ SINGLE NEURON UNIT (WITH $\underline{x}$ INPUT) IMPLEMENTS A 2-CLASS LINEAR CLASSIFIER.



algorithm: perceptron

PERCEPTRON LEARNING

ORIG. PERCEPTRON RULE:

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

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

$\underline{w}(i+1) = \underline{w}(i) + \eta(i) z_n \underline{x}_n$

otherwise

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

Next i

perceptron in Neural Network

based on the w update function

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

LET $\Delta \underline{w} \triangleq \underline{w}(i+1) - \underline{w}(i)$

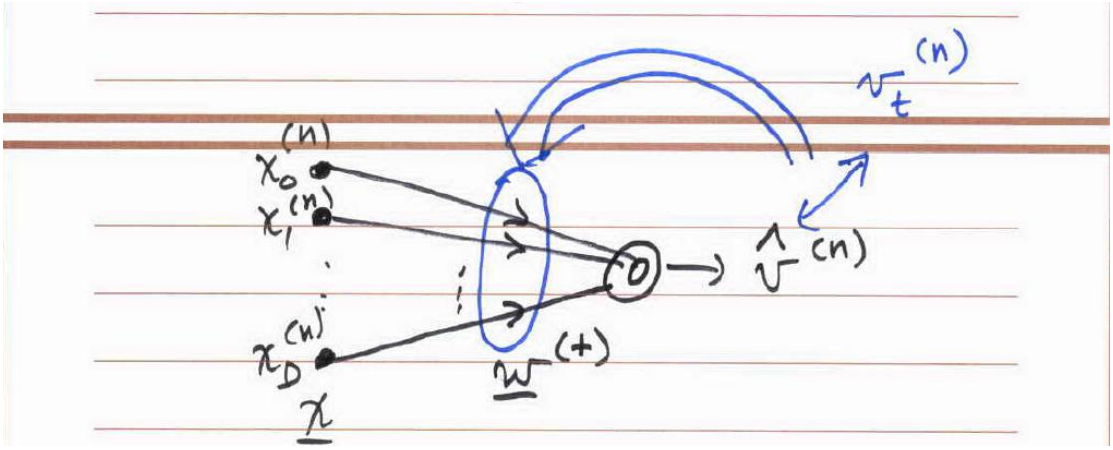
$\Delta \underline{v}^{(n)} \triangleq v_t^{(n)} - \hat{v}^{(n)}$

(2) $\Rightarrow$

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

Choose:

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



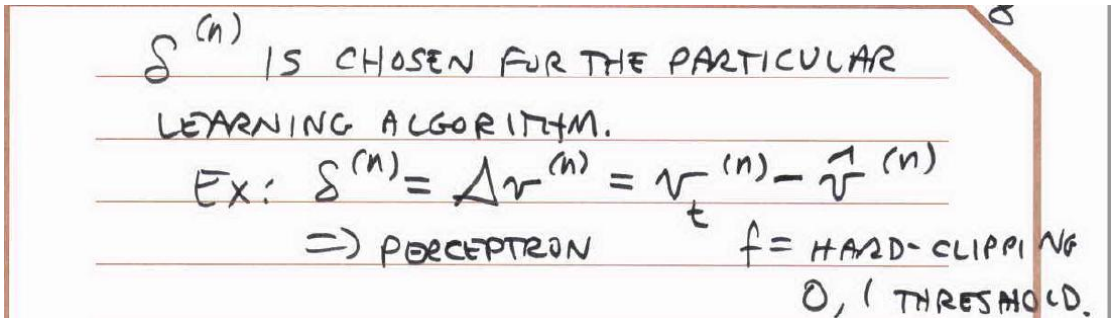
MORE GENERALLY

(3) $\boxed{\Delta \underline{w} = \eta \, \delta^{(n)} \, \underline{x}^{(n)}}$

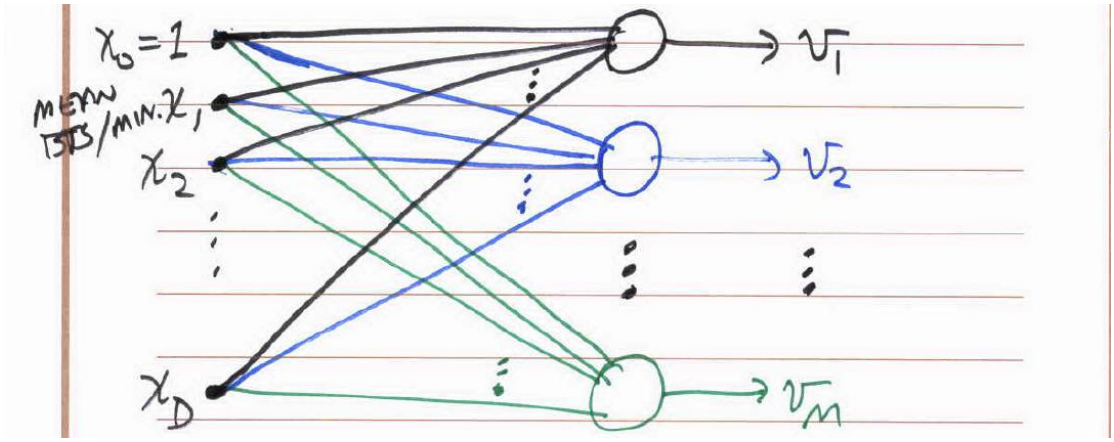
n = DATA PT. INDEX

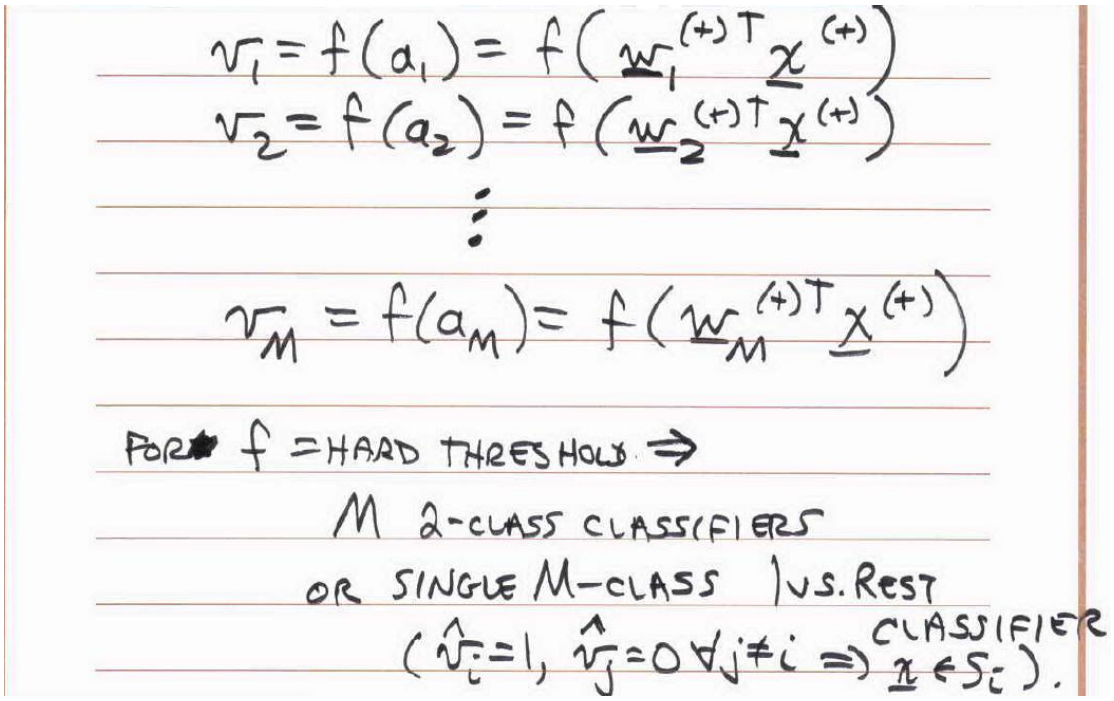
OUTER PRODUCT LEARNING

use δx to explain output error and as input for back prop



multi-neuron units (single layer)





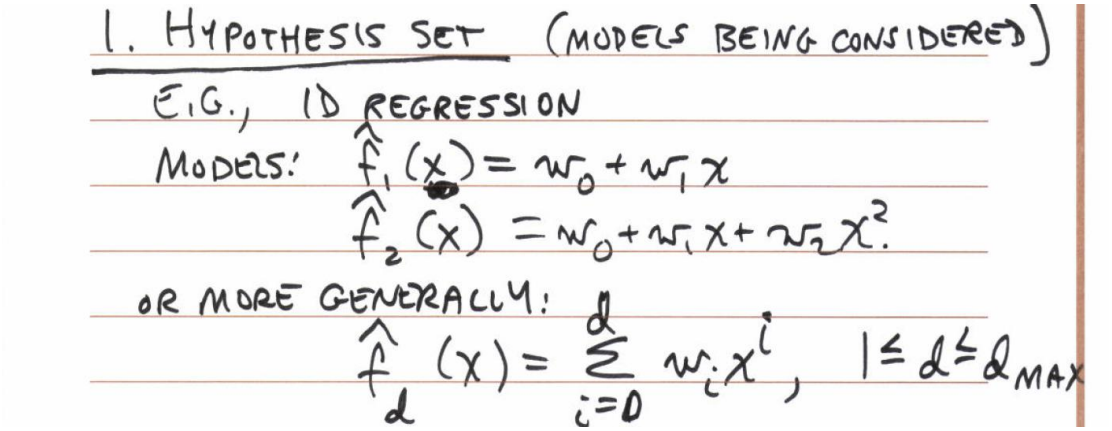
das

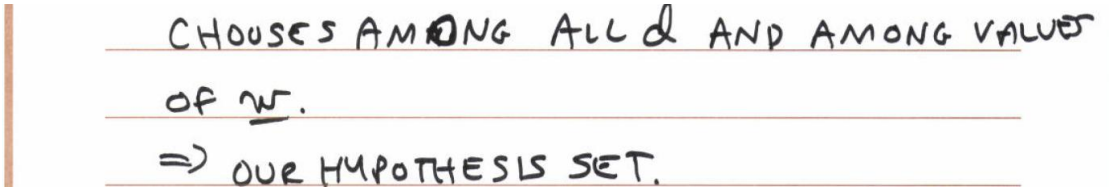
EE660 Machine Learning

1.Introduction

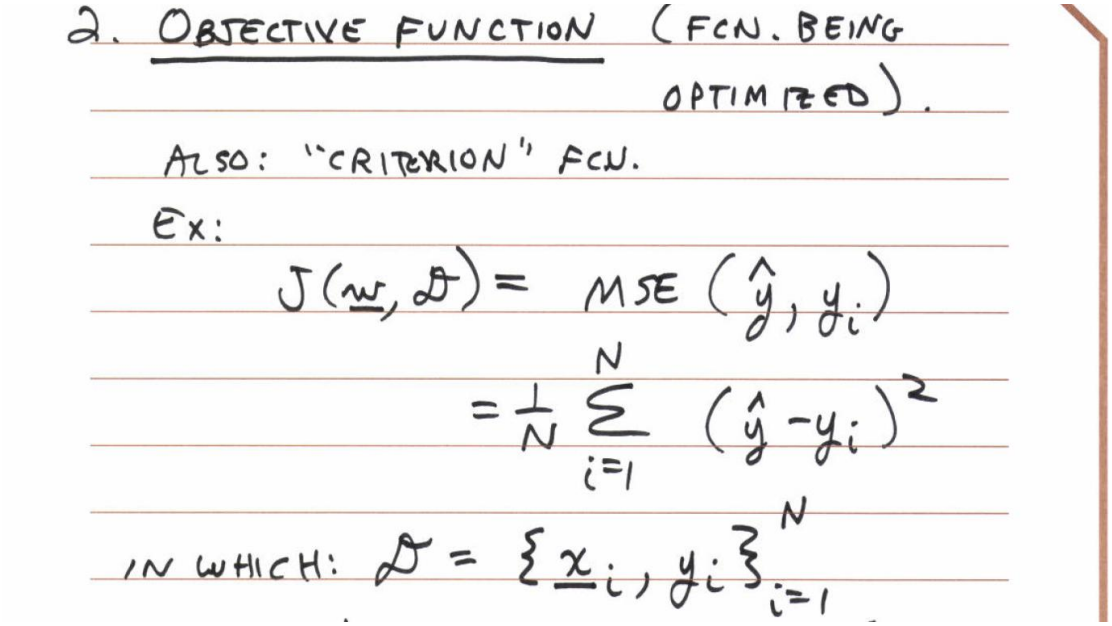
Basic Concept

Hypothesis set





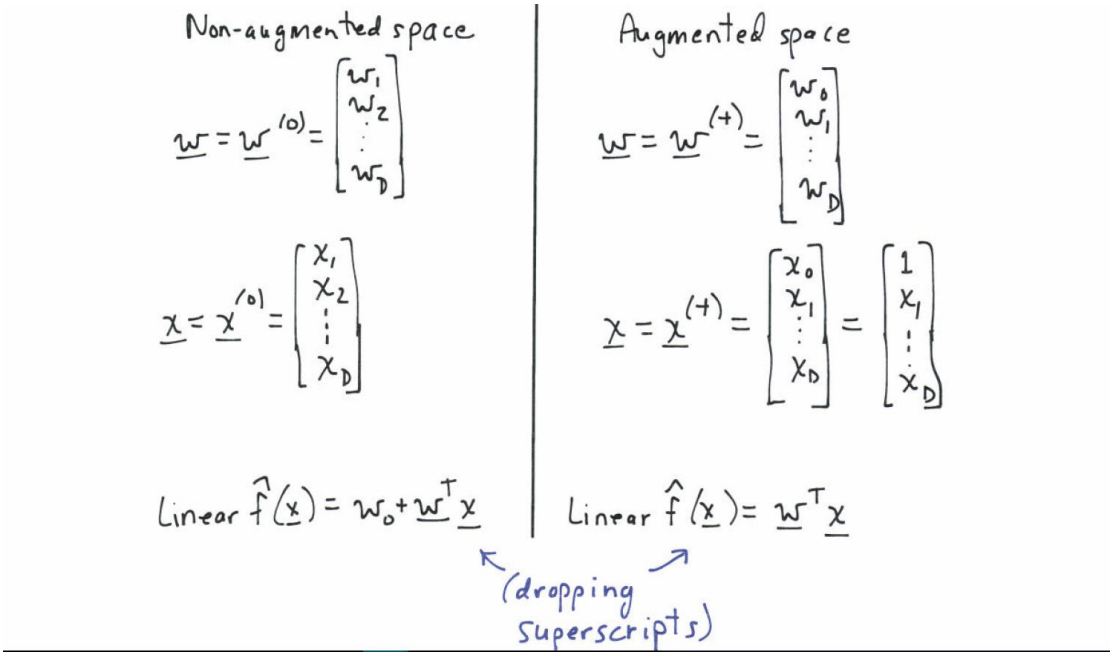
objective function



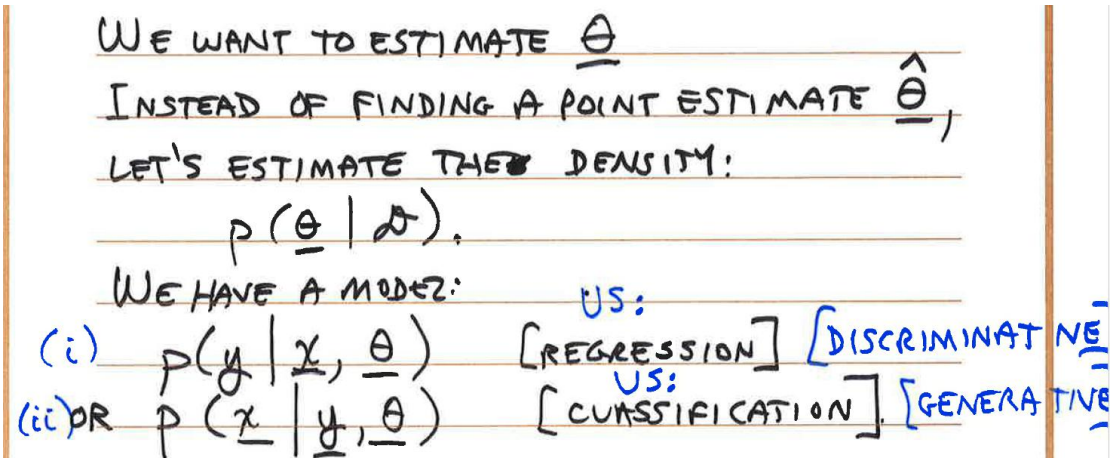
optimization method

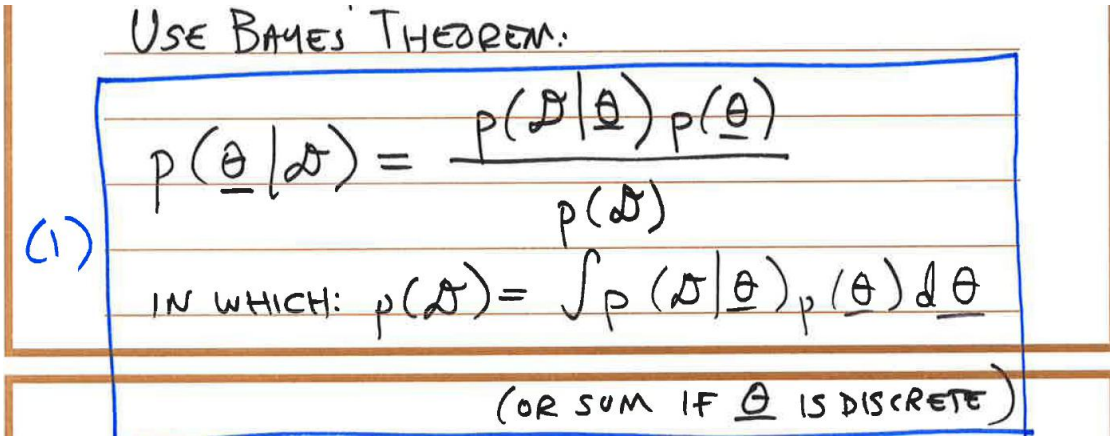
complexity of hypothesis set, data,

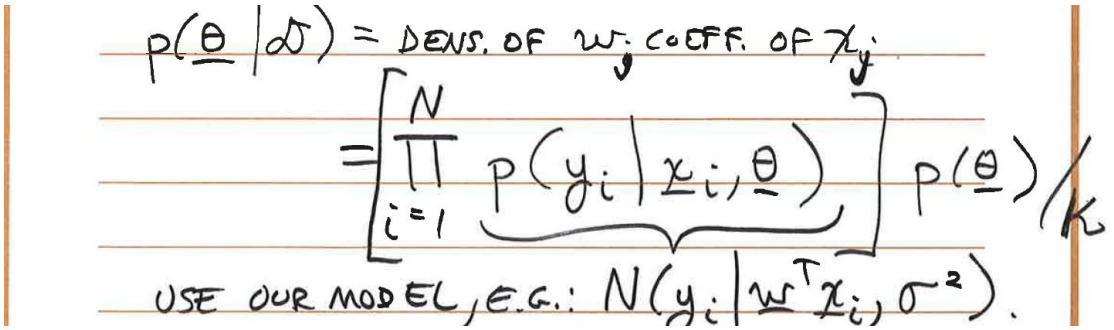
augmemt space (π machine)

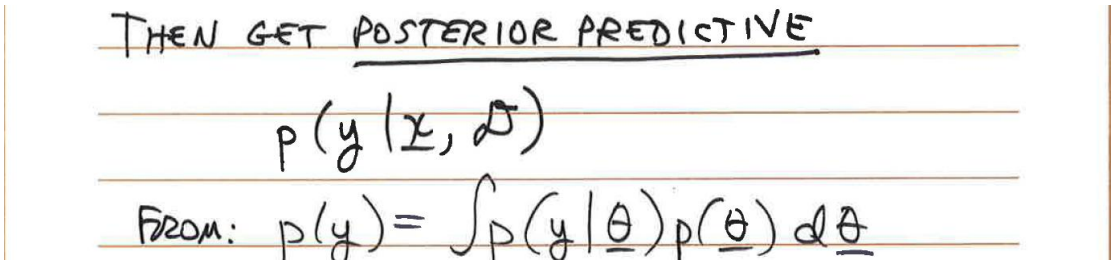


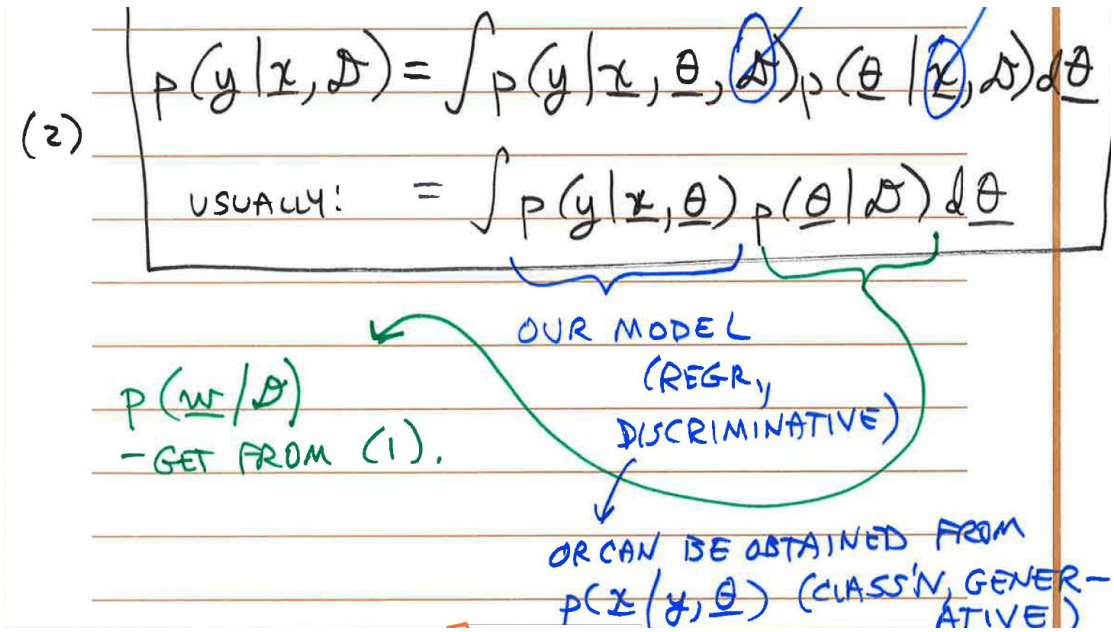
Bayes Estimation











2.Algorithm: Regression

Bayesian Regression

Linear Regression

MAP & ML

REGRESSION USING MAXIMUM LIKELIHOOD ESTIMATE (MLE)

$\underline{w} = \underline{w}^{(+)}$ THROUGHOUT [M 7.3]

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

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

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

AND:

$$\hat{\underline{w}}_{MLE} = \underset{\underline{w}}{\operatorname{argmax}} \underbrace{\ln p(\mathcal{D} | \underline{w})}$$

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

ASSUMING POINTS IN $\mathcal{D}$ ARE INDEPENDENTLY DISTRIBUTED (i.d.).

Using $p(\mathcal{D}|\underline{\theta}) = \prod_{i=1}^{N} p(y_i|\underline{x}_i, \underline{\theta})$

$$= \prod_{i=1}^{N} N(y_i | \underline{w}^T \underline{x}_i, \sigma^2)$$

↑ plug in

$$\Rightarrow J_1(\underline{w}) = \underbrace{\frac{1}{2\sigma^2}}_{\text{Drop}} \sum_{i=1}^{N} (y_i - \underline{w}^T \underline{x}_i)^2 + \underbrace{\frac{N}{2} \log(2\pi\sigma^2)}_{\text{const. of } \underline{w}}$$

Let $J(\underline{w}) = \frac{1}{2} RSS(\underline{w})$

$$= \frac{1}{2} \sum_{i=1}^{N} (y_i - \underline{w}^T \underline{x}_i)^2$$

$$= \frac{1}{2} \| \underline{y} - \underline{\underline{X}}\, \underline{w} \|_2^2$$

$$= \frac{1}{2} (\underline{y} - \underline{\underline{X}}\, \underline{w})^T (\underline{y} - \underline{\underline{X}}\, \underline{w})$$

$\nabla_{\underline{w}} J(\underline{w}, \mathcal{D}) = \underline{0}$ & SOLVE ALGEBRAICALLY

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

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

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

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

conclusion of MLE and MAP

SUMMARY OF ESTIMATION TECHNIQUES

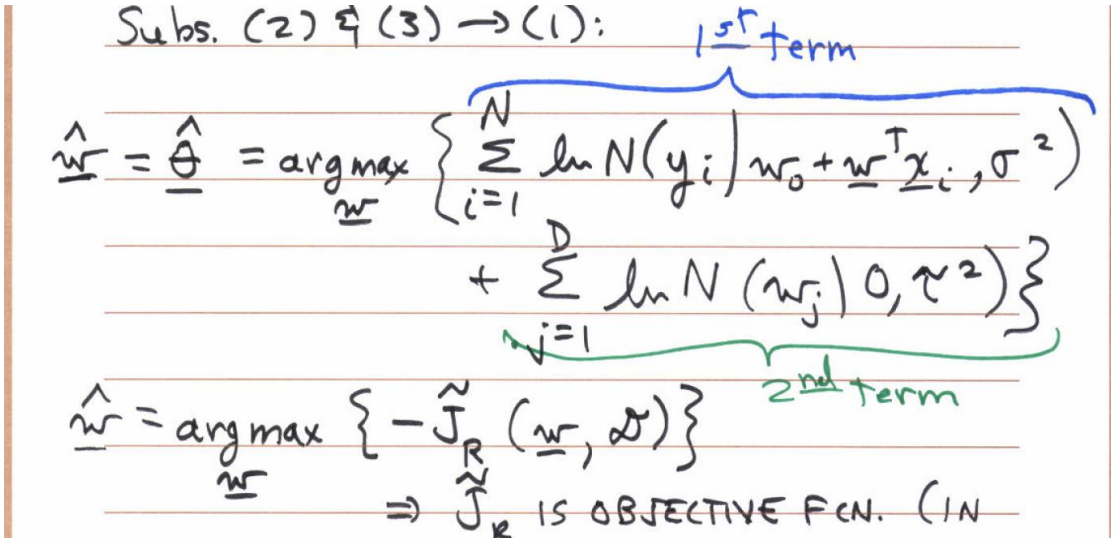
MLE: $\hat{\underline{\theta}}_{MLE} = \underset{\underline{\theta}}{\operatorname{argmax}} \{\ln p(\mathcal{D}|\underline{\theta})\}$

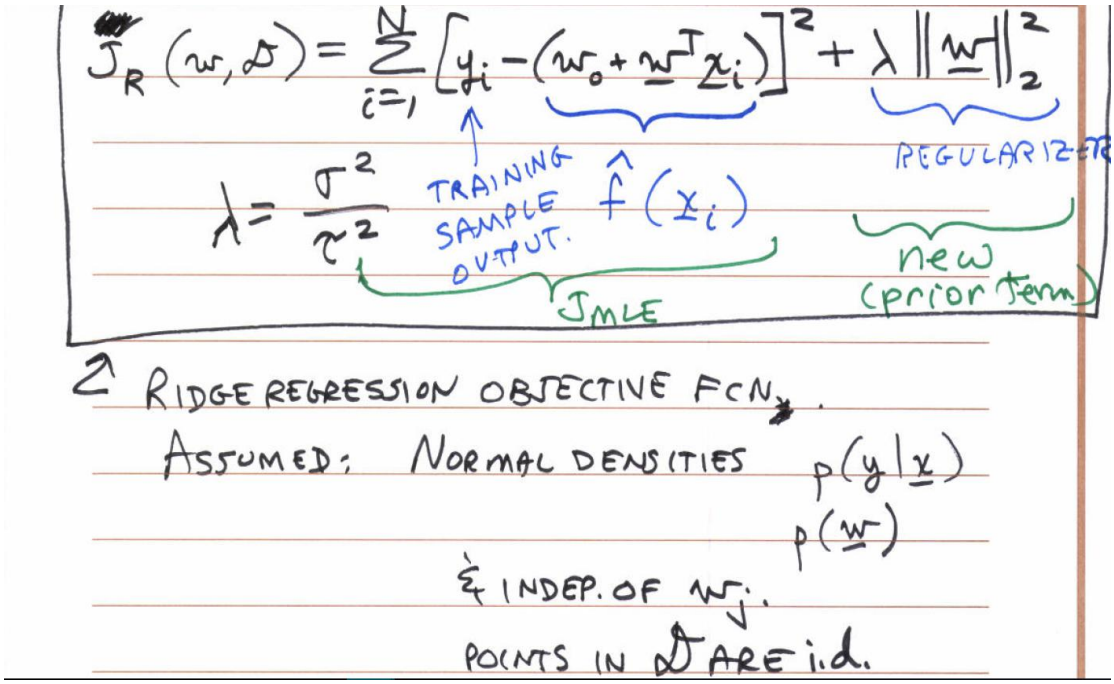
GAUSSIAN CASE $\Rightarrow$ $J_{MLE}(\underline{w}, \mathcal{D}) = MSE$

MAP: $\hat{\underline{\theta}}_{MAP} = \underset{\underline{\theta}}{\operatorname{argmax}} \{\ln p(\mathcal{D}|\underline{\theta}) + \ln p(\underline{\theta})\}$

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

Ridge Regression

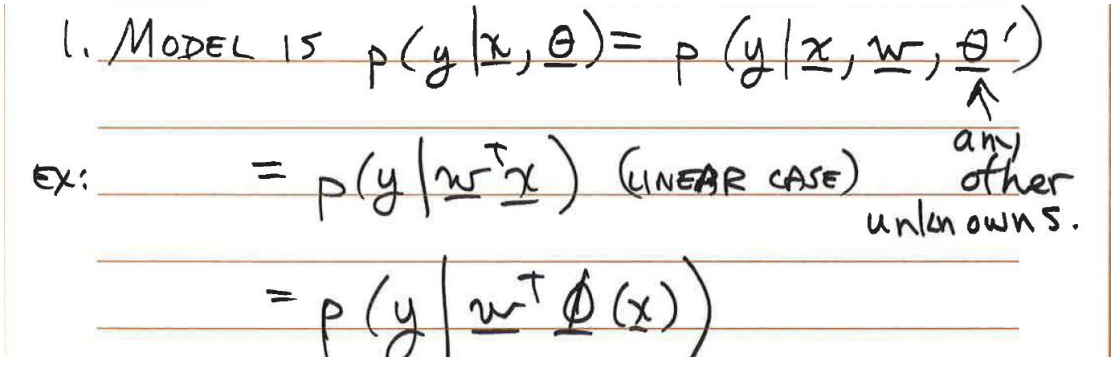




parameter estimate

could be similar as

Bayesian Regression



2. PARAMETER POSTERIOR

FROM EQ. (1):

$$p(\underline{w}|\mathcal{D}) = p(\underline{w}|\underline{y}, \underline{\underline{X}}) = \frac{p(\underline{y}|\underline{w}, \underline{\underline{X}})\, p(\underline{w}|\underline{\underline{X}})}{K}$$

likelihood, prior

$$K = \int p(\underline{y}|\underline{w}, \underline{\underline{X}})\, p(\underline{w}|\underline{\underline{X}})\, d\underline{w}$$

3. POSTERIOR PREDICTIVE

FROM EQ. (2):

$$p(y|\underline{x}, \mathcal{D}) = \int p(y|\underline{x}, \underline{w}, \cancel{\mathcal{D}})\, p(\underline{w}|\cancel{\underline{x}}, \mathcal{D})\, d\underline{w}$$

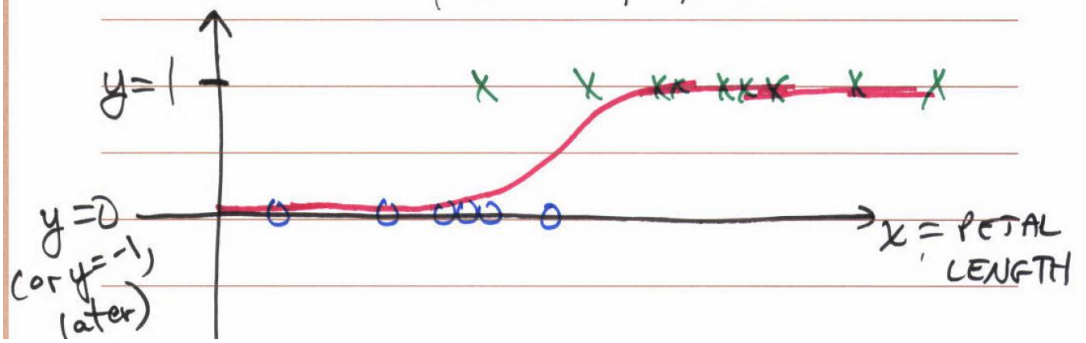
$$= \int p(y|\underline{x}, \underline{w})\, p(\underline{w}|\mathcal{D})\, d\underline{w}.$$

Logistic Regression

basic function

LOGISTIC REGRESSION

$$\hat{f}(x) = p(y=1 \mid x, \mathcal{D})$$
$$= p(\text{setosa} \mid x, \mathcal{D})$$



$$\hat{f}(\underline{x}) = \text{sigm}\{\underline{w}^T \underline{x}\} \quad \text{[Murphy notation]}$$
$$= \frac{e^{\underline{w}^T \underline{x}}}{1 + e^{\underline{w}^T \underline{x}}}$$
$$= \theta(\underline{w}^T \underline{x}) \quad \text{[AML notation]}$$

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

model and simple form

MODEL

$$p(y \mid \underline{x}, \underline{w}) = \text{Ber}(y \mid \text{sigm}(\underline{w}^T \underline{x}))$$
$$= \mu^{\mathbb{I}(y=1)} (1-\mu)^{\mathbb{I}(y=0)} \quad \text{[M 8.2]}$$

IN WHICH $\mu = \text{sigm}(\underline{w}^T \underline{x})$

CHANGE OUTPUT (y) REPRESENTATION:

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

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

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

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

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

objective function for logistic regression

OBJECTIVE FCN.

MAX. LIKELIHOOD

$$\text{LIKELIHOOD} = p(\mathcal{D} \mid \underline{w})$$

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

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

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

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

$$= J(\underline{w}, \mathcal{D})$$

↑ OBJ. FCN. FOR MLE OF $\underline{w}$ IN LOGISTIC REGRESSION.

J IS CONVEX.

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

minimize lost(objective function)

CAN SHOW:

$$\nabla_{\underline{w}} E_i = \frac{\tilde{y}_i \underline{x}_i}{1 + e^{\tilde{y}_i \underline{w}^T(i) \underline{x}_i}}$$

> STOCHASTIC GRADIENT DESCENT

(i) INITIALIZING $\underline{w}(0)$

(ii) ITERATE OVER DATA POINTS UNTIL CONVERGENCE:

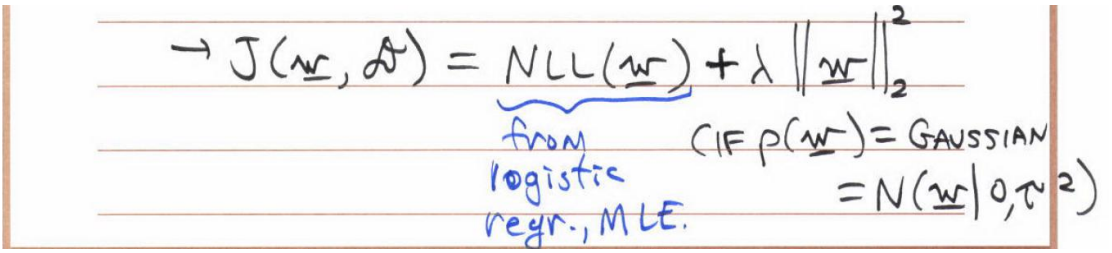
(a) RANDOMLY PICK A DATA POINT (WITH REPLACEMENT)

(b) PERFORM A SINGLE-SAMPLE UPDATE:

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

(i = ITERATION NUMBER)

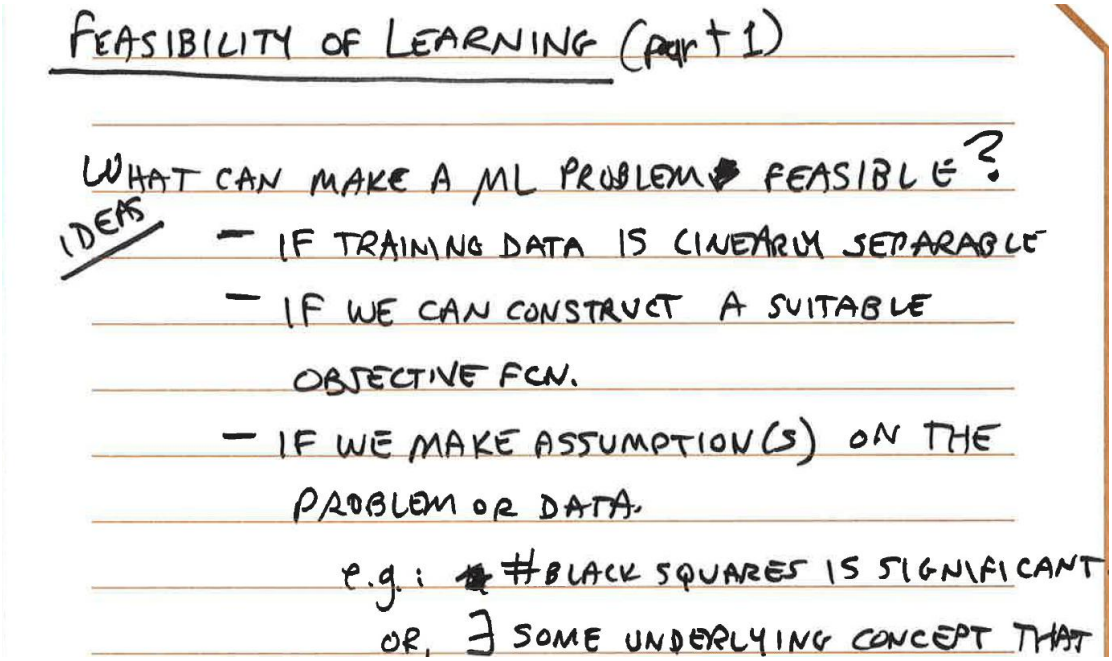
regularization



3.Complexity

feasibility of learning (theory of hypothesis set)

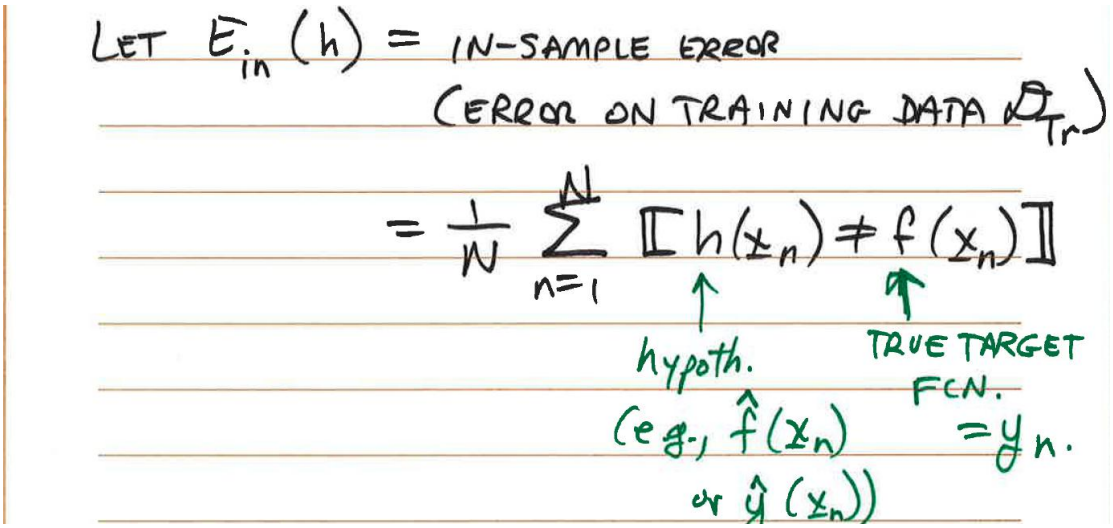
concept of feasibility



to constrain feasibility of learning: **choose hypothesis set**

general error

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



$E_{OUT}(h)$ = OUT-OF-SAMPLE ERROR
(ERROR ON UNKNOWNS)

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

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

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

WE CAN MEASURE E_{in} AND E_{Test}; WE WANT TO KNOW E_{OUT}.

Hoeffding inequity

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

HOW ACCURATE IS OUR ESTIMATE?

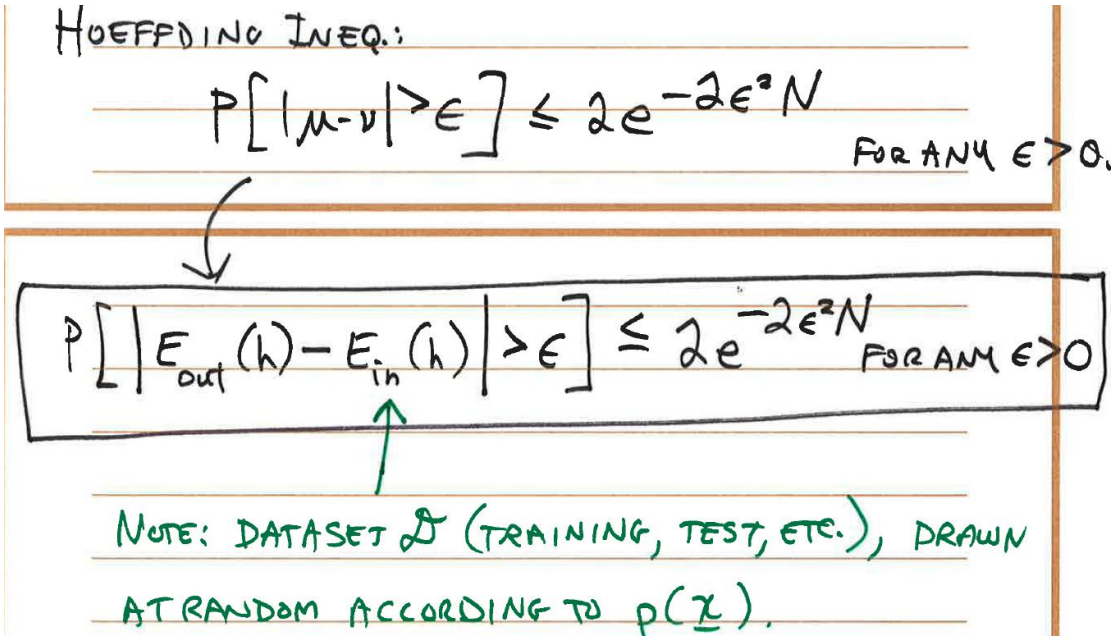
HOEFFDING INEQUALITY:

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

FOR ANY $\epsilon > 0$

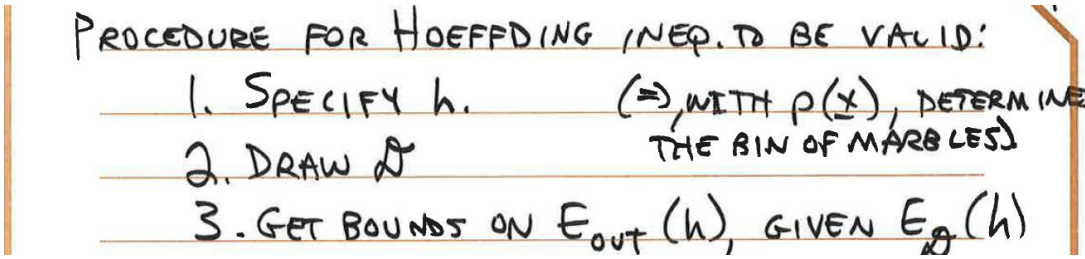
ϵ IS OUR "TOLERANCE"

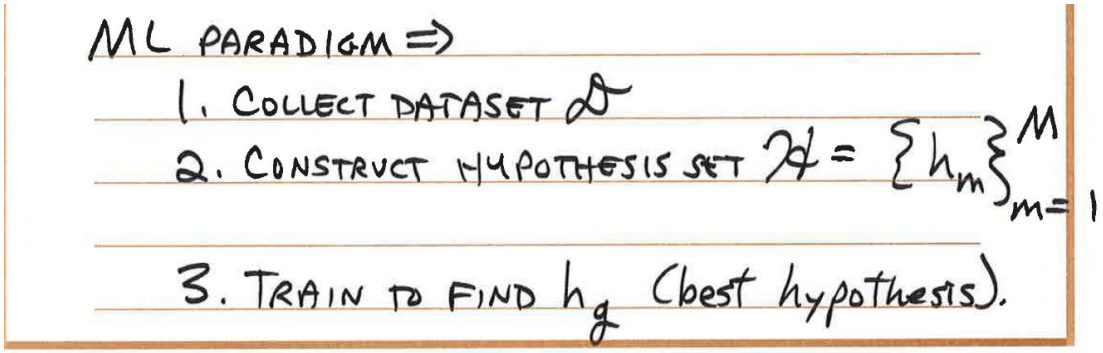
ν IS THE ONLY RANDOM QUANTITY.

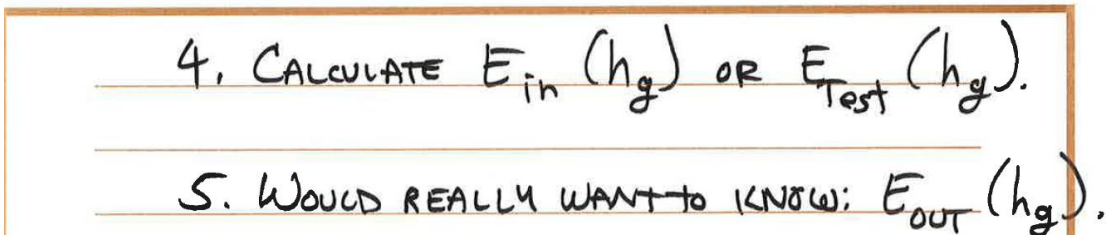


here we get the bound of model in hypothesis set(we assume the ideal best model that can predict the dateset perfectly) that show different between our output model and real word model

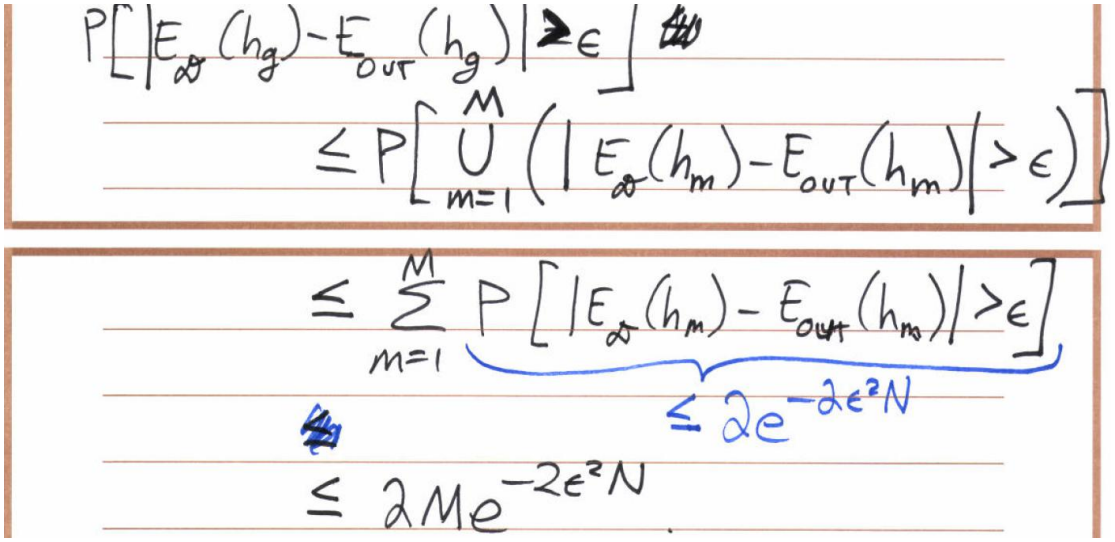
Training precedure

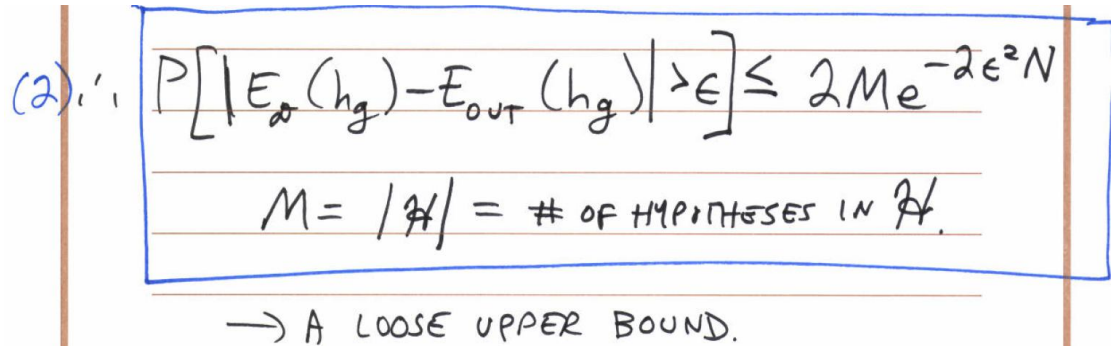




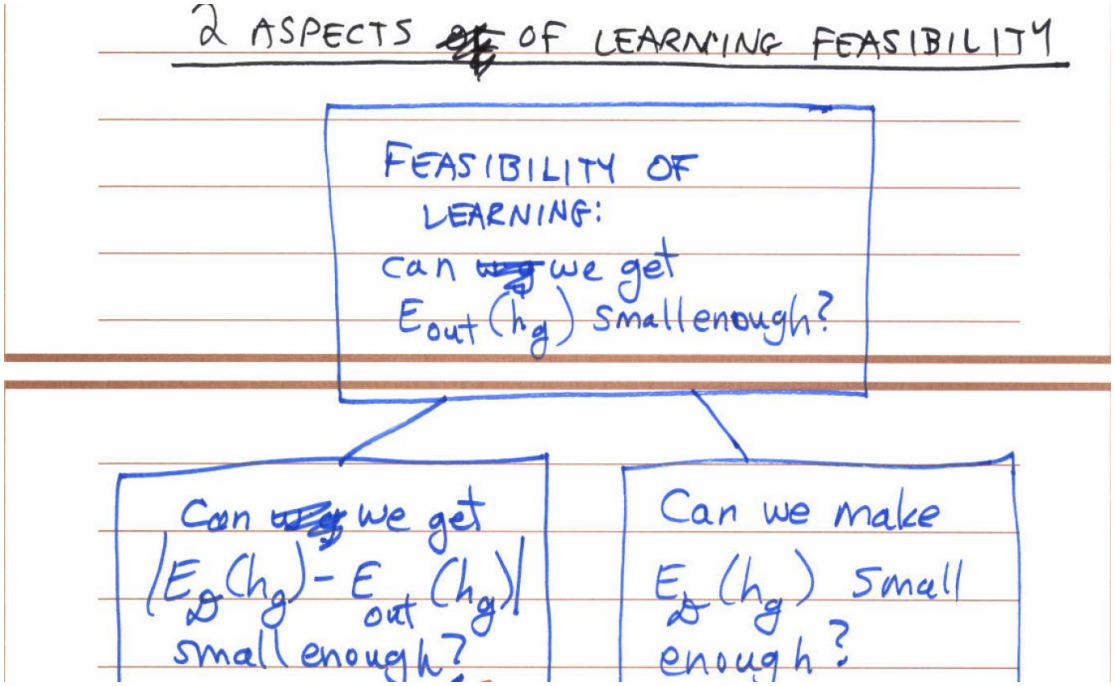


here change **h** to **hg** because we specified a hg that is one hypothesis in hypothesis set, a detailed one, that may not perfect, but is the best model we can get, can we want to find bound between it and

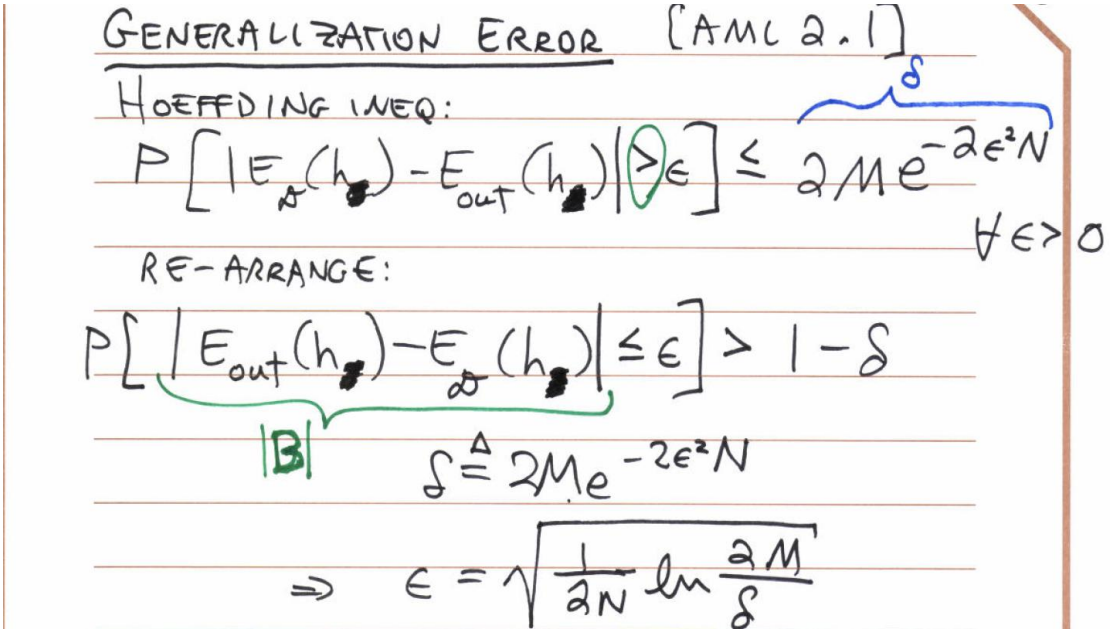


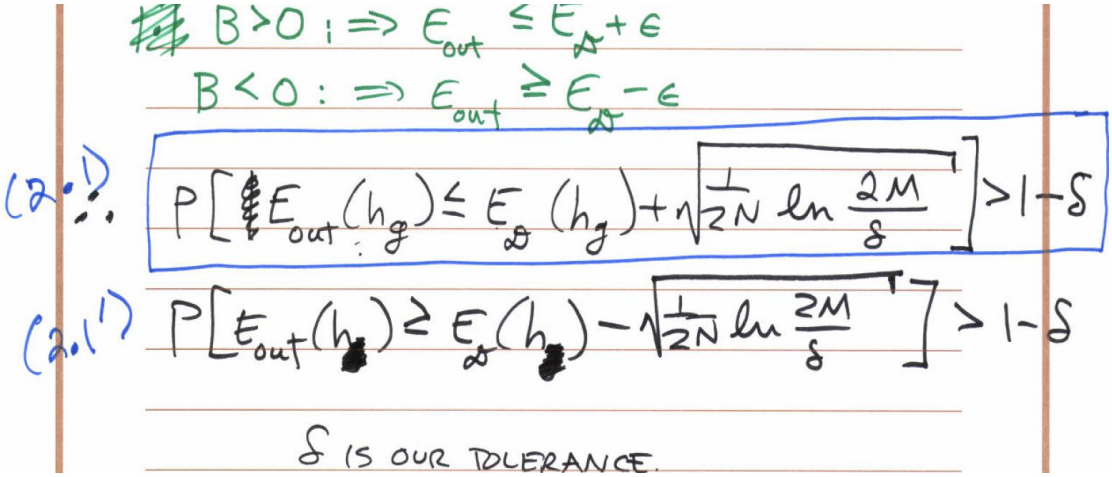


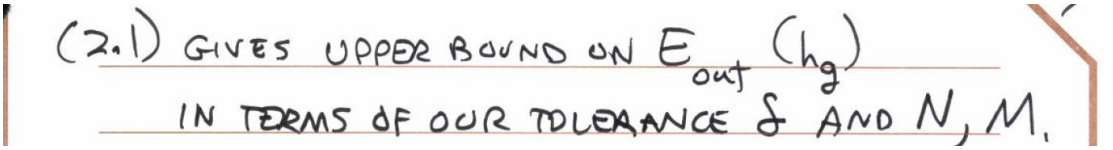
2 aspects of learning feasibility



Generalization Error







In this measurement, M--> # of hypothesis set is based on parameters, and perameters has infinite choices, so the **M = infinite** , makes the bound useless (bound = infinite)
need a better method to measure:

Dichotomies

DEF: THE SET OF DICHOTOMIES GENERATED BY $\mathcal{H}$ ON $\{x_n\}_{n=1}^N$:

$$\mathcal{H}(x_1, x_2, \cdots, x_N) = \{(h(x_1), h(x_2), \cdots, h(x_N)) \mid h \in \mathcal{H}\}$$

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

$|\mathcal{H}(x_1, \cdots, x_N)|$ = CARDINALITY OF $\mathcal{H}(x_1 \cdots x_N)$
= # OF DICHOTOMIES.

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

grow function

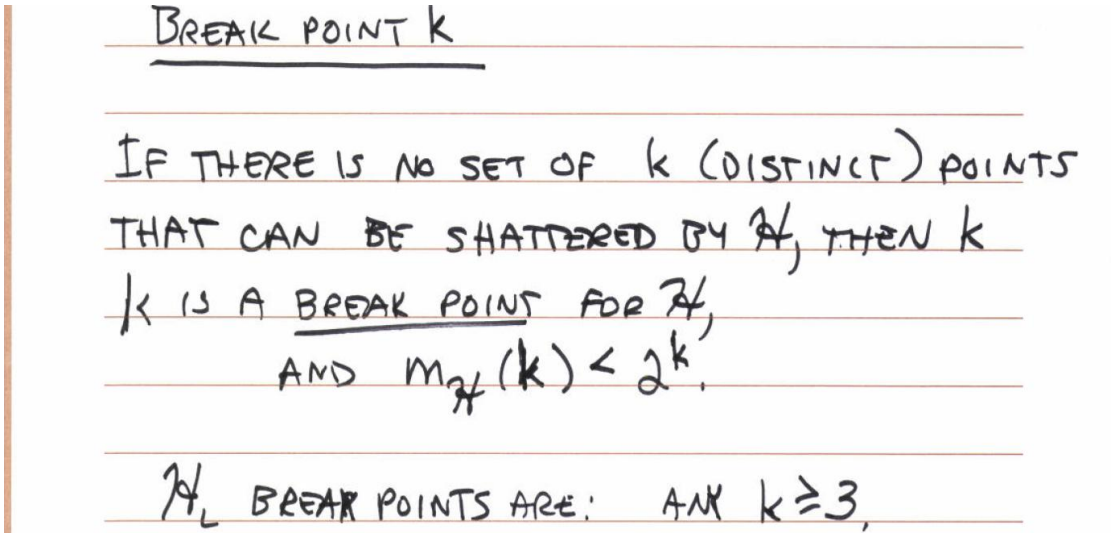
DEF: GROWTH FUNCTION FOR $\mathcal{H}$ IS:

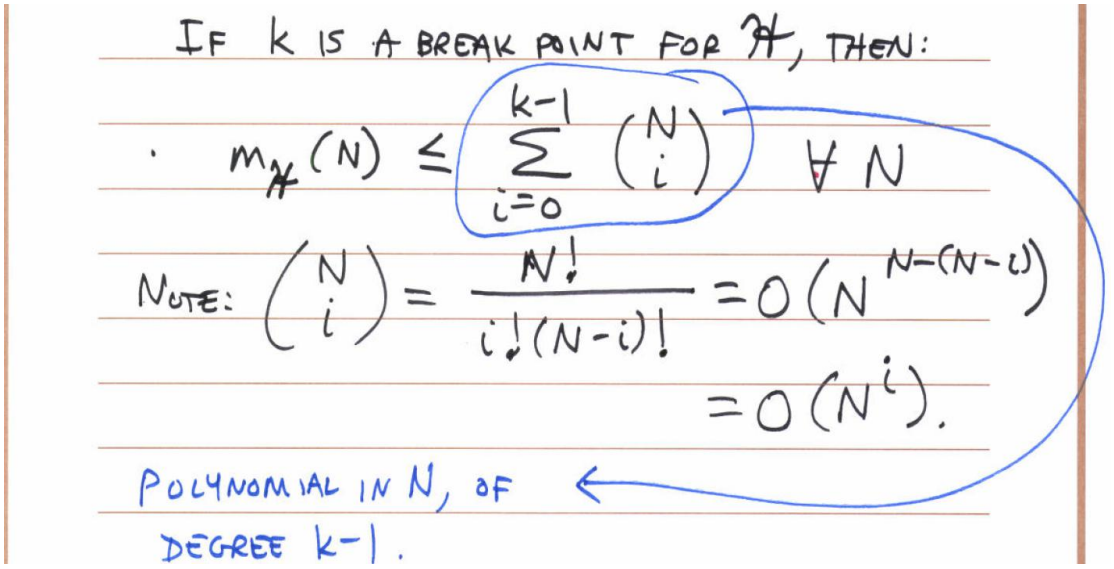
$$m_{\mathcal{H}}(N) = \max_{x_1, \cdots, x_N \in \mathcal{X}} |\mathcal{H}(x_1, \cdots, x_N)|$$

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

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

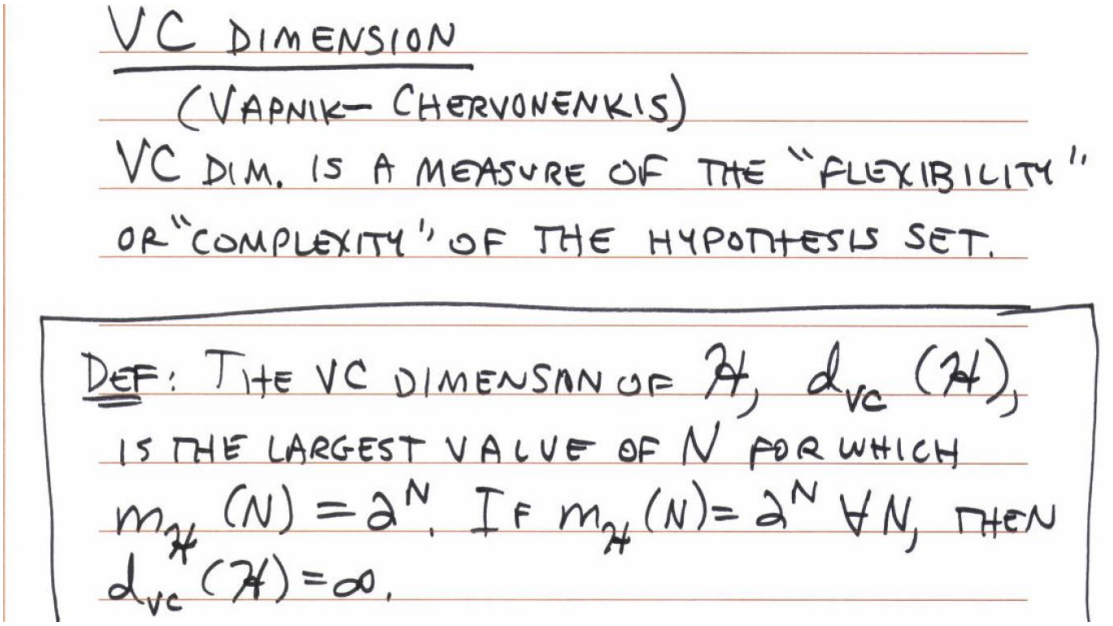
breaking point

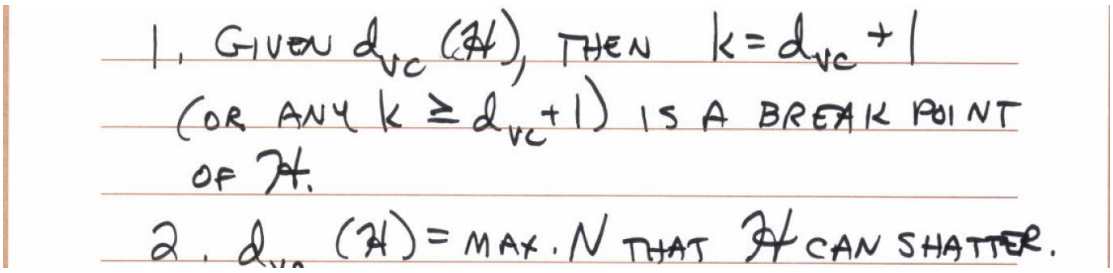




VC Dimension

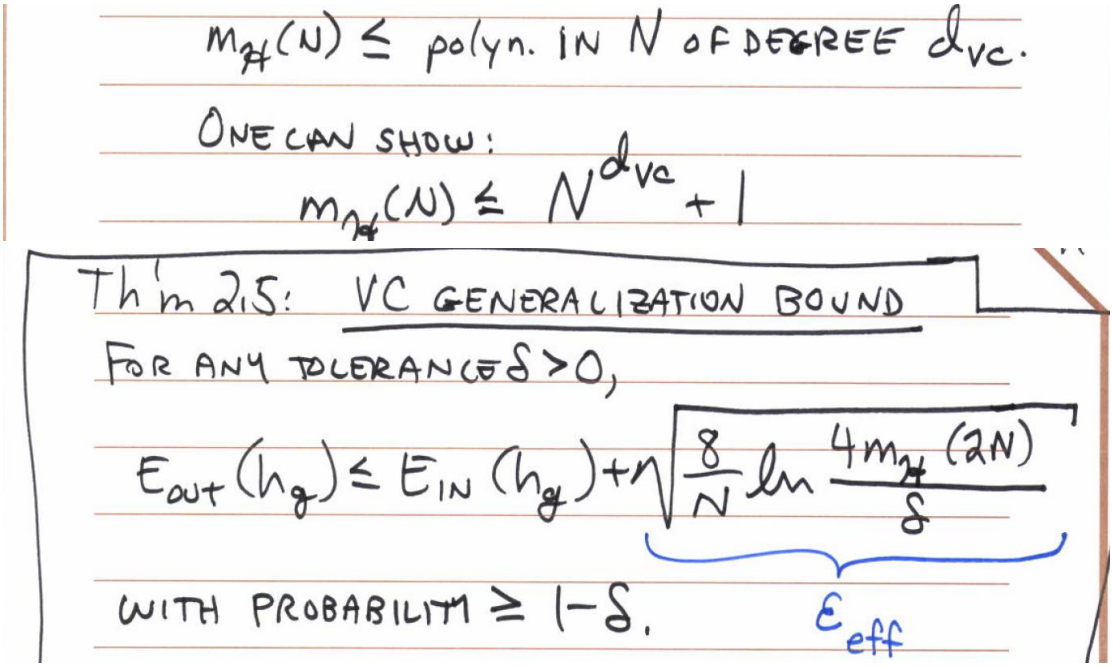
(all about discussion in sample error and out sample error)





use VC-dimension to calculate error bound

if we know VC-dimension--> can estimate # of hypothesis set based on # of date



VC generazation and understanding

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

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

IF $\mathcal{D}_{Test}$ HAS NOT BEEN USED TO PICK h_g OR $\mathcal{H}$, THEN:

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

WITH PROBABILITY $1-\delta$.

→ USE $M = 1$.

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

3. TH'M 2.5 AGAIN: (using $\mathcal{E}_{VC}$)

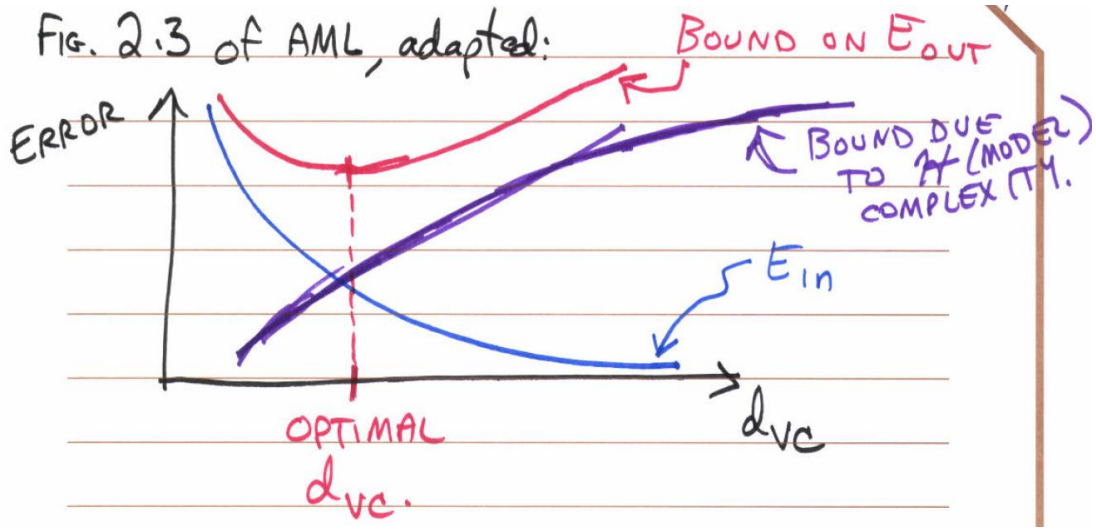
$$E_{OUT}(h_g) \leq E_{in}(h_g) + \sqrt{\frac{8}{N} \ln \frac{4[(2N)^{d_{VC}} + 1]}{\delta}}$$

d_{VC} ↓ typ. ↑

d_{VC} ↑ typ. ↓

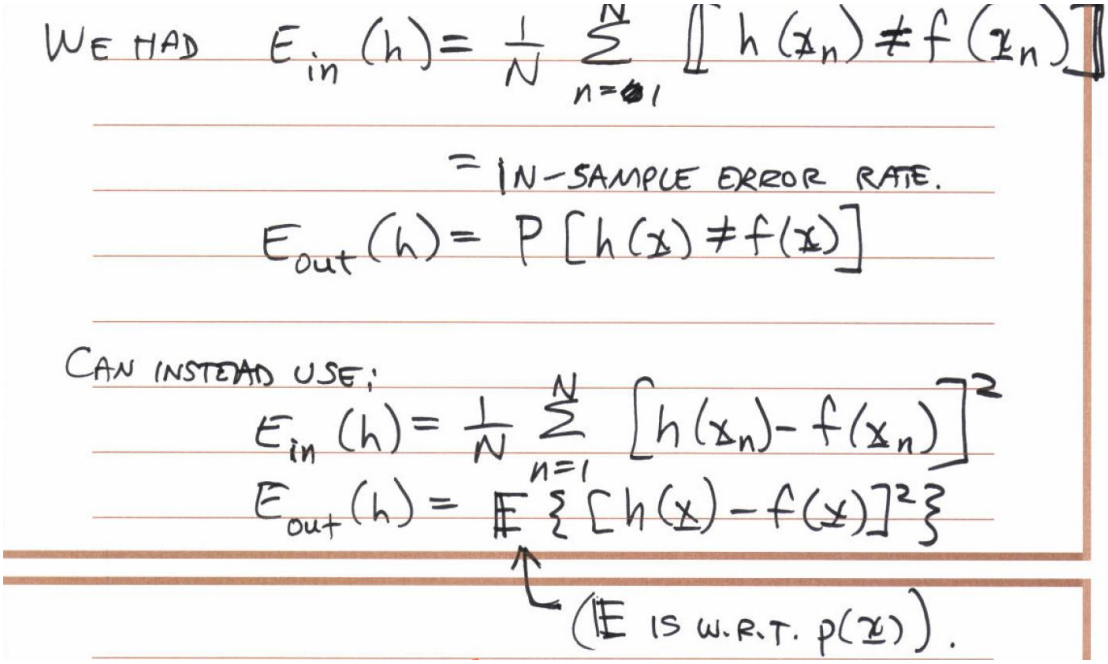
BOUND DUE TO MODEL COMPLEXITY (FOR A GIVEN N, δ).

FIG. 2.3 of AML, adapted:

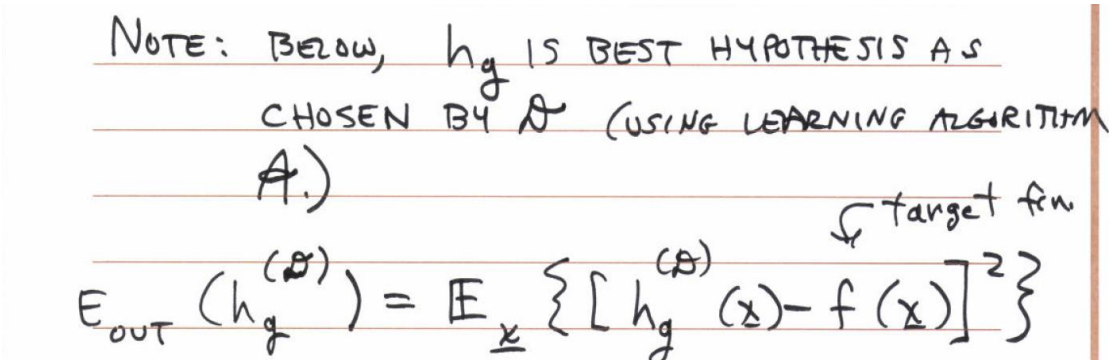


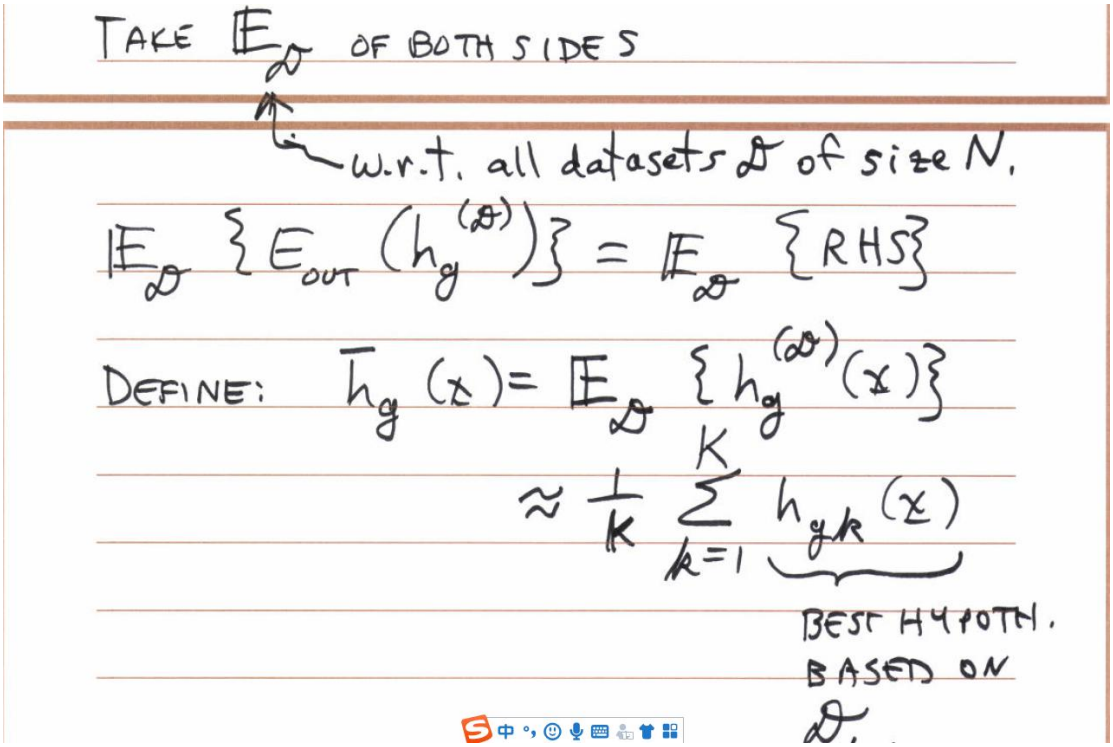
Bias vs Variance

concept: error measurement

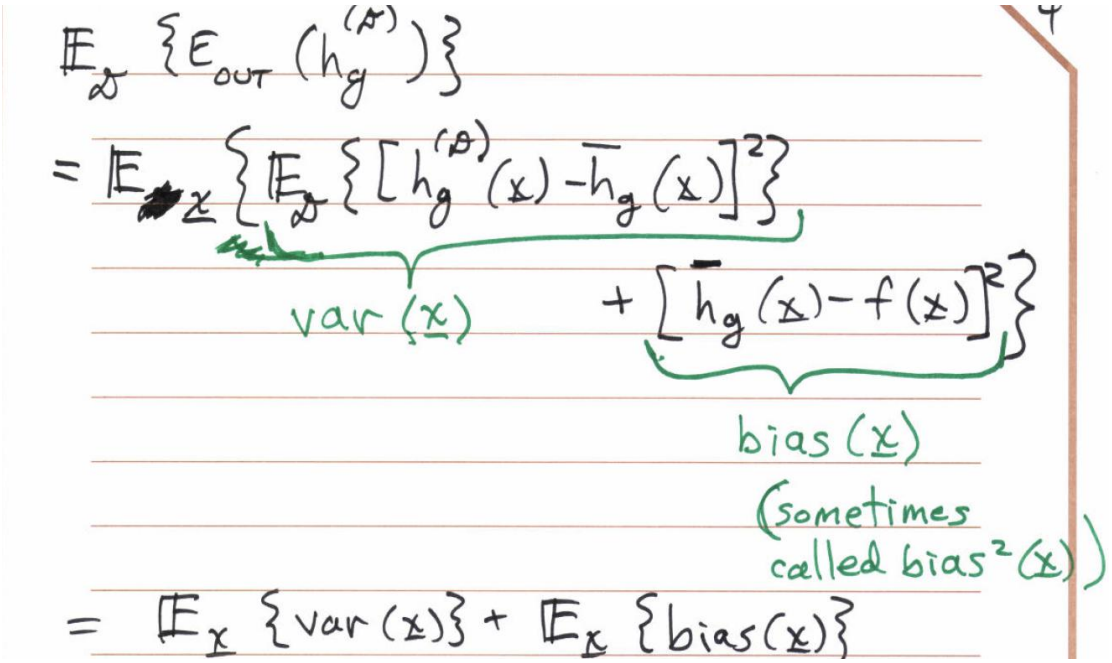


Bias vs. Variance: it's a trade off when considering model complexity
(different bias and variance when model complexity is different)





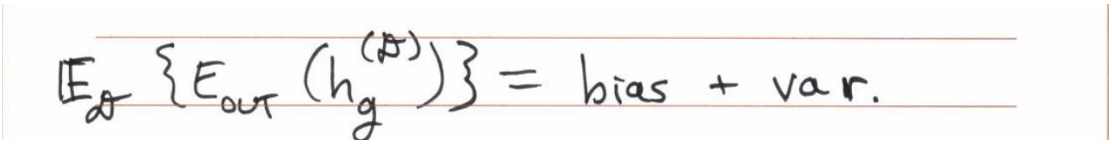
E(D){Eout(hg)} means built many hg(best model based on current dataset) and find average of these models to get the final best model(based on different dateset)



bias and variance:

bias: means different between predict result and target result

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



learning curve

BIAS-VAR VIEWPOINT —

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

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

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

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

VC VIEWPOINT —

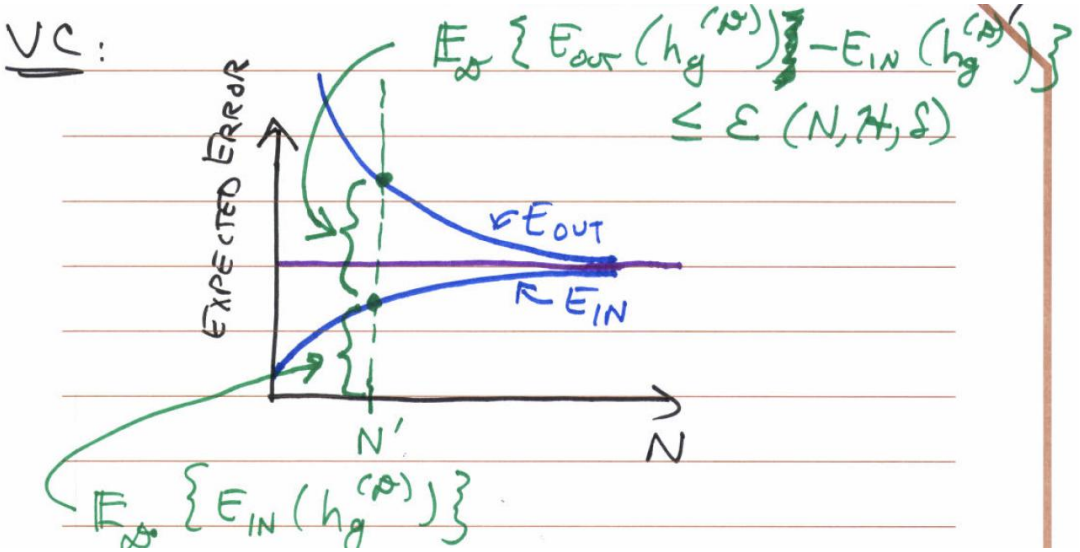
$$E_{OUT}\left(h_g^{(\mathcal{D})}\right) \leq E_{IN}\left(h_g^{(\mathcal{D})}\right) + \varepsilon(N, \mathcal{H}, \delta)$$

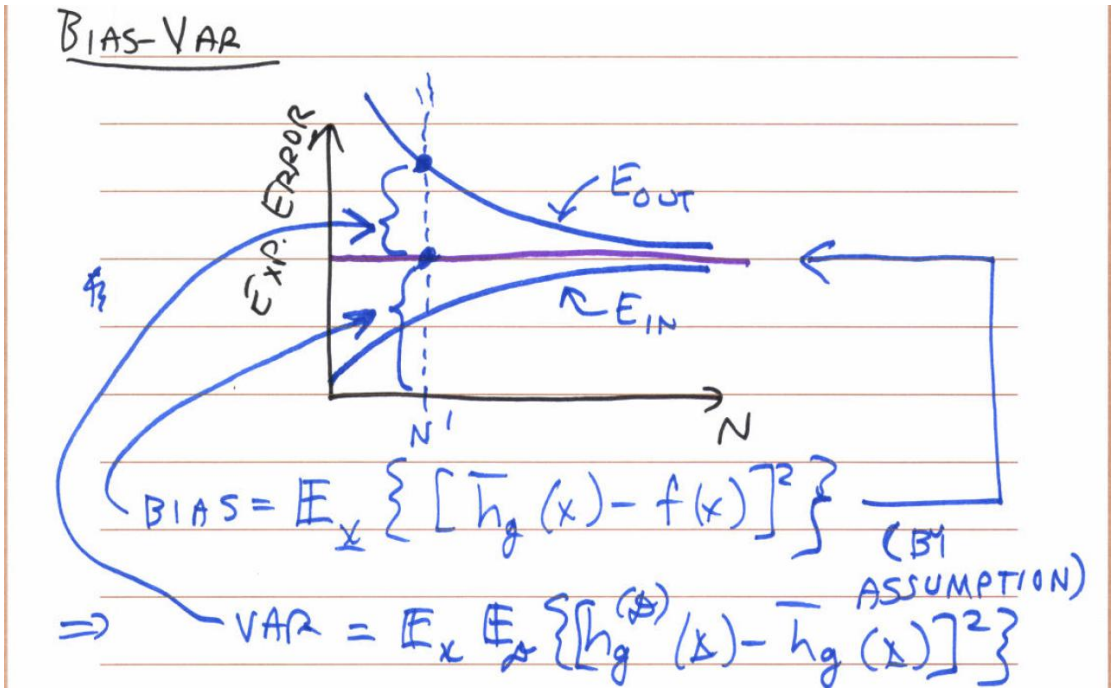
WITH PROBABILITY $\geq 1-\delta$.

$$\mathbb{E}_{\mathcal{D}}\left\{E_{OUT}\left(h_g^{(\mathcal{D})}\right)\right\} \leq \mathbb{E}_{\mathcal{D}}\left\{E_{IN}\left(h_g^{(\mathcal{D})}\right)\right\} + \varepsilon(N, \mathcal{H}, \delta)$$

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

VC:



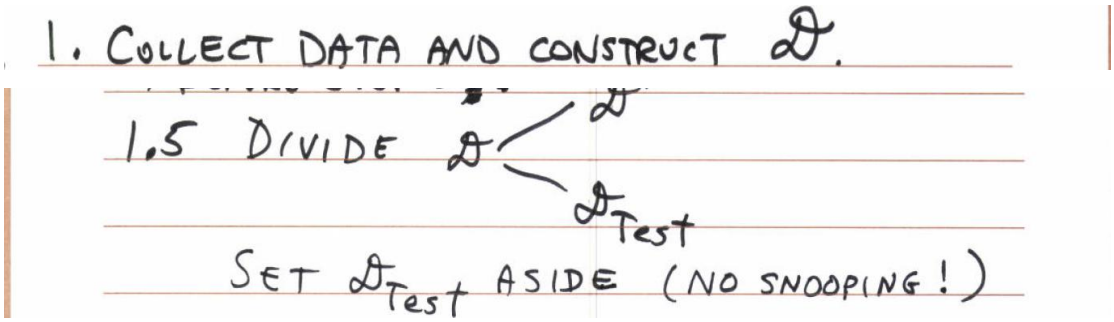


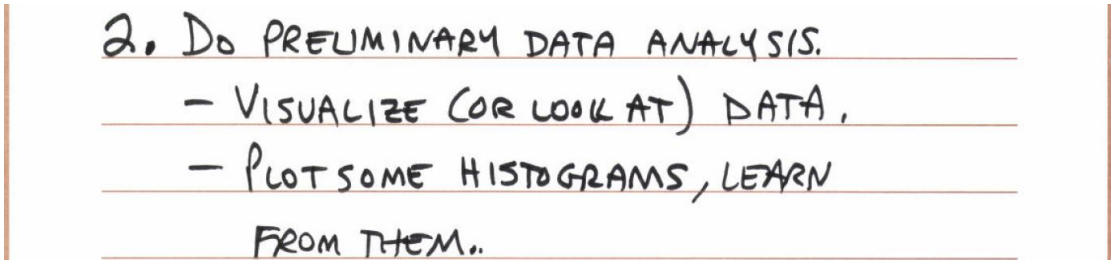
Id

4.Fundation of Model and Training procedure

Whole Training Method

method1:





3. PREPROCESSING

~~CHARACTER~~

- ~~NORMALIZE~~ NORMALIZE

⋮

4. FEATURE EXTRACTION.

- ASSESS WHETHER EXTRACTED FEATURES ARE USEFUL FOR CLASSIFICATION.

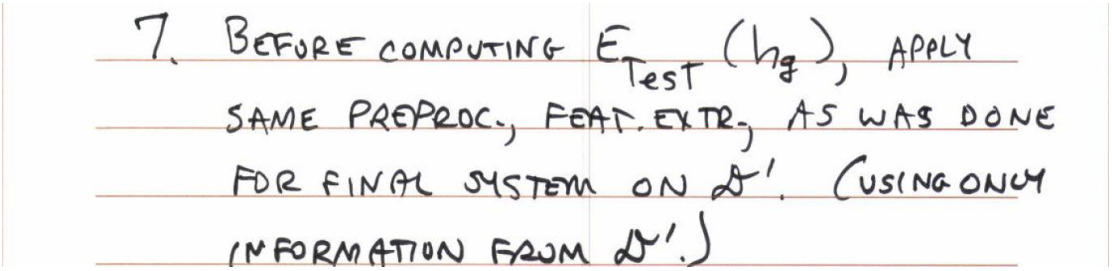
5. DO SOME PRELIMINARY TRIALS ON $\mathcal{D}$.

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

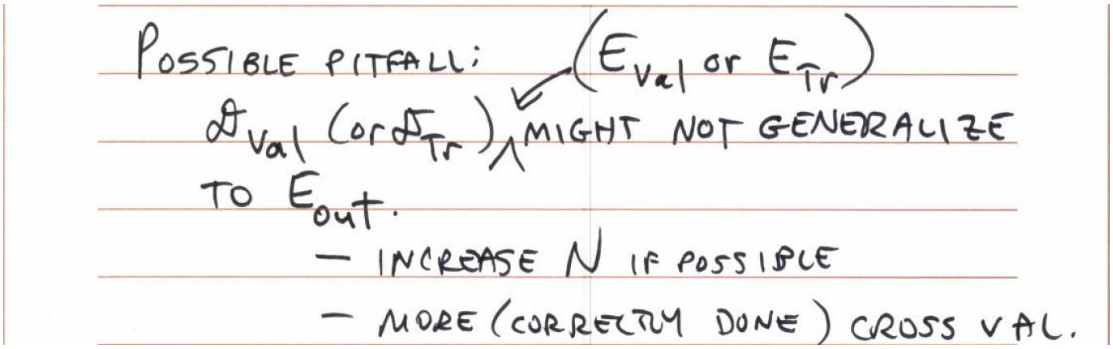
6. SET UP AND RUN MODEL SELECTION AND LEARNING ALGORITHMS.

- DECIDE ON $\mathcal{H}$

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

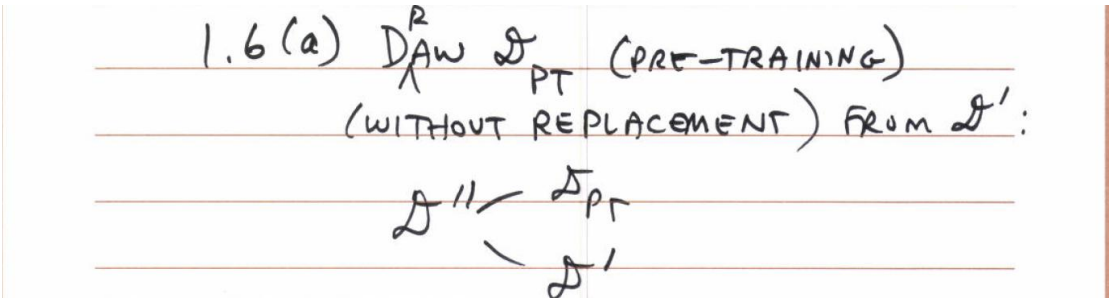


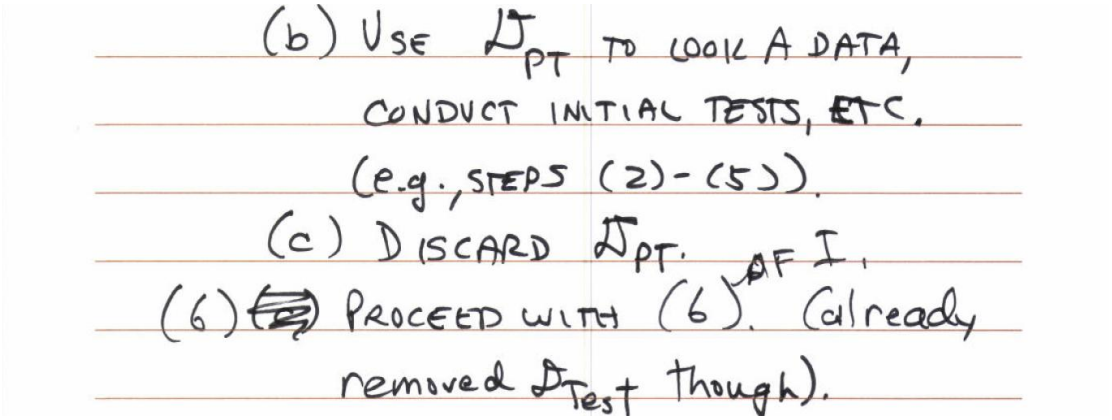
possible pitfall



method 2 -- add a pre-train set

step1--> 1.5





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

7. EVALUATE ITS PERFORMANCE:
(a) CALCULATE $E_{in}(h_g)$ USING $\mathcal{D}_{Tr}$ OR $\mathcal{D}_{Val}$
CALCULATE ε_{VC} USING d_{VC} OF $\mathcal{H}$ (ALSO N, δ).
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 $\mathcal{H}$ (CONSIDERING $d_{VC}(\mathcal{H})$, N_{Tr}, N_{Val}, ETC.)

(2) DRAW $\mathcal{D}'$, $\mathcal{D}_{Test}$
(3) SET ASIDE $\mathcal{D}_{Test}$
(4) USE $\mathcal{D}'$ FOR MODEL SELECTION, TRAINING, CHOOSING h_g.
(5) ONCE h_g IS FINAL, 7(a) AND 7(b) CAN BE USED.

method 4

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

KEY ASSUMPTIONS

(i) $E_{out}(h_g) \leq E_{\mathcal{D}_0}(h_g) + \sqrt{\frac{8}{N} \ln \frac{4[(2N)^{d_{vc}} + 1]}{\delta}}$

BASED ON $\mathcal{H}_0$.

ASSUMES:

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

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

$M = 1$.

ASSUME:

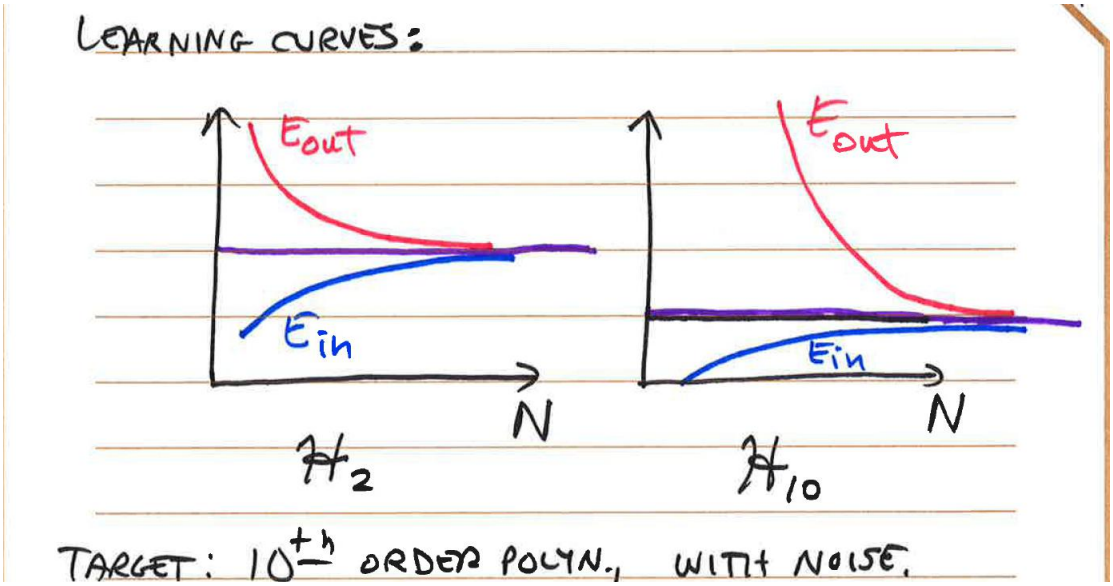
1. INFO. IN $\mathcal{D}_{Test}$ CAN'T INFLUENCE CHOICE OF h_g.
2. $N = N_{Test}$.

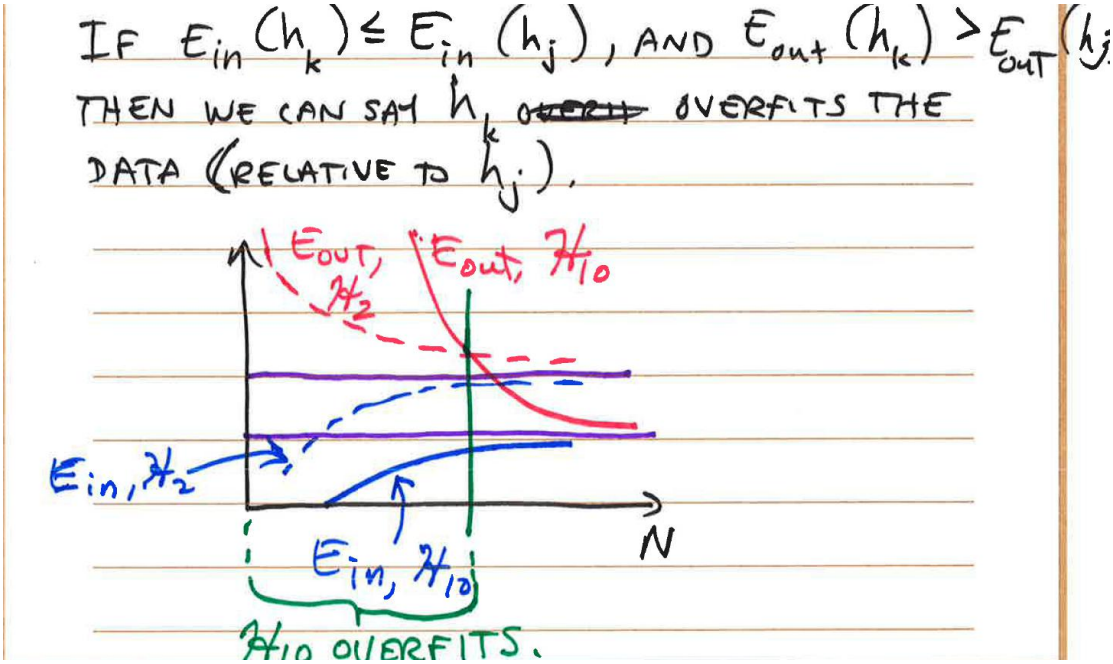
Overfitting

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

[OXFORD DICTIONARY]

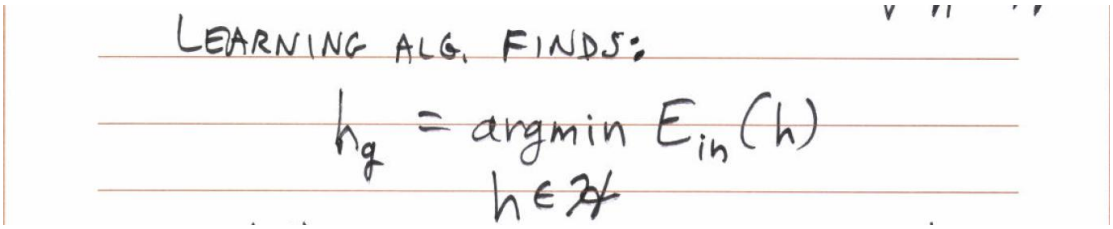
COMMON SYMPTOM OF OVERFITTING: PICKING A HYPOTHESIS WITH LOWER E_{in} RESULTS IN A HIGHER E_{out}.





Regularization

basic concept



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

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

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

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

(C IS A PARAMETER).

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

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

→ DIFFERENT HYP. SET.

diffierent conditions in Regularization

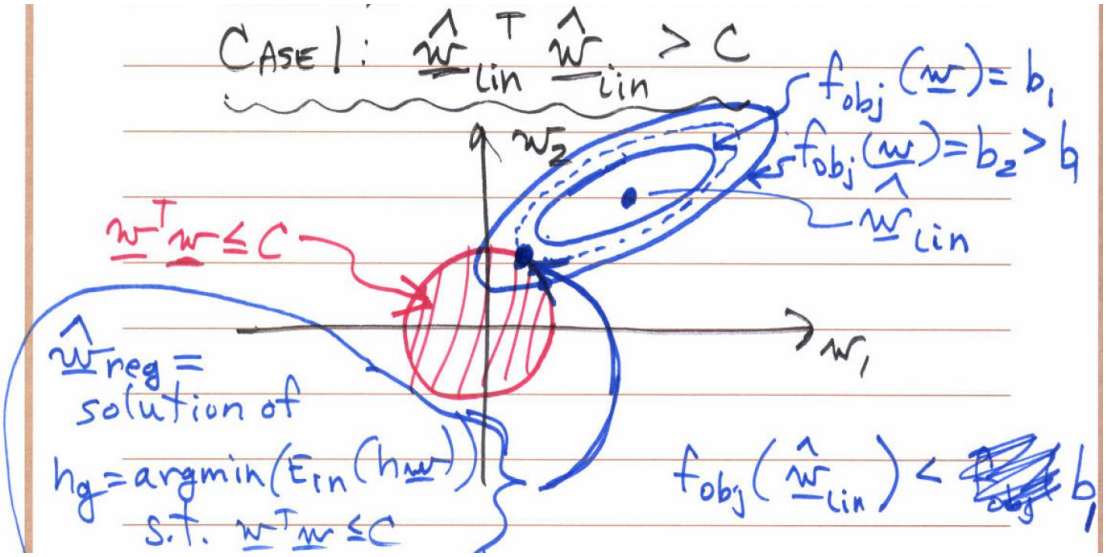
GRAPHICAL VIEW

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

example

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

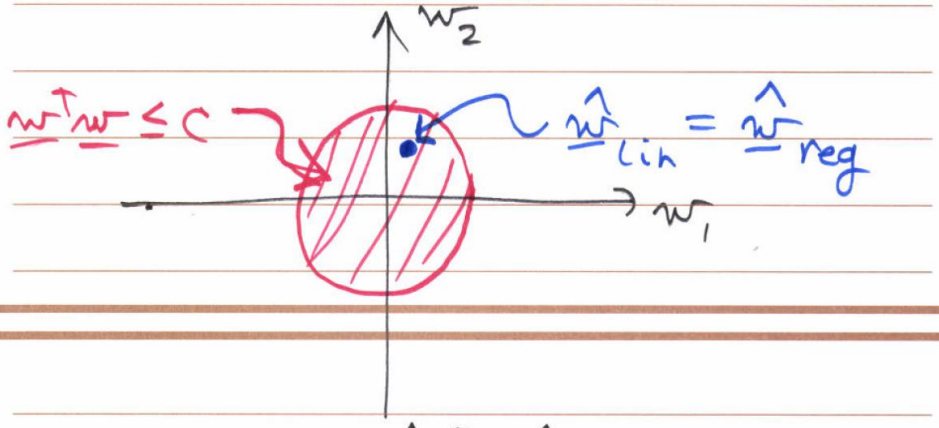
USING (i) ABOVE:



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

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

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



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

Regularization in a Lagrange way

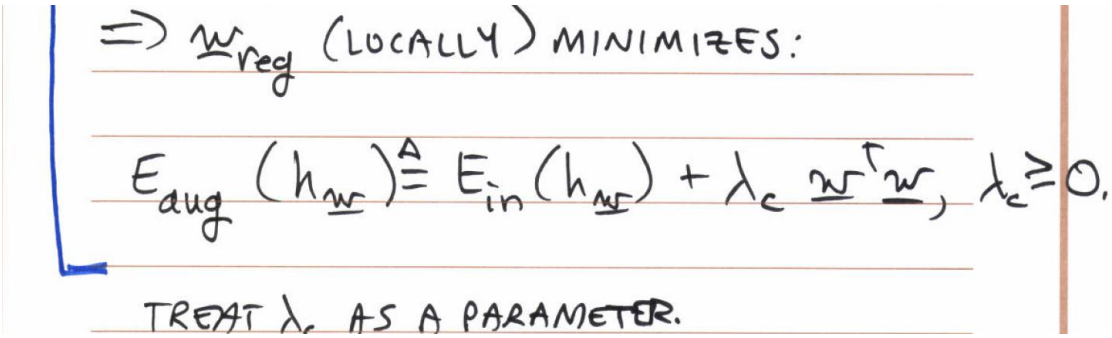
USE LAGRANGIAN OPTIMIZATION (WITH INEQUACITY CONSTRAINT):

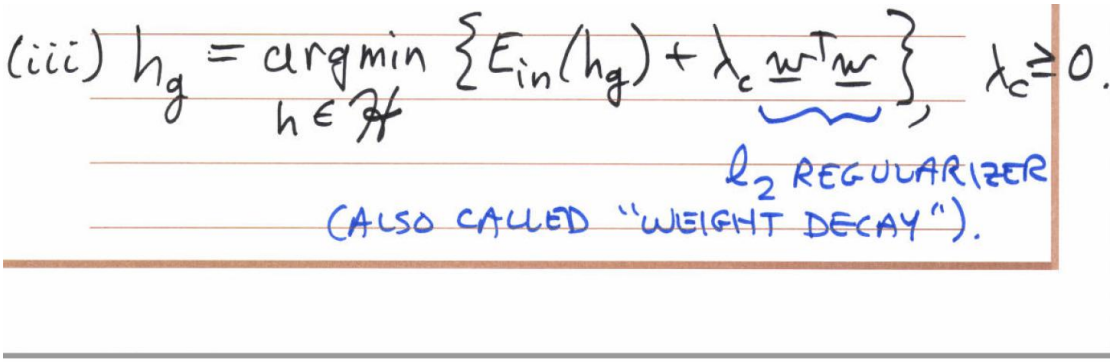
$$L(\underline{w}, \lambda_c) = E_{in}(h_{\underline{w}}) - \lambda_c \left[C - \underline{w}^T\underline{w} \right],$$

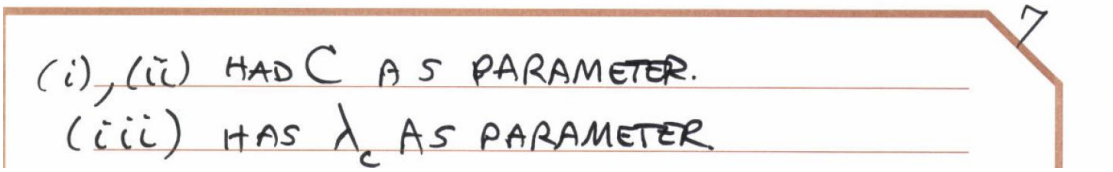
$\lambda_c \geq 0$,

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

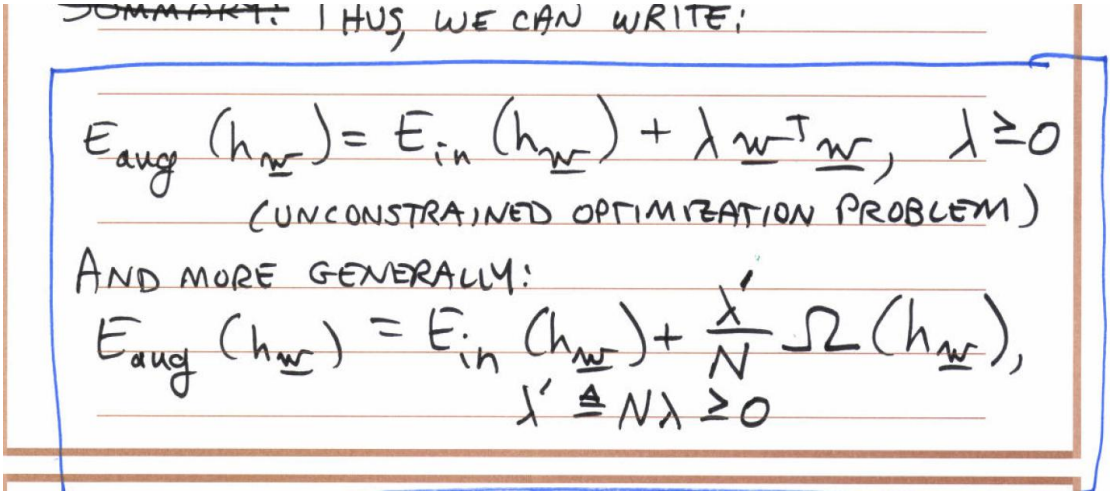
$$\nabla_{\underline{w}} \left\{ E_{in}(h_{\underline{w}}) + \lambda_c \underline{w}^T\underline{w} \right\}\Big|_{\underline{w}=\underline{w}_{reg}} = 0$$







final function of loss(objective)



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.

$$p(\underline{w}|\mathcal{D}) = \frac{1}{K} p(\mathcal{D}|\underline{w}) p(\underline{w})$$

$$= \frac{1}{K}\left[\prod_{i=1}^{N} p(y_i|\underline{x}_i, \underline{w})\right] p(\underline{w}) \quad (1)$$

$$= \frac{1}{K} p(\underline{y}|\underline{X}, \underline{w}) p(\underline{w})$$

IF LINEAR REGRESSION W/ USUAL ASSUMPTION:

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

PRIOR TERM:

$$p(\underline{w}) = p(\underline{w}|\lambda)$$

$$= \prod_{j=1}^{D} \text{Lap}(w_j|0, \frac{1}{\lambda})$$

$$= \prod_{j=1}^{D} \frac{\lambda}{2} e^{-\lambda|w_j|} \propto \prod_{j=1}^{D} e^{-\lambda|w_j|}$$

$$J(\underline{w}) = -\ln p(\underline{y}|\underline{X}, \underline{w}) - \ln p(\underline{w}|\lambda)$$

$$= NLL(\underline{w}) + \lambda \|\underline{w}\|_1,$$

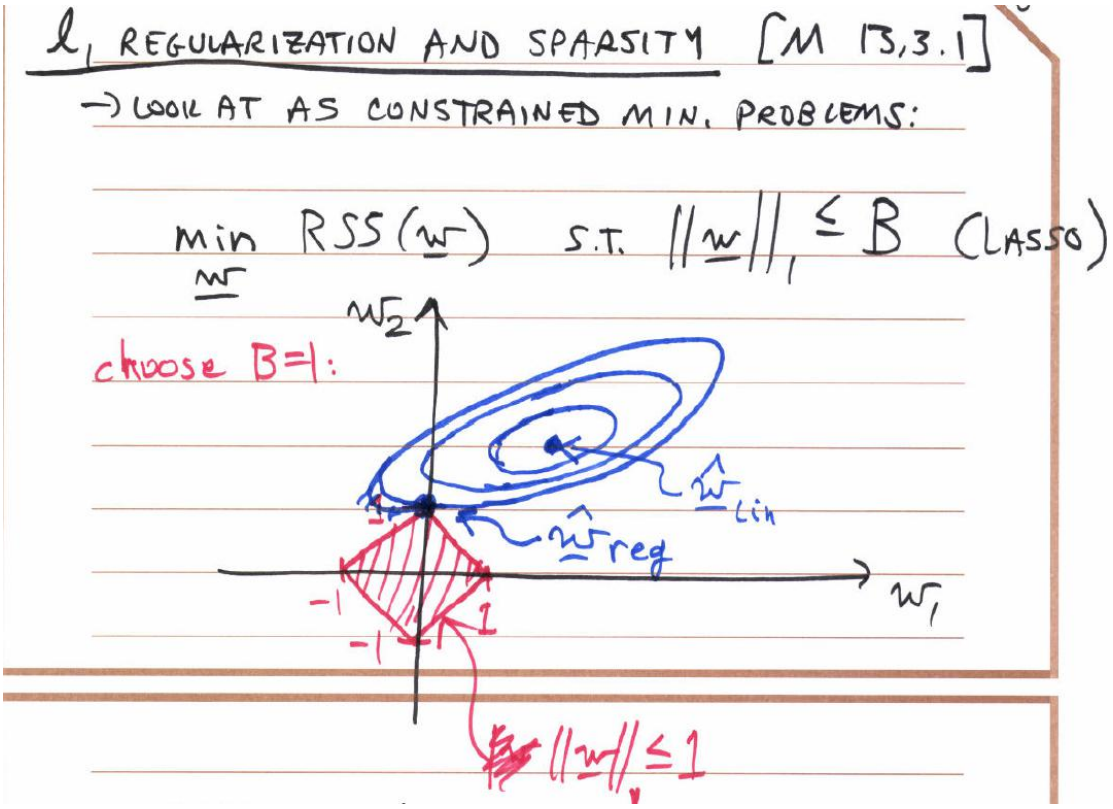
$$\|\underline{w}\|_1 \triangleq \sum_{j=1}^{D} |w_j|.$$

FOR LINEAR MODEL:

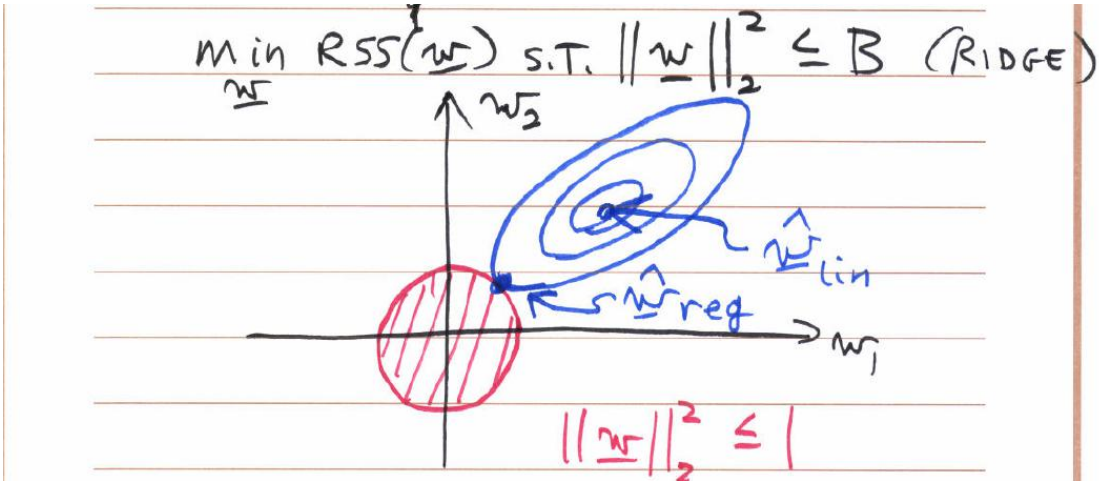
$$= \sum_{i=1}^{N} \frac{1}{2\sigma^2}\left[y_i - (w_0 + \underline{w}^T\underline{x}_i)\right]^2 + \lambda\|\underline{w}\|_1$$

$$J(\underline{w}) \propto RSS(\underline{w}) + \lambda'\|\underline{w}\|_1, \quad \lambda' = 2\lambda\sigma^2$$

L1 Regularization(Lasso)



L2 Regularization(Ridge)



bayesian variable feature selection

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

FIND $p(\underline{\gamma} \mid \mathcal{D})$.

$$p(\underline{\gamma} \mid \mathcal{D}) = \frac{p(\mathcal{D} \mid \underline{\gamma})\, p(\underline{\gamma})}{p(\mathcal{D})}$$

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

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

COMMON CHOICE:

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

π_0 = PROBABILITY THAT A FEATURE IS RELEVANT.

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

LET $\|\underline{\gamma}\|_0 \triangleq \sum_{j=1}^{D} \gamma_j$ = # 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:

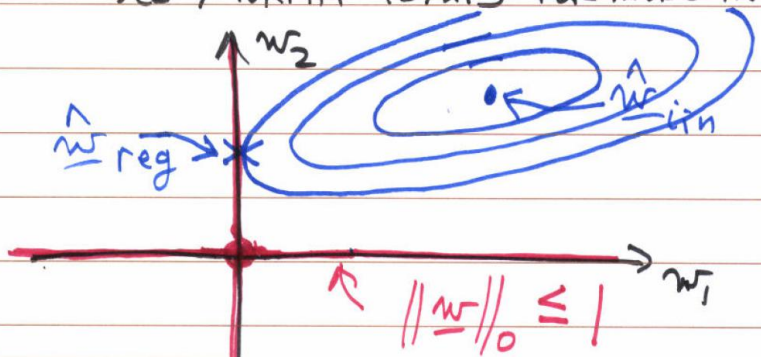
$$f_{obj}(\underline{w}) = \|\underline{y} - \underline{\underline{X}}\,\underline{w}\|_2^2 + \lambda \|\underline{w}\|_0$$

1. MIN. OVER CONTINUOUS VARIABLES $\underline{w}$ NOW.

2. Is f_{obj} CONVEX?

→ No, $\|\underline{w}\|_0$ IS NOT CONVEX.

→ SEE MURPHY 13.2.3 FOR MORE INFO [N.R.F.]



Validation

case1:

KEY RELATIONS AND BOUNDS

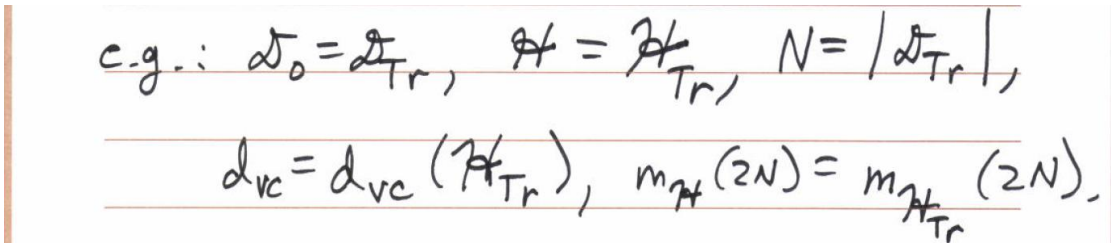
(i)

$$\left.\begin{aligned} E_{out}(h_g) &\leq E_{\mathcal{D}_0}(h_g) + \varepsilon_{eff} \\ E_{out}(h_g) &\leq E_{\mathcal{D}_0}(h_g) + \varepsilon_{vc} \end{aligned}\right\} \text{ with probability } \leq 1-\delta$$

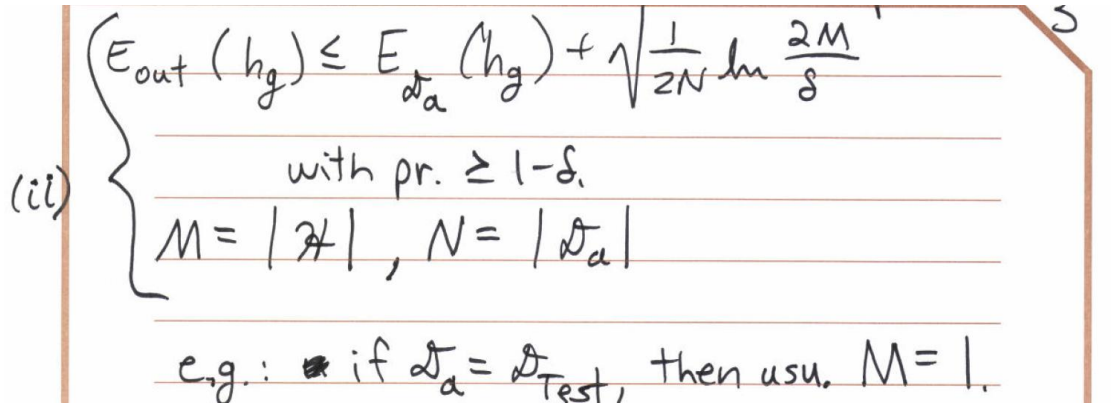
$$\varepsilon_{eff} = \sqrt{\frac{8}{N} \ln \frac{4\, m_{\mathcal{H}}(2N)}{\delta}}$$

$$\varepsilon_{vc} = \sqrt{\frac{8}{N} \ln \frac{4\left[(2N)^{d_{vc}} + 1\right]}{\delta}}$$

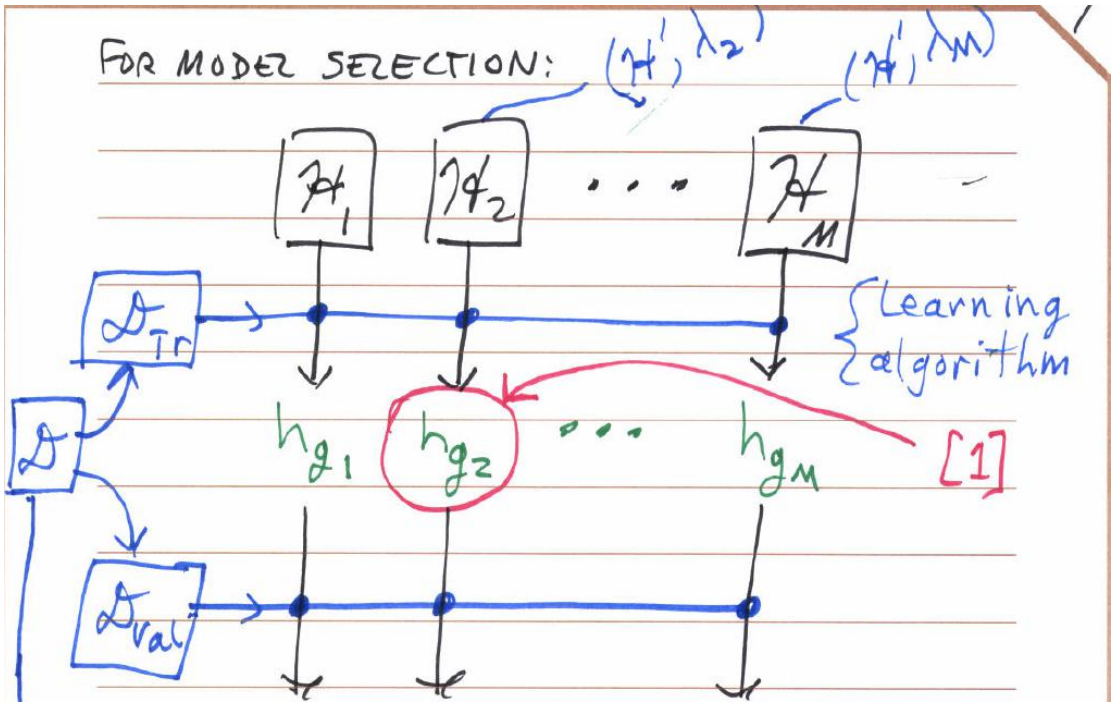
$$\varepsilon_{eff} \leq \varepsilon_{vc}, \quad N = |\mathcal{D}_0|$$

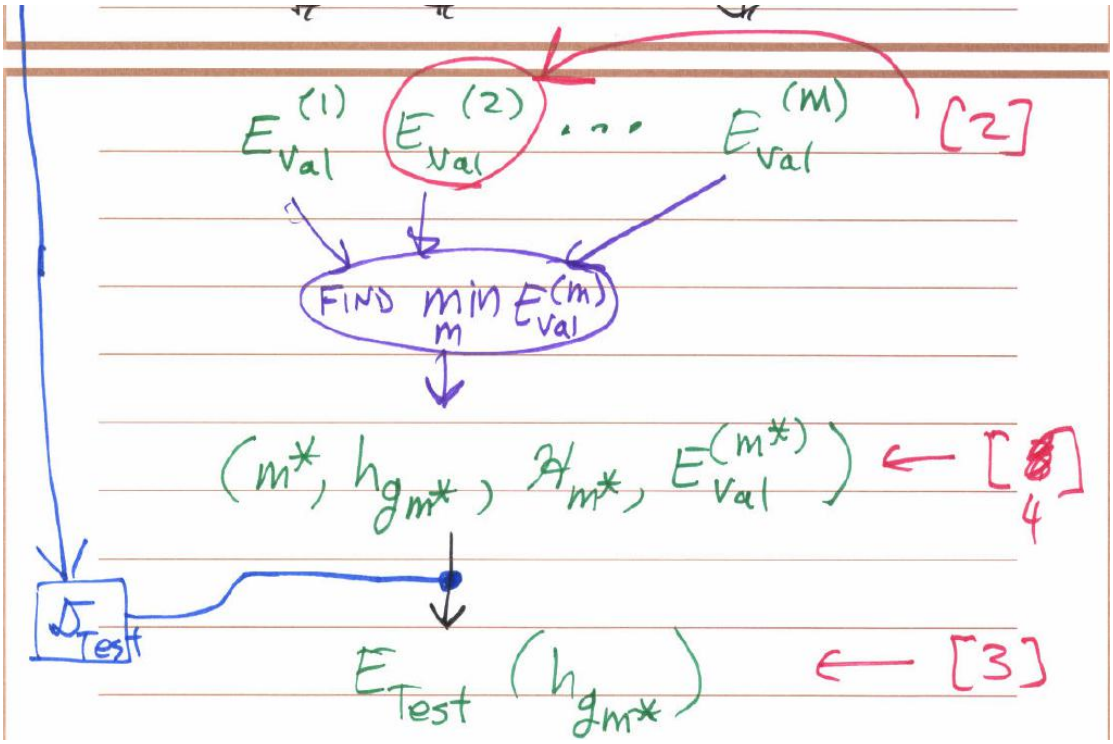


case2:



validation : for model selection (choose λ in regularization)





4 questions to understand training procedure:

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

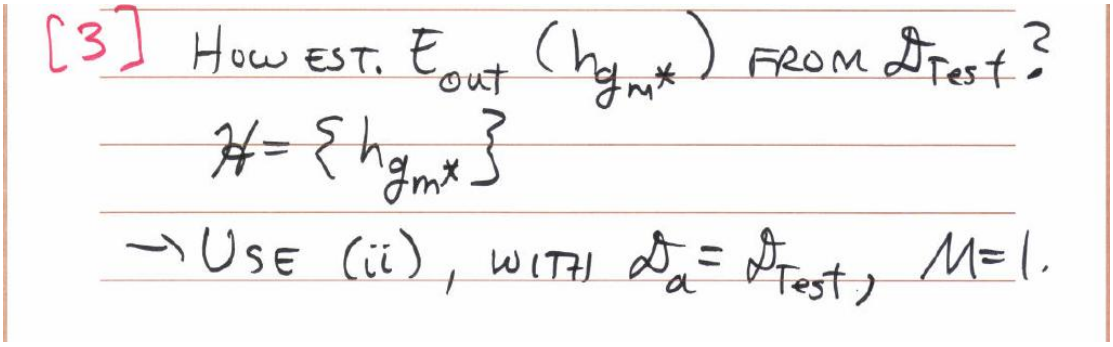
A: $\mathcal{H}_2$.

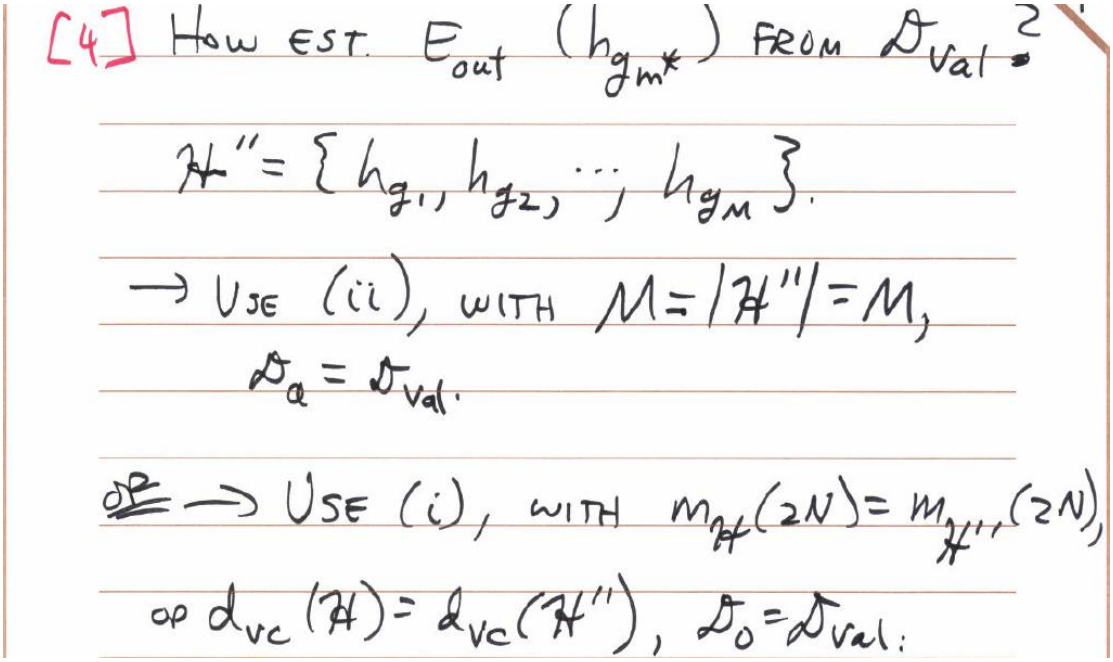
THEORETICAL BOUNDS $E_{out}(h_{g_2})$ USING $\mathcal{D}_{Tr}$?

→ USE (i) WITH $\mathcal{D}_0 = \mathcal{D}_{Tr}$,
$\mathcal{H} = \mathcal{H}_2$, $N = |\mathcal{D}_{Tr}|$
$d_{VC} = d_{VC}(\mathcal{H}_2)$, $m_{\mathcal{H}}(2N) = m_{\mathcal{H}_2}(2N)$

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

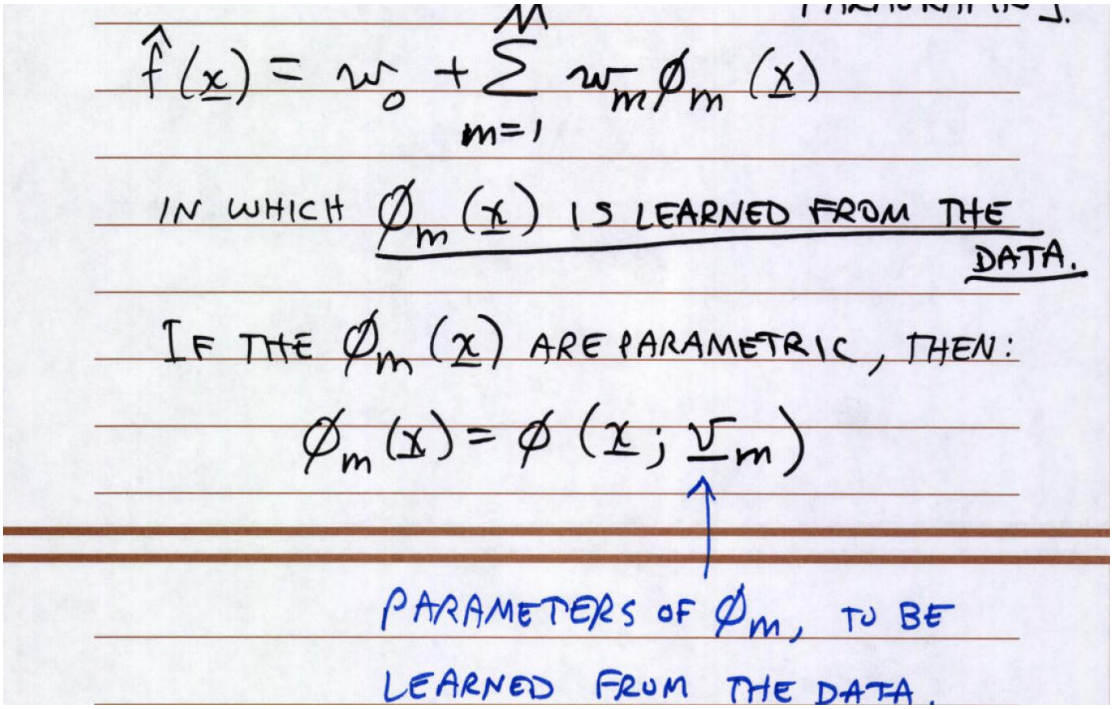
$\mathcal{H} = \{h_{g_2}\}$
→ USE (ii) WITH $\mathcal{D}_q = \mathcal{D}_{Val}$, $M = 1$.





5.Nonlinear Method

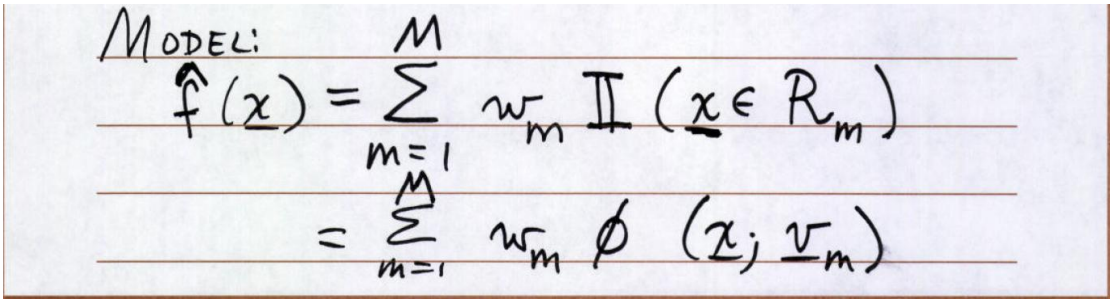
Adaptive basis function models



Decision Tree and Random Forest

Decision Tree (CART)

concept



IN WHICH R_m = m^{TH} REGION,

γ_m = PARAMETERS OF ϕ_m (LEARNED FROM DATA).

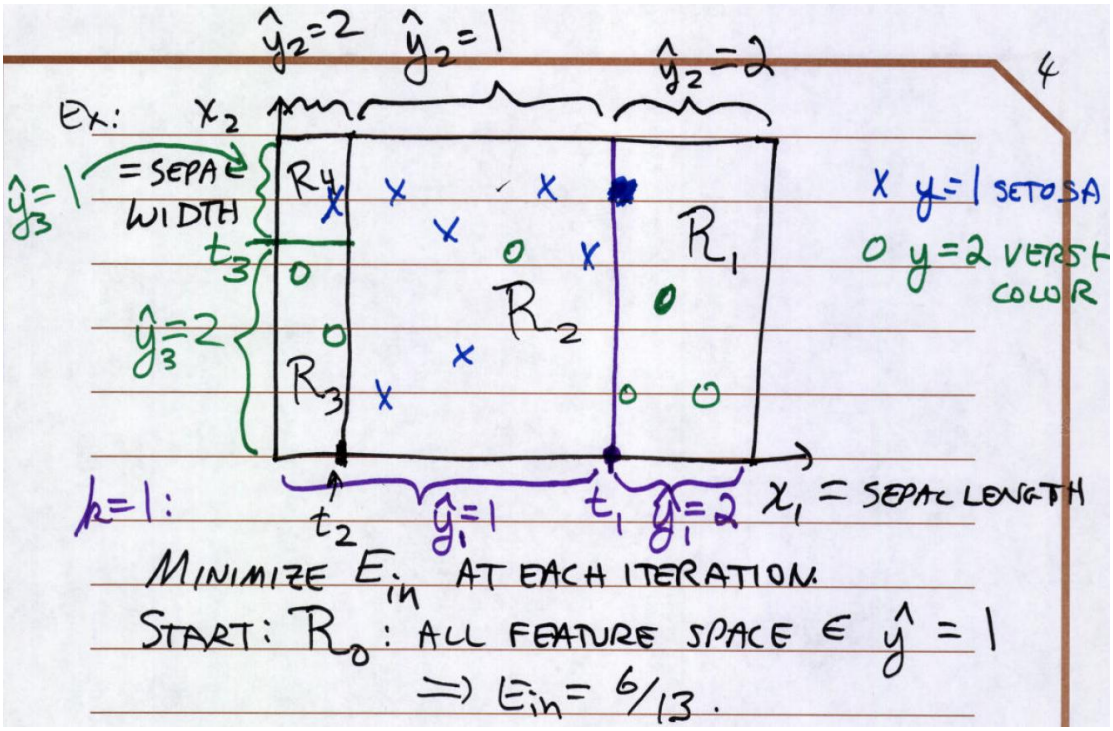
$\Rightarrow \hat{f}(x)$ IS A PIECEWISE CONSTANT APPROX. TO $f(x)$.

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 precedure

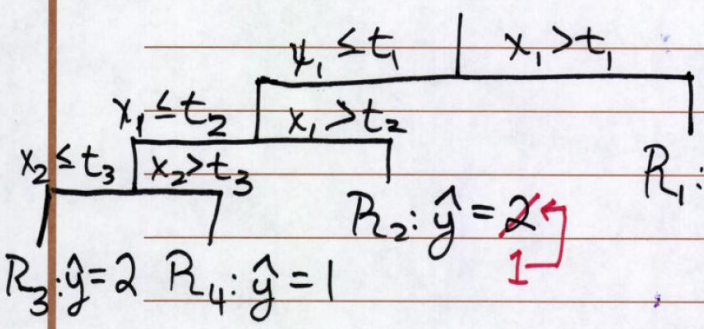


$k=1$: CHOOSE x_1 OR x_2 TO THRESHOLD
PICK THRESHOLD VALUE t_1
CHOOSE ~~CLAS~~ REGION LABELS.
$\Rightarrow E_{in} = 3/13$

$k=2$: 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:



∃ A VARIETY OF HALTING CONDITIONS, SUCH AS:
- MAX. DEPTH OF TREE.
- MIN. REDUCTION IN COST FCN. 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.

theoritical

MORE RIGOROUSLY:

AT EACH ITERATION (EA. NODE OF TREE), WE DIVIDE ONE REGION R_m INTO TWO, BY THRESHOLDING ONE FEATURE X_j; THUS:

AT k^{th} ITERATION:

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

IN WHICH:

$$f_{obj}^{(k)} = \text{cost}_k \{ (x_i, y_i) \in \mathcal{D} \}_{\text{after split}}$$

FOR COST FCNS. THAT ARE ADDITIVE BY REGION, OF R_m THAT IS:

$$\text{cost} \{ (x_i, y_i) \in \mathcal{D} \} = \alpha \sum_{m=1}^{M} \text{cost} \{ (x_i, y_i) \in R_m \}$$

WE CAN INSTEAD USE THE INCREMENTAL CHANGE IN COST:

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

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:

$$\text{cost}\{(\underline{x}_i, y_i) \in R_{m'}\} = \sum_{\underline{x}_i \in R_{m'}} (y_i - w_{m'})^2$$

FOR GIVEN $R_{m'}$, THIS IS MINIMIZED BY:

$$w_{m'} = w_{m'}^* = \bar{y}_{R_{m'}} \triangleq \frac{1}{N_{R_{m'}}} \sum_{\underline{x}_i \in R_{m'}} y_i$$

NOTE: TO SAVE ON COMPUTATION, CART TYPICALLY CYCLES THROUGH ALL REGIONS R_m, $m = 1, 2, \cdots$, 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.:

$$\text{cost}\{(\underline{x}_i, y_i) \in R_{m'}\} = \frac{1}{N_{R_{m'}}} \sum_{\underline{x}_i \in R_{m'}} \mathbb{I}\left[y_i \neq \hat{y}(R_{m'})\right]$$

$\hat{y}(R_{m'})$: 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 σ).

CART SUMMARY [CM 16.2.4]

PROS	CONS
• ALGORITHM IS FAIRLY SIMPLE	• PREDICTIVE ACCURACY OFTEN ISN'T AS GOOD AS WITH SOME OTHER MODELS.
• INCORPORATES LINEAR AND NONLINEAR BOUNDARIES ON ITS OWN.	• CAN BE UNSTABLE TO SLIGHT CHANGES IN DATA.
• CAN INCLUDE SOME AUTOMATIC FEATURE SELECTION.	

Random Forest

(a) DRAW MANY DATASETS FROM $\mathcal{D}_{Tr}$, 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 [Alg. 15.1].

(a) Draw a sample dataset $\mathcal{D}^*$ at random, with replacement, of size N^*, from $\mathcal{D}_{Tr}$. Typically, $N^* = N_{Tr}$.

(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 (~~For~~ from all D features)

- Common values: $d \approx \sqrt{D}$, or $d = \lfloor \sqrt{D} \rfloor$

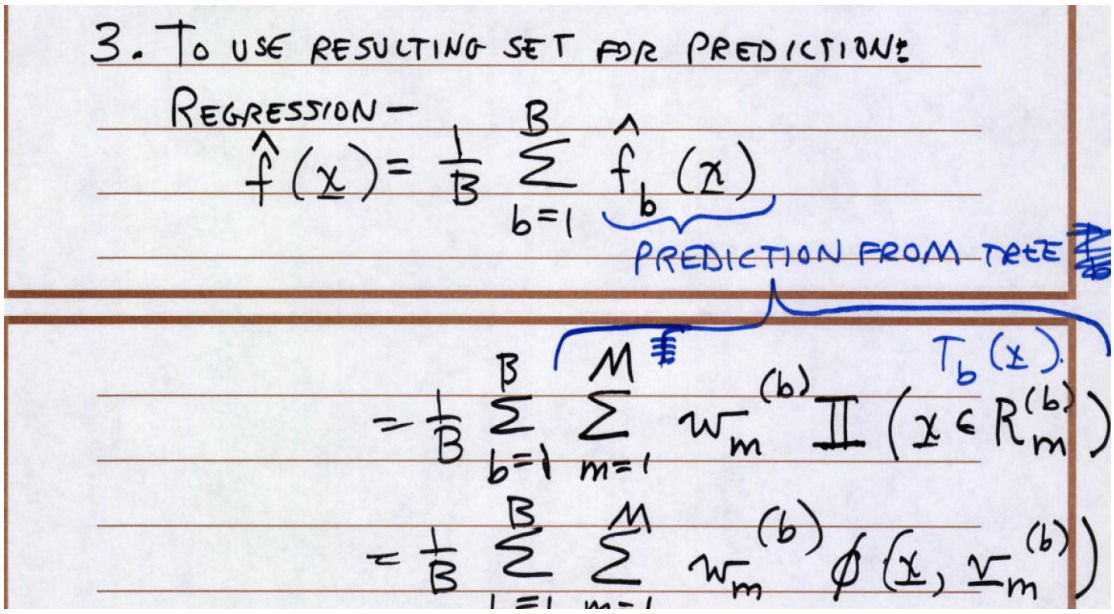
(ii) Use CART approach to split R_m by finding optimal $j, t_k, \underline{w}$

(iii) Split the tree node into two daughter nodes.

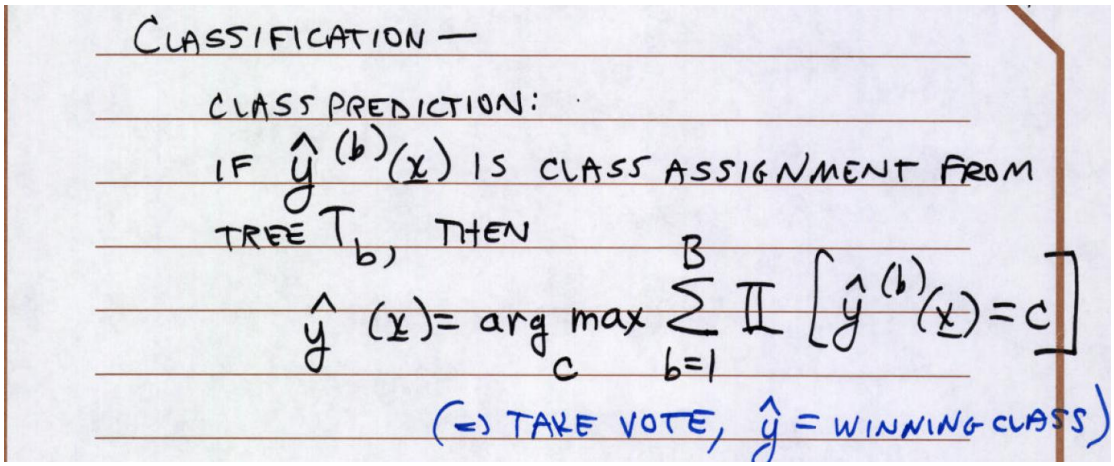
(iv) Iterate until a halting condition is reached

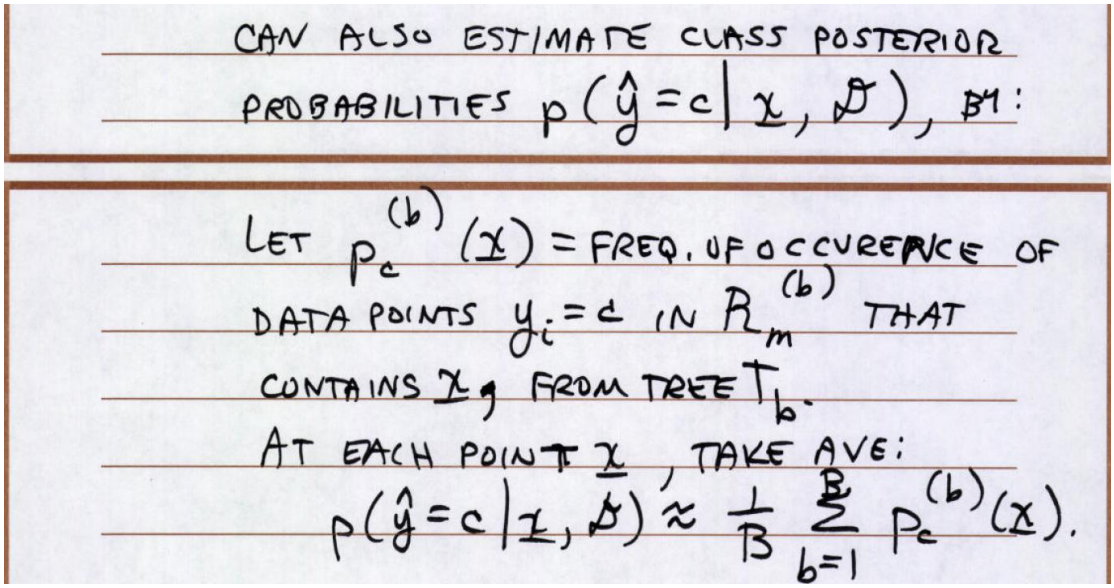
- See CART halting conditions

2. Output the set of trees $\{T_b, b = 1, 2, \cdots, B\}$



to predict result





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

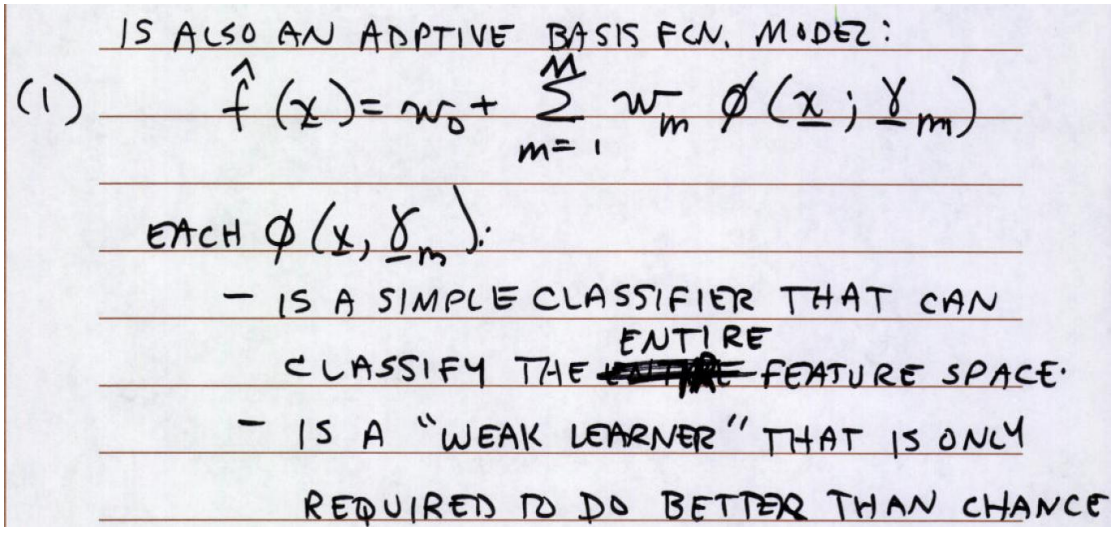
Information gain decides which **feature should be used to split** the data.

Information Gain = Entropy(parent) – [Average entropy(children)]

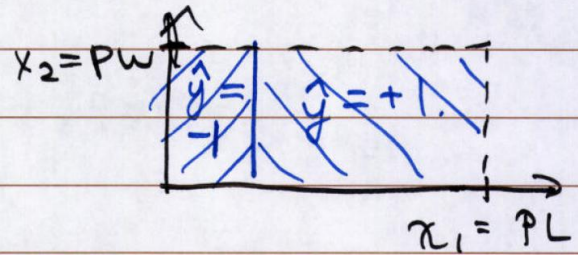
$$\text{Entropy} = \sum_{i} -P_i(\log_2 P_i)$$

P_i is probability of class i

Boosting



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 BE ±1.

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

"WEIGHT" OR IMPORTANCE OF m^{th} CLASSIFIER ($\beta_m \in \mathbb{R}$)

SIMPLE BASE CLASSIFIER $\phi_m \in \{-1, +1\}$.

$$= f_0 + \sum_{m=1}^{M} \beta_m \phi(\underline{x}; \underline{\gamma}_m)$$

INTERMEDIATE EXPRESSIONS FOR $\hat{f}$:

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

DATA POINTS: $(\underline{x}_i, \tilde{y}_i)$, $\tilde{y}_0 \in \{-1, +1\}$.

TO FIND EACH $\phi_m(\underline{x})$,

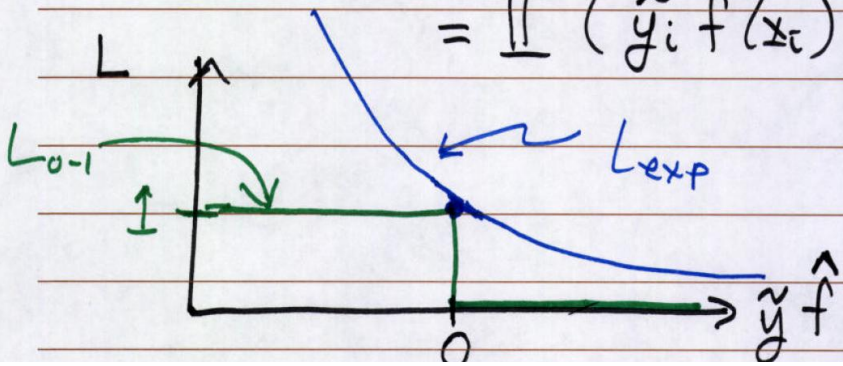
$$f_{obj}\left[\{\beta_m, m=1, \cdots, M\}, \{\underline{\gamma}_m, m=1, \cdots, M\}, \phi(\underline{x})\right]$$

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

↑ LOSS FUNCTION.

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

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



f_{obj} using L_{0-1}: - NOT CONVEX
- NOT AMENABLE TO GRADIENT TECHNIQUES.

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

(∃ OTHER L FCNS.)

- CONVEX
- DIFFERENTIABLE
- PROVIDES CONFIDENCE MEASURE.

Forward Stagewise Additive Modeling

1. Initialize $f_0 = 0$.
2. For $m = 1$ to M:
   (i) Find:
   $$(\beta_m, \underline{\gamma}_m) = \underset{\beta_m, \underline{\gamma}_m}{\operatorname{argmin}} \sum_{i=1}^{N} L\left[\tilde{y}_i, \hat{f}_{m-1}(\underline{x}_i) + \beta_m \phi(\underline{x}_i, \underline{\gamma}_m)\right]$$
   (ii) $\hat{f}_m(\underline{x}) = \hat{f}_{m-1}(\underline{x}) + \beta_m \phi_m(\underline{x}, \underline{\gamma}_m)$
3. Final classifier is:

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

Want to find:

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

$$\rightarrow \underset{f_0, \beta_m, \underline{\gamma}_m \; \forall m=1,2,\ldots,M.}{\operatorname{argmin}} \sum_{i=1}^{N} L_{exp}\left[\tilde{y}_i, f_0 + \sum_{m=1}^{M} \beta_m \cdot \phi(\underline{x}_i, \underline{\gamma}_m)\right]$$

Easier to min. each term, sequentially:

At m^{th} iteration:

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

Forward Stagewise Additive Modeling

1. INITIALIZE $f_0 = 0$.

2. FOR $m = 1$ TO M:

(i) FIND:

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

(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)

week learning: decision stup (one step decision tree)

ADABOOST

→ USE $L = L_{exp}$

FROM ABOVE:

2.(i) FIND (β_m, γ_m)

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

L_m

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

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

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

(7) $$L_m = \sum_{i=1}^{N} w_{i,m} \exp\left[-\tilde{y}_i \left(\beta_m' \phi(\underline{x}_i, \underline{\gamma}_m')\right)\right], \quad \beta_m' > 0.$$

- $\sum_{i=1}^{N}$: sum over all data pts.
- $w_{i,m}$: weight on i^{th} data pt., at m^{th} iteration.
- $\exp[\cdots]$: 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.

$\Rightarrow$

(*) $\phi(x_i, \gamma_m)$ IS CHOSEN TO MINIMIZE A SUM OF WEIGHTS OF MISCLASSIFIED DATA POINTS.

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

(***) $$err_m = \frac{\sum_{i=1}^{N} w_{i,m} \mathbb{I}[\tilde{y}_i \neq \phi(x_i, \gamma_m)]}{\sum_{i=1}^{N} w_{i,m}}$$

= SAMPLE-WEIGHTED ERROR RATE (AT m^{TH} ITERATION)

Algorithm

1. INITIALIZE $w_i = \frac{1}{N} \quad \forall i$.
2. FOR $m = 1$ TO M:
   (i) TRAIN CLASSIFIER $\phi_m(x)$ [1-NODE CART] ON WEIGHTED DATASET $\mathcal{D}_m$ (WEIGHTS w_i) PER (*).
   (ii) COMPUTE [FROM (***)]:

$$err_m = \frac{\sum_{i=1}^{N} w_i \mathbb{I}[\tilde{y}_i \neq \phi_m(x_i)]}{\sum_{i=1}^{N} w_i}$$

(iii) COMPUTE $\alpha_m = \log\left[\frac{1-err_m}{err_m}\right] = 2\beta_m$ in (**),

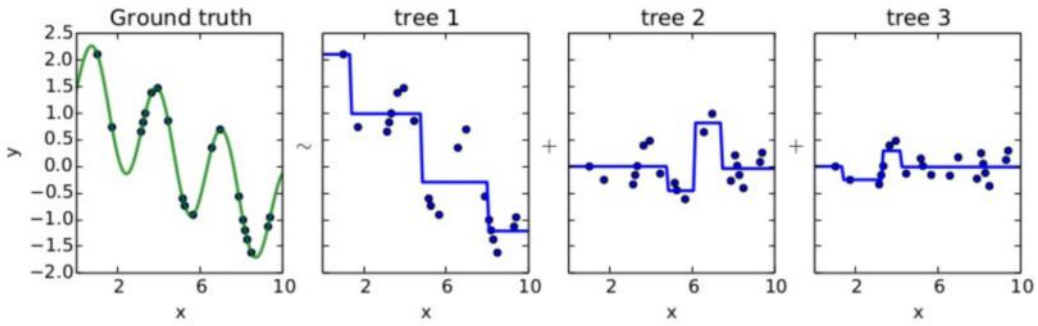
(iv) UPDATE $w_i \leftarrow w_i \exp\left[\alpha_m \mathbb{I}(y_i \neq \phi_m(x_i)\right]$ $\forall i$.

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

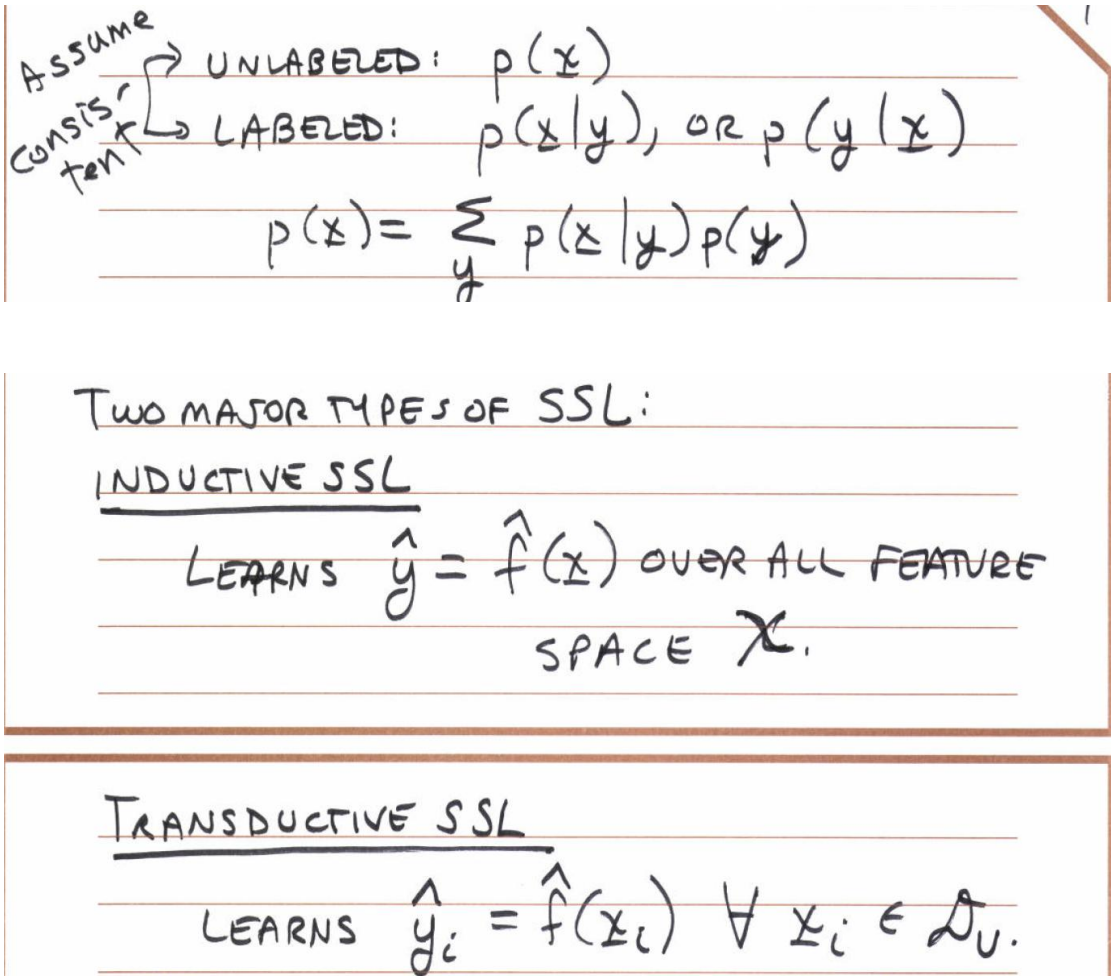
6.Semi-Supervised Learning

TRAINING SET:

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

AND

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



SSL Models/ Algorithm

Self- Training

Algorithm 2.4. Self-training. (WRAPPER)

Input: labeled data $\{(\mathbf{x}_i, y_i)\}_{i=1}^{l}$, *unlabeled data* $\{\mathbf{x}_j\}_{j=l+1}^{l+u}$.
1. Initially, let $L = \{(\mathbf{x}_i, y_i)\}_{i=1}^{l}$ *and* $U = \{\mathbf{x}_j\}_{j=l+1}^{l+u}$.
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 $\{(\mathbf{x}, \hat{f}(\mathbf{x})) | \mathbf{x} \in S\}$ *to* L.

↳ DATA POINTS WITH HIGHEST CONFIDENCE OF $\hat{f}(x)$ PREDICTION.

ASSUMPTION: DATA POINTS WITH HIGHEST CONFIDENCE OF $\hat{f}(x)$ PREDICTION TEND TO BE CORRECT.

SPECIFIC EX: **Algorithm 2.7. Propagating 1-Nearest-Neighbor.**

Propagating 1-Nearest-Neighbour

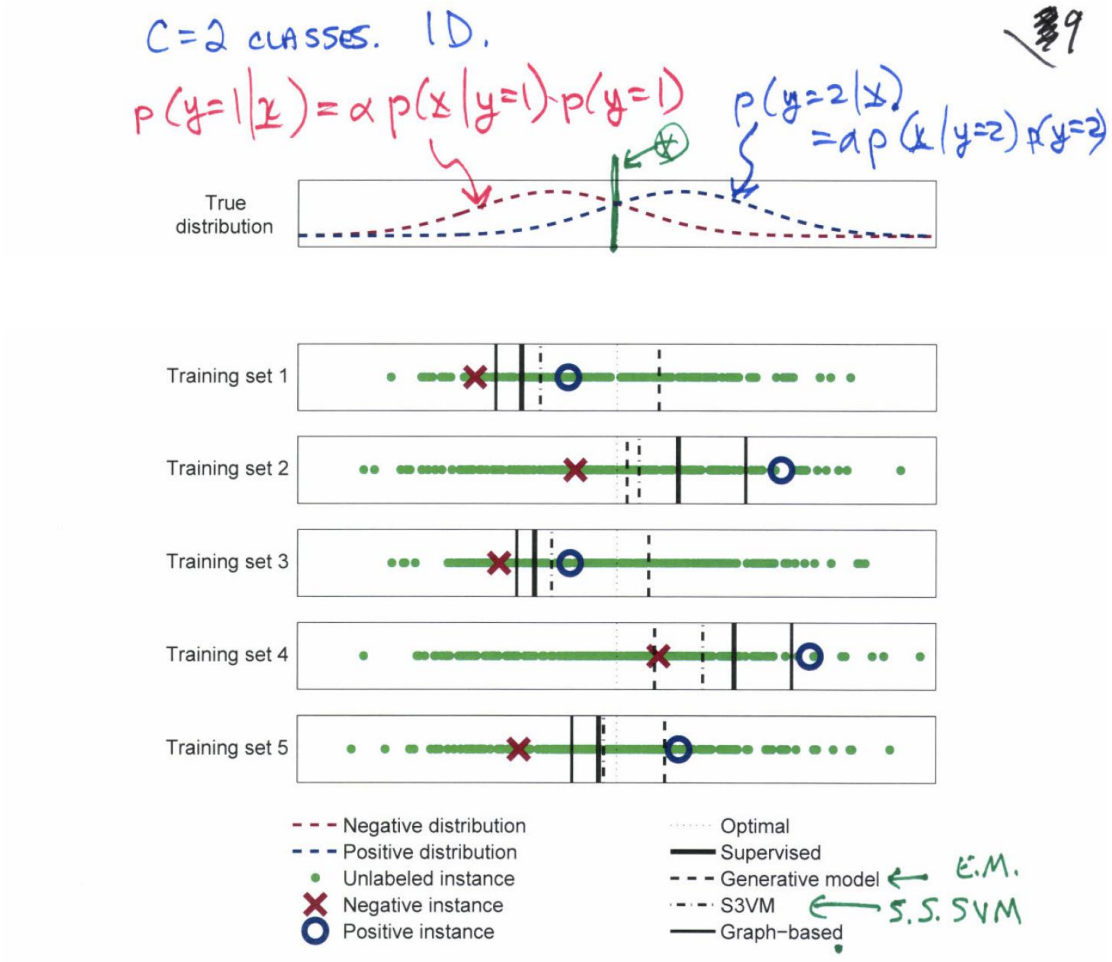
Ex: **Algorithm 2.7. Propagating 1-Nearest-Neighbor.** CORRECT.

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* $\mathbf{x} = \text{argmin}_{\mathbf{x} \in U} \min_{\mathbf{x}' \in L} d(\mathbf{x}, \mathbf{x}')$.
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 x_j THAT IS CLOSEST TO A LABELED DATA POINT x_i'.
$\Rightarrow x_j$ IS S.

Mixture Model and Parameter Classification



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

$$p(\underline{x} \mid y) = N(\underline{x} \mid \underline{\mu}_y, \underline{\underline{\Sigma}}_y)$$

(y: class index)

$\underline{\mu}_y$ AND $\underline{\underline{\Sigma}}_y$, $y = 1, 2, \cdots, C$ ARE UNKNOWN PARAMETERS $\underline{\theta}$.

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

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

($p(y)$: PRIOR)

2 ways: MLE and MAP

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

$$p(\underline{x}, y \mid \underline{\theta}) = p(\underline{x} \mid y, \underline{\theta})\, p(y \mid \underline{\theta})$$

$$= p(\underline{x} \mid y, \underline{\theta})\, \pi_y \quad (1)$$

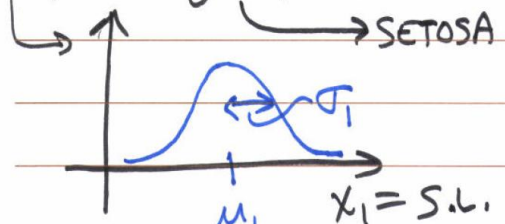
($p(\underline{x} \mid y, \underline{\theta})$: CLASS-CONDITIONAL DENSITY, CONDITIONED ON $\underline{\theta}$.)

($\pi_y = p(y)$ = PRIOR ON y).

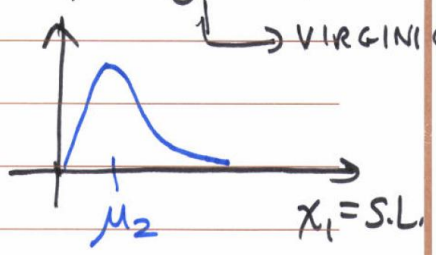
OUR MODEL:

LABELED DATA:

$p(\underline{x} \mid y=1, \underline{\theta})$ → SETOSA



$p(\underline{x} \mid y=2, \underline{\theta})$ → VIRGINICA



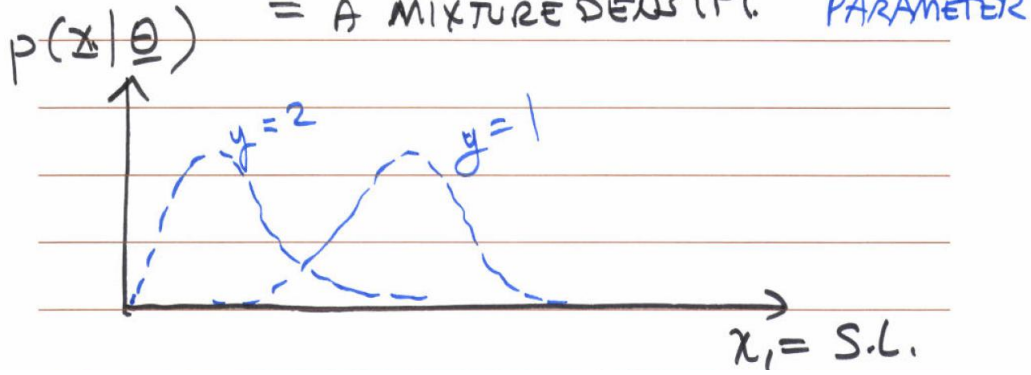
$$\underline{\theta} = \begin{bmatrix} \mu_1 \\ \sigma_1 \\ \mu_2 \\ \vdots \end{bmatrix}$$

UNLABELED DATA:

(2) $$p(\underline{x} \mid \underline{\theta}) = \sum_{y=1}^{C} p(\underline{x} \mid y, \underline{\theta}) \, \underbrace{p(y \mid \underline{\theta})}_{\pi_y}$$

COMPONENT DENS. — MIXING PARAMETER

= A MIXTURE DENSITY.



MLE

ONE WAY: MLE:

$$\hat{\underline{\theta}} = \underset{\underline{\theta}}{\operatorname{argmax}}\ p(\mathcal{D}|\underline{\theta}) = \underset{\underline{\theta}}{\operatorname{argmax}}\ \ln p(\mathcal{D}|\underline{\theta})$$

$$= \underset{\underline{\theta}}{\operatorname{argmax}} \sum_{i=1}^{l} \ln p(x_i, y_i|\underline{\theta})$$

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

EM Algorithm

EM ALGORITHM (GENERAL FORMULATION)

[FOLLOWS SSL TEXT; ALSO MURPHY 11.4]

COMMONLY USED FOR:

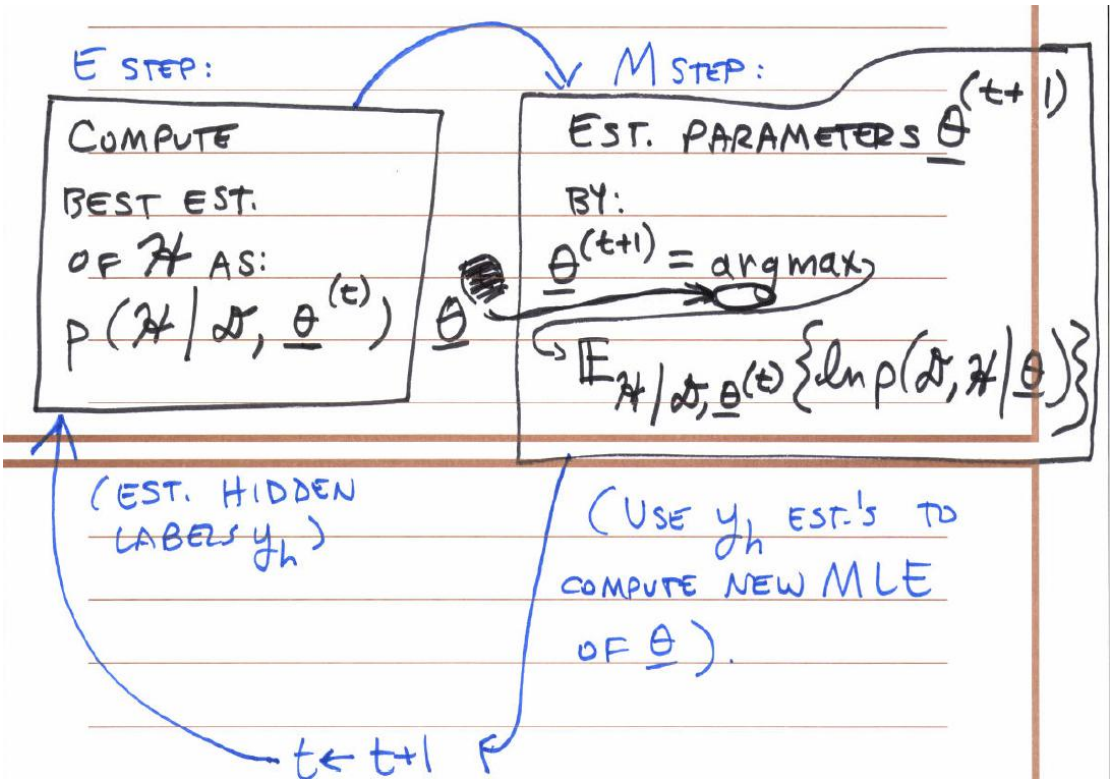
- WORKING WITH MISSING DATA.
- FINDING MLE IN DIFFICULT SITUATIONS.
- ESTIMATING QUANTITIES IN MIXTURE MODELS.

LET $\mathcal{D}$ BE ALL THE DATA:

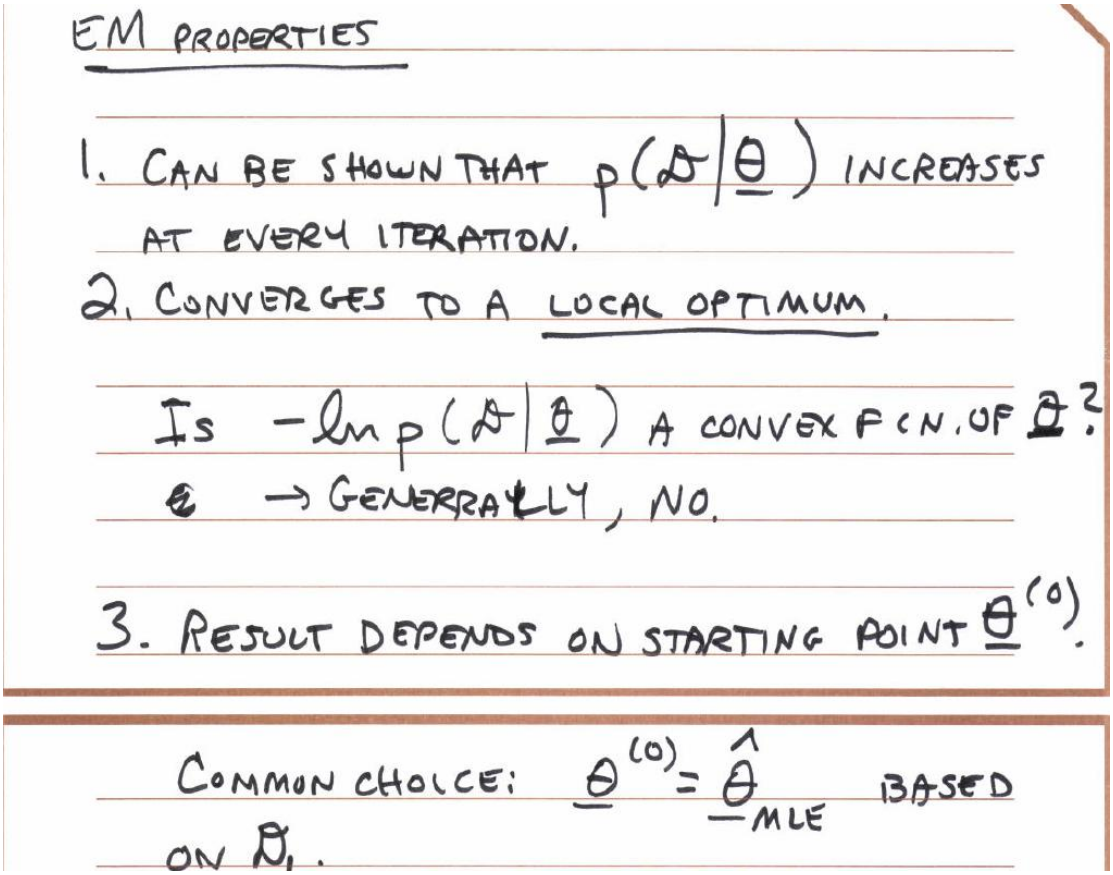
$$\{(x_i, y_i), i=1,\cdots,l;\ x_h,\ h=l+1,\cdots,l+u\}$$

LET $\mathcal{H}$ BE THE HIDDEN LABELS

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



halt when P(D/θ(t)) converge



Application of EM Algorithm

E step

FOR E STEP

LET i INDEX THE DATA PTS. IN $\mathcal{D}_L$;
h " " " " IN $\mathcal{D}_U$.

$$p(\mathcal{H} \mid \mathcal{D}, \underline{\theta}) = \prod_{h=l+1}^{l+u} p(y_h \mid \underline{X}_U, \underline{Y}_L, \underline{X}_L, \underline{\theta})$$

$$= \prod_h p(y_h \mid \underline{x}_h, \underline{\theta}) \quad (3)$$

(KNOW $\underline{\theta}$ $\Rightarrow$ DON'T NEED OTHER DATA.)

$$p(y_h = c \mid \underline{x}_h, \underline{\theta}) = \frac{p(\underline{x}_h \mid y_h = c, \underline{\theta})\, p(y_h \mid \underline{\theta})}{\sum_{y_h=1}^{C} p(y_h \mid \underline{\theta})\, p(\underline{x}_h \mid y_h, \underline{\theta})} \quad (4)$$

LET $\gamma_{hc} \triangleq p(y_h = c \mid \underline{x}_h, \underline{\theta})$

DATA PT. INDEX (UNLABELED) — CLASS ASSIGNMENT

γ_{hc} = RESPONSIBILITY OF $y_h = c$ LABEL FOR DATA PT. $\underline{x}_h$.
= "SOFT LABEL" FOR DATA PT. $\underline{x}_h$.
$p(y_h = c \mid \underline{x}_h, \underline{\theta})$.

M step

FOR M STEP

$$\max_{\underline{\theta}} E_{\mathcal{H}|\mathcal{D},\underline{\theta}^{(t)}} \{ \ln p(\mathcal{D}, \mathcal{H}|\underline{\theta}) \}$$

$$= \max_{\underline{\theta}} \left\{ \sum_{\mathcal{H}} p(\mathcal{H}|\mathcal{D},\underline{\theta}^{(t)}) \ln p(\mathcal{D},\mathcal{H}|\underline{\theta}) \right\}$$

$$p(\mathcal{D},\mathcal{H}|\underline{\theta}) = p(\mathcal{H}|\mathcal{D},\underline{\theta})\, p(\mathcal{D}|\underline{\theta})$$

$p(\mathcal{H}|\mathcal{D},\underline{\theta})$: GIVEN ABOVE. EQ. (3), (4).

$p(\mathcal{D}|\underline{\theta})$: LIKELIHOOD OF ALL KNOWN (GIVEN) DATA.

$$p(\mathcal{D}|\underline{\theta}) = \left[\prod_{i=1}^{l} p(x_i, y_i|\underline{\theta}) \right] \left[\prod_{h=l+1}^{l+u} p(x_h|\underline{\theta}) \right]$$

$$p(x_i, y_i|\underline{\theta}) = p(x_i|y_i,\underline{\theta})\, p(y_i|\underline{\theta})$$

$p(x_i|y_i,\underline{\theta})$: MODEL FOR LABELED DATA.

$p(y_i|\underline{\theta})$: π_{y_i}

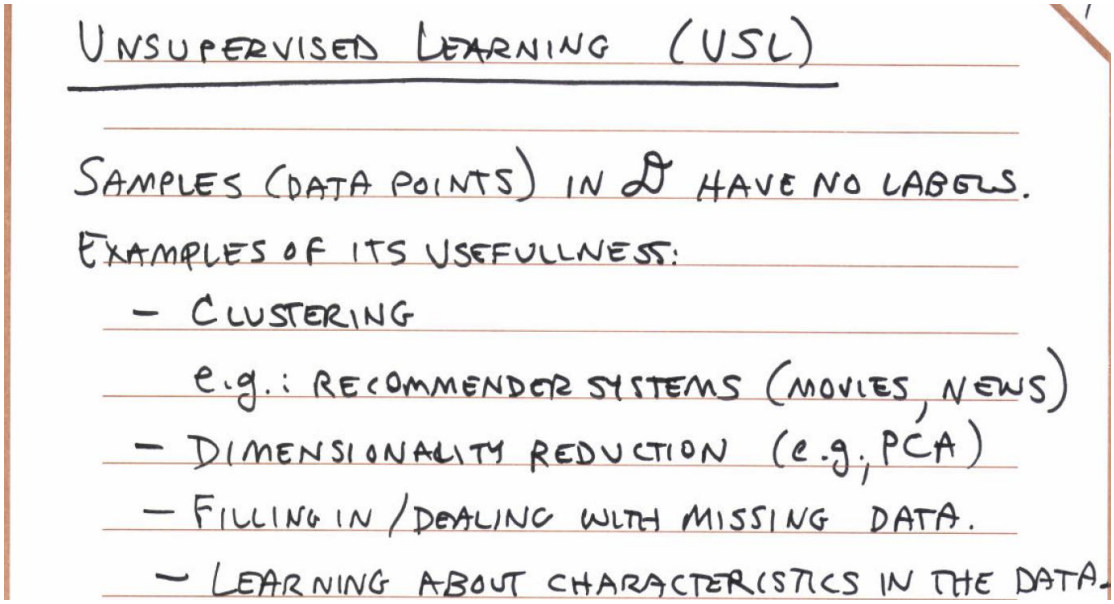
MIXTURE DENSITY (EQ. (2)):

$$p(x_h|\underline{\theta}) = \sum_{y_k=1}^{C} p(x_h|y_k,\underline{\theta})\, \pi_{y_k} \quad (*)$$

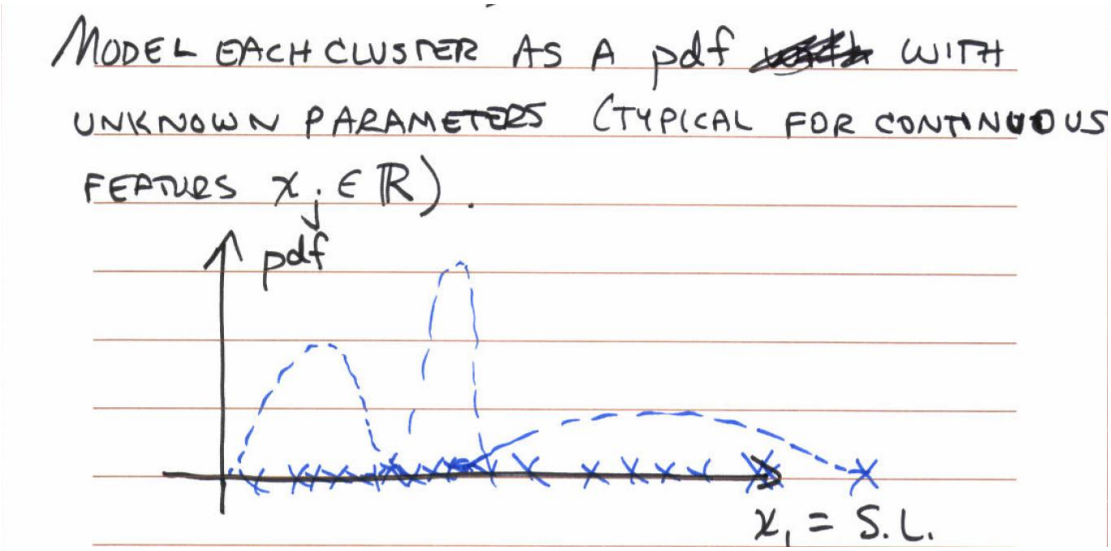
conclusion of EM:

works well when model is correct, otherwise may not good

7. Unsupervised Learning



Density Estimate



THUS:

$$p(\underline{x}|\underline{\theta}) = \sum_{k=1}^{K} p(\underline{x}|z=k, \underline{\theta})\, p(z=k|\underline{\theta})$$

$p(z=k|\underline{\theta}) = \pi_k$

z = CLUSTER INDEX

$\underline{\theta}_k$ = PARAMETERS FOR CLUSTER k.

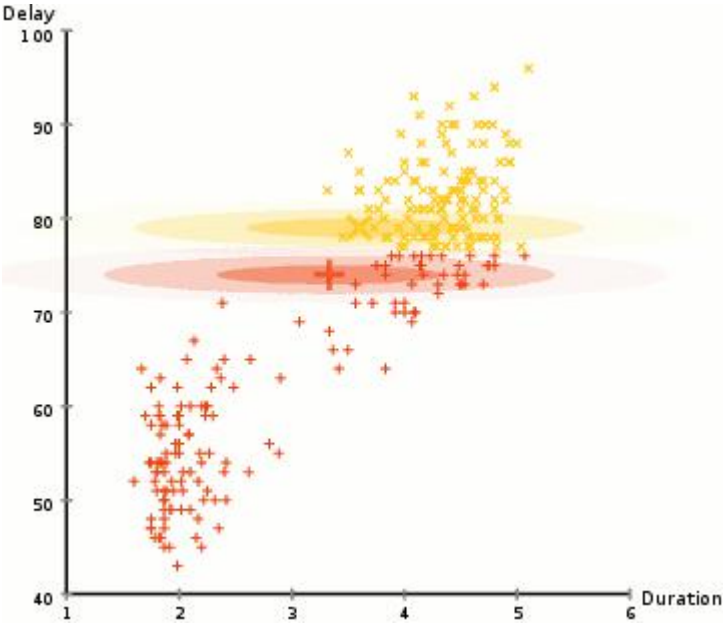
= A MIXTURE MODEL

$$p(\underline{x}|\underline{\theta}) = \sum_{k=1}^{K} \pi_k\, p(\underline{x}|z=k, \underline{\theta}_k)$$

$p(\underline{x}|z=k, \underline{\theta}_k)$ = OUR MODEL FOR CLUSTER k.

AND $\sum_{k=1}^{K} \pi_k = 1$.

EM Algorithm



EM FOR CLUSTERING USING MIXTURE MODELS

SAME BASIC ALGORITHM AS EM FOR SSL.

LET $\mathcal{D} = \{x_i\}_{i=1}^N$

$\mathcal{H} = \{z_i\}_{i=1}^N$, z_i IS CLUSTER LABEL FOR DATA PT. x_i

$z_i \in \{1, 2, \cdots, K\}$.

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:

$$\underline{\theta}^{(t+1)} = \underset{\underline{\theta}}{\operatorname{argmax}} \, \mathbb{E}_{\mathcal{H}|\mathcal{D},\underline{\theta}^{(t)}} \{\ln p(\mathcal{D}, \mathcal{H} | \underline{\theta})\}$$

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

$$p(\mathcal{H} | \mathcal{D}, \underline{\theta}) = \prod_{i=1}^{N} p(z_i | x_i, \underline{\theta})$$

$$p(z_i | x_i, \underline{\theta}) = p(x_i | z_i, \underline{\theta}) \, \pi_{z_i}$$

our model

$$p(z=k | \underline{x}, \underline{\theta}_k) = \frac{p(\underline{x} | z=k, \underline{\theta}_k) \, p(z=k | \underline{\theta}_k)}{\sum_{k'=1}^{K} p(\underline{x} | z=k', \underline{\theta}_{k'}) \, p(z=k' | \underline{\theta}_{k'})}$$

$\rightarrow$ SOFT LABEL $\gamma_{ik}^{(t)} = p(z_i = k | x_i, \underline{\theta}_k^{(t)})$

M Step

EQNS. FOR M STEP

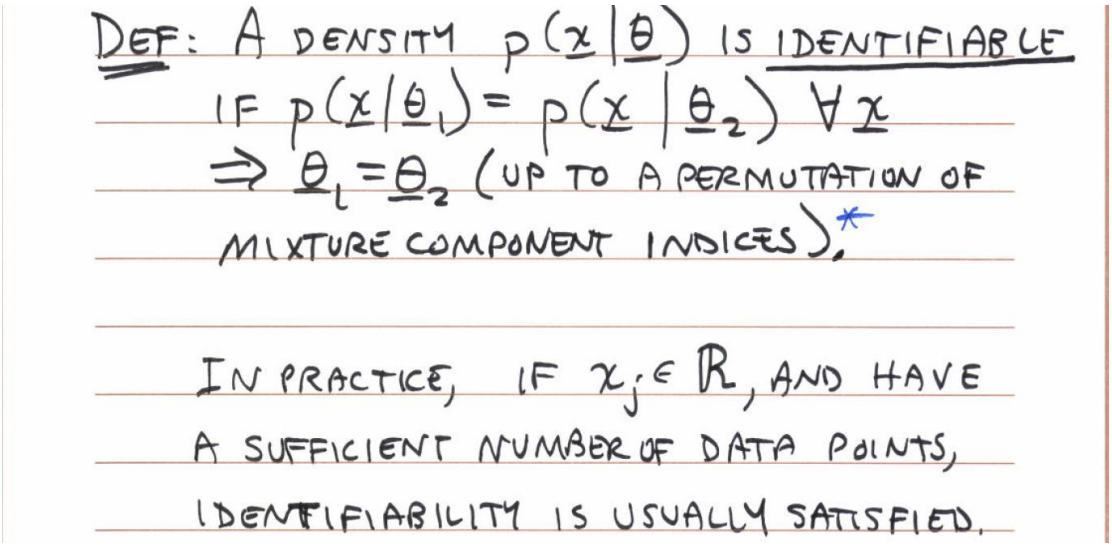
$$p(\mathcal{X}, \mathcal{H} \mid \underline{\theta}) = \prod_{i=1}^{N} p(x_i, z_i \mid \underline{\theta})$$

$$= \prod_{i=1}^{N} p(x_i \mid z_i, \underline{\theta})\, p(z_i \mid \underline{\theta})$$

($p(x_i \mid z_i, \underline{\theta})$: our model; $p(z_i \mid \underline{\theta})$: π_k)

$$\rightarrow \underline{\theta}^{(t+1)} = \underset{\underline{\theta}}{\operatorname{argmax}} \left\{ \sum_{\mathcal{H}} p(\mathcal{H} \mid \mathcal{X}, \underline{\theta}^{(t)}) \cdot \ln p(\mathcal{X}, \mathcal{H} \mid \underline{\theta}) \right\}$$

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 (N RANDOM) CHOICES OF $\underline{\theta}^{(0)}$. COMPARE RESULTS USING $p(\mathcal{X} \mid \underline{\theta})$.
3. DOES THERE EXIST A UNIQUE BEST SOLUTION? IF MIXTURE $p(\underline{x} \mid \underline{\theta})$ IS IDENTIFIABLE, THEN IN LIMIT $N \rightarrow \infty$, YES.



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

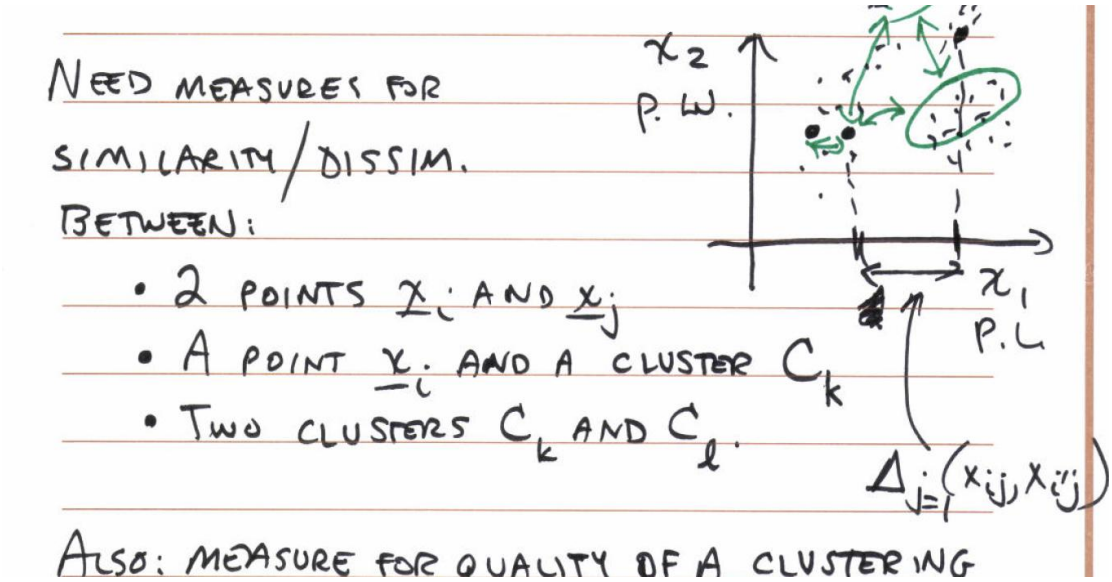
Connectivity models (Hiarerchy Clustering): As the name suggests, these models are based on the notion that the data points closer in data space exhibit more similarity to each other than the data points lying farther away. These models can follow two approaches. In the first approach, they start with 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.

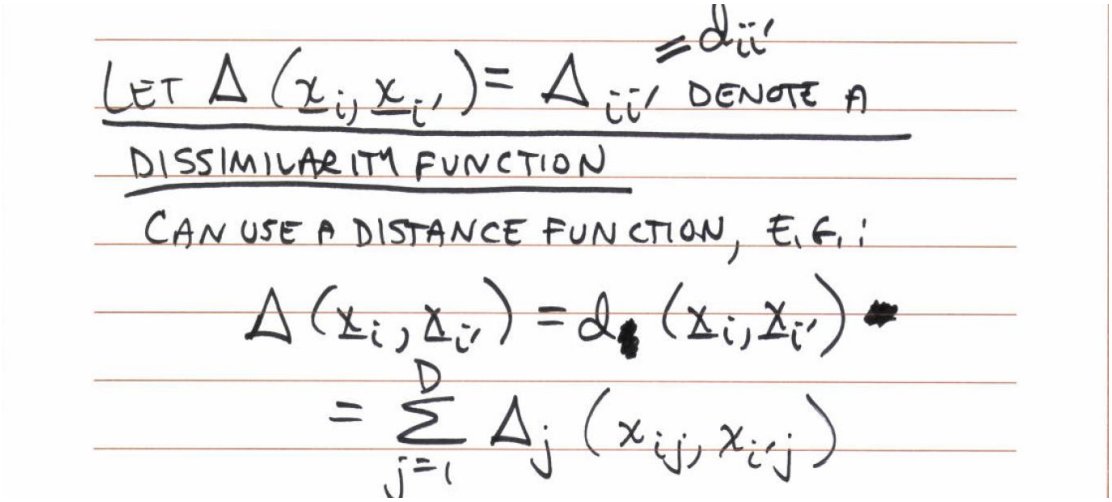
Density Models (DBSCAN): These models search the data space for areas of varied density of data points in the data space. It isolates various different density regions and assign the data points within these regions in the same cluster. Popular examples of density models are DBSCAN and OPTICS.

Similarity/ Dissimilarity Measure



Euclide Distance | Manhatton Distance | Hamming Distance

2 | 1 | 0 (# of different features)



$\Delta_j(x_{ij}, x_{i'j})$ CAN BE:

$(x_{ij} - x_{i'j})^2$ (EUCL. DIST.^{2})

$|x_{ij} - x_{i'j}|$ (ℓ_1 NORM DIST. OR CITY-BLOCK DIST. OR MANHATTAN DIST.)

FOR NOMINAL FEATURES (SYMBOLS, CATEGORICAL, OR LABELS):

$\Delta(x_{ij}, x_{i'j})$ = # OF FEATURES THAT ARE DIFFERENT

$$= \sum_{j=1}^{p} \mathbb{I}(x_{ij} \neq x_{i'j})$$

= HAMMING DISTANCE.

CAN LET $\Delta(x_i, x_{i'}) =$ THIS OR OTHER $d_{ii'}$, s.t. $d_{ii'} \geq 0 \ \forall i, i'$, AND $d_{ii} = 0 \ \forall i$

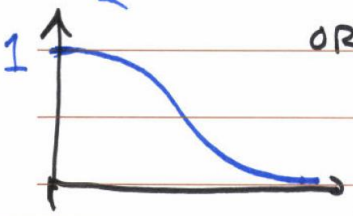
OR $f(d_{ii'})$, $f =$ ANY MONOTONICALLY INCREASING FUNCTION.

LET $s(x_i, x_{i'}) = s_{ii'}$ DENOTE A SIMILARITY FCN.

CAN CHOOSE $s_{ii'} = g[d_{ii'}]$, g = ANY MONOTONICALLY DECREASING FUNCTION.

E.G.: $s(x_i, x_{i'}) = \left(\max_{l,l'} d_{ll'}\right) - d_{ii'}$

OR $= \exp\left\{\frac{-d_{ii'}^2}{\sigma^2}\right\}$



similarity : when near 1 (large)--> two data point are almost same

FOR BINARY FEATURES

(e.g., CONTAINS AN ATTRIBUTE OR NOT):

$$s(x_i, x_{i'}) = \frac{x_i^T x_{i'}}{x_i^T x_i + x_{i'}^T x_{i'} - x_i^T x_{i'}}$$

= % OF ATTRIBUTES THAT ARE SHARED.

CORRELATION COEFFICIENT, E.G. (PEARSON);

LET j BE INDEX OVER SPATIAL LOCATION OR TIME

$$r_{ii'} = \frac{\sum_{j=1}^{D} (x_{ij} - \bar{x}_i)(x_{i'j} - \bar{x}_{i'})}{\left[\sum_j (x_{ij} - \bar{x}_i)^2 \sum_j (x_{ij} - \bar{x}_{i'})^2\right]^{1/2}}$$

IN WHICH:

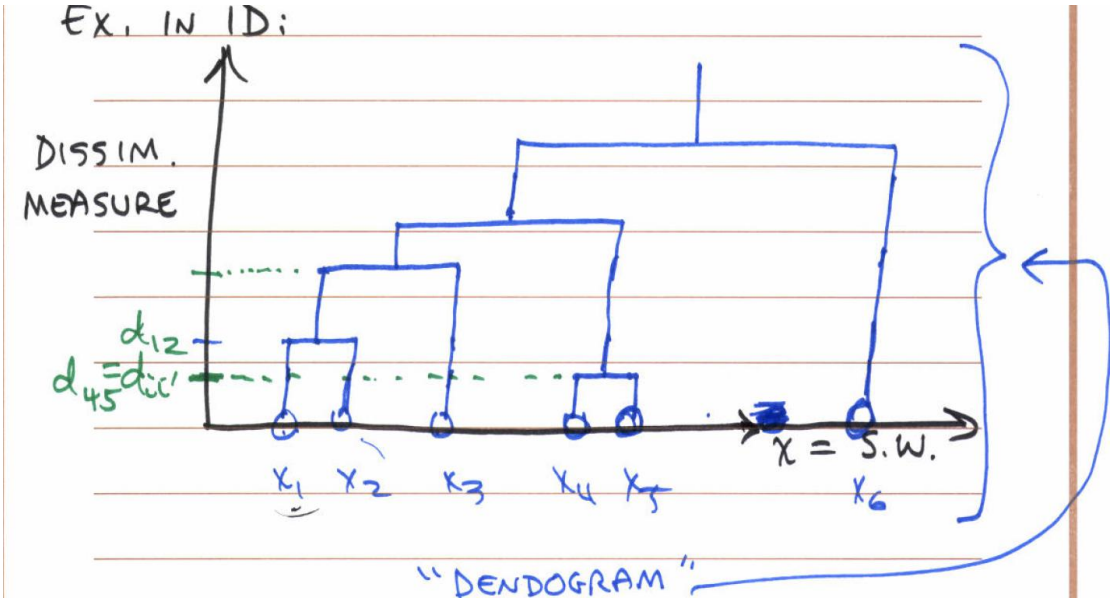
$$\bar{x}_i = \frac{1}{D}\sum_{j=1}^{D} x_{ij}$$. NOTE: $-1 \le r_{ii'} \le 1$ ALWAYS.

CAN LET $S_{ii'} = r_{ii'}$. FOR SIMILARITY.

OR: $d_{ii'} = \frac{1 - r_{ii'}}{2}$

Hierarchical Clustering

find the nearest one and combine together as a new node. Then expand



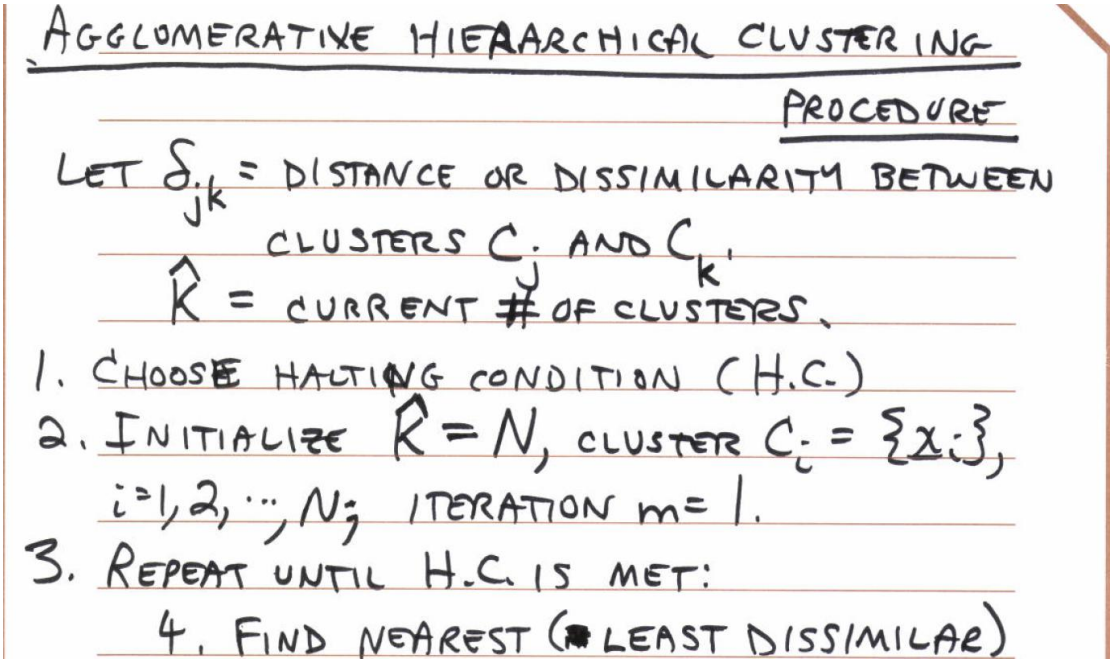
TWO BASIC APPROACHES:

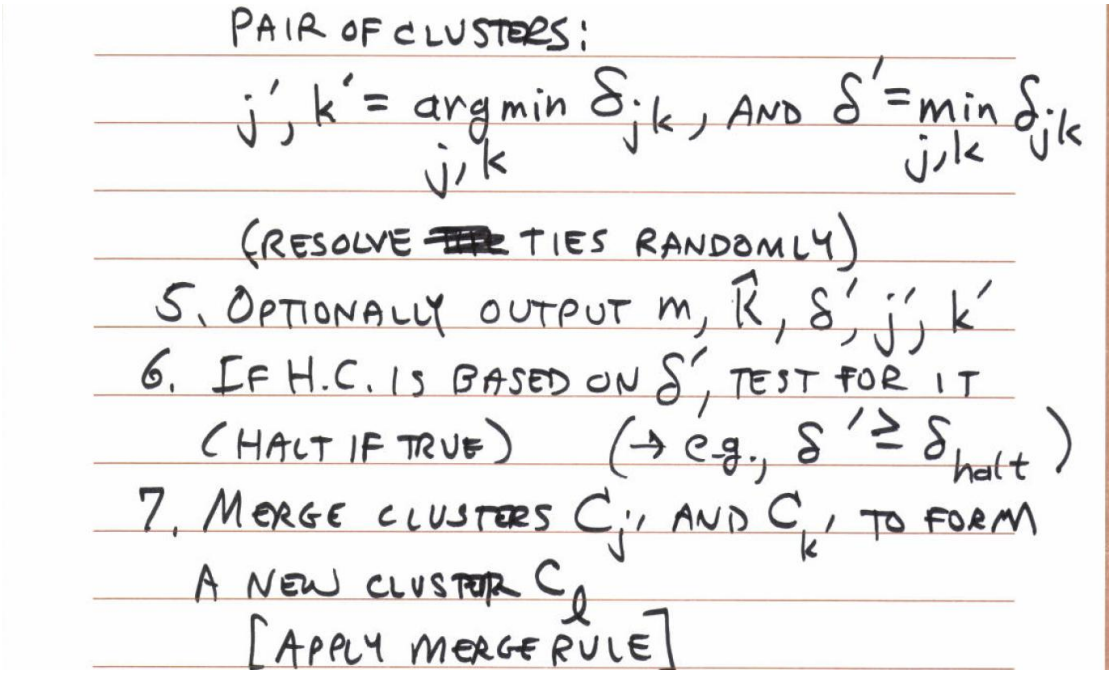
- AGGLOMERATIVE (BOTTOM UP).
- DIVISIVE (TOP DOWN)

→ 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





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, $\ell = 1, \cdots, \hat{K}_{FINAL}$; $\hat{K}_{FINAL}$; DISSIMILARITY $\delta(m)$.

IF $\hat{K}_{FINAL} = 1$, THE RESULTING HIERARCHY IS A DENDOGRAM.

USEFUL DISTANCE OR DISSIMILARITY MEASURES: (BETWEEN CLUSTERS C_k, C_ℓ):

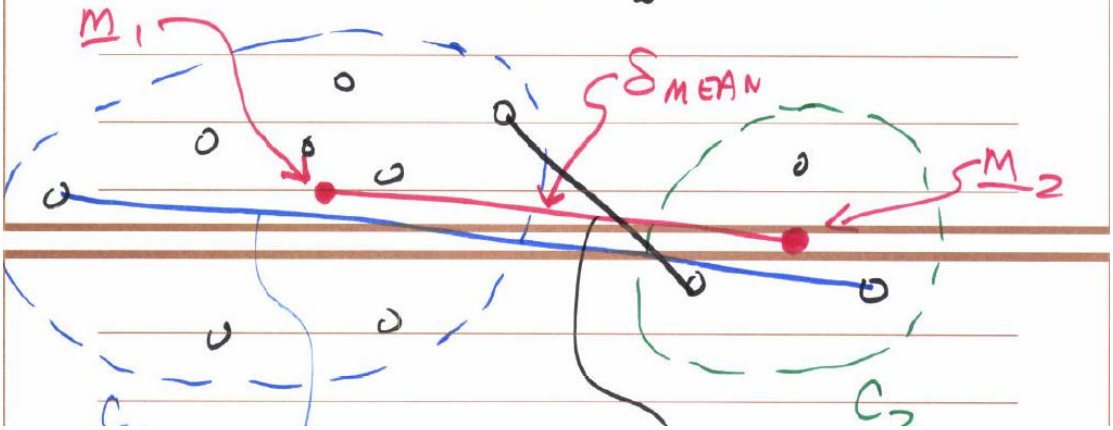
$$\delta_{MEAN}(C_k, C_\ell) \triangleq \| \underline{m}_k - \underline{m}_\ell \|_2$$

$$\delta_{AVG}(C_k, C_\ell) \triangleq \frac{1}{N_k N_\ell} \sum_{\underline{x} \in C_k} \sum_{\underline{x}' \in C_\ell} \| \underline{x} - \underline{x}' \|_2$$

(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 $\hat{K}_{FINAL} = 1 \Rightarrow$ SPANNING TREE

USE $\delta_{min} \Rightarrow$ MINIMAL SPANNING TREE

4

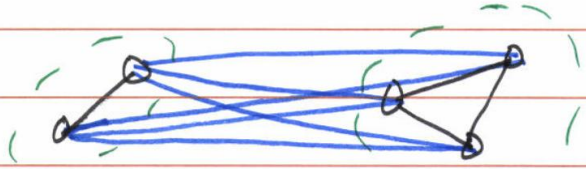
(A SPANNING TREE THAT HAS MIN. TOTAL LENGTH OF ALL EDGES).

Farthest Neighbor

USE δ_{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

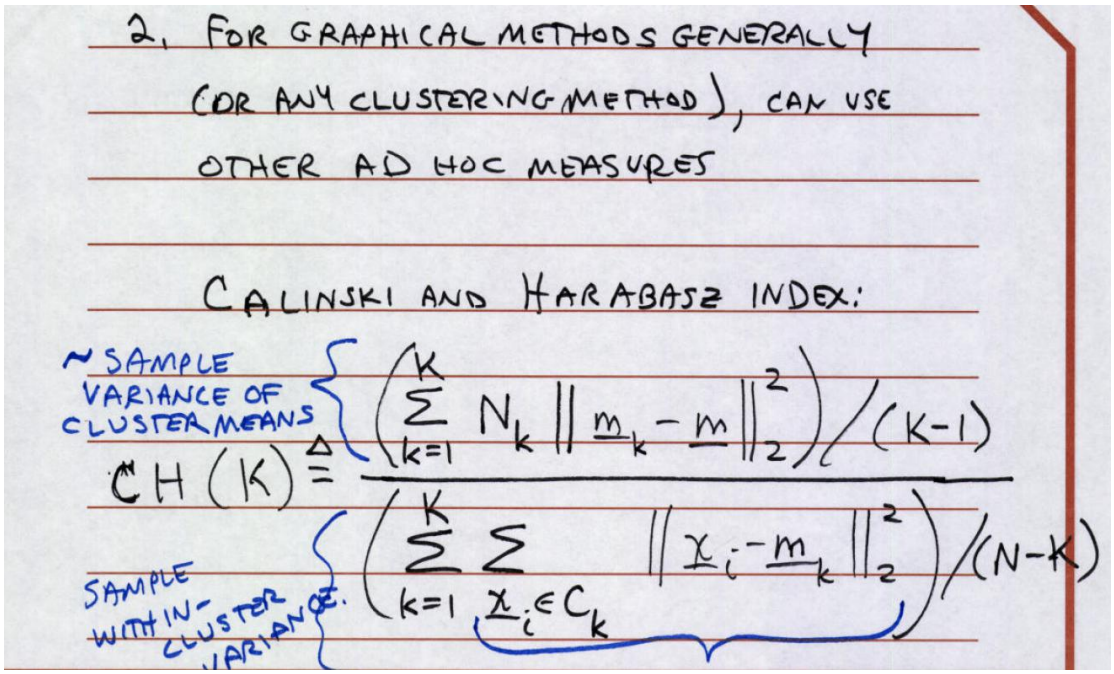
1. FOR AGGLOMERATIVE HIERARCHICAL CLUSTERING, CAN USE THE STEPWISE CRITERION:

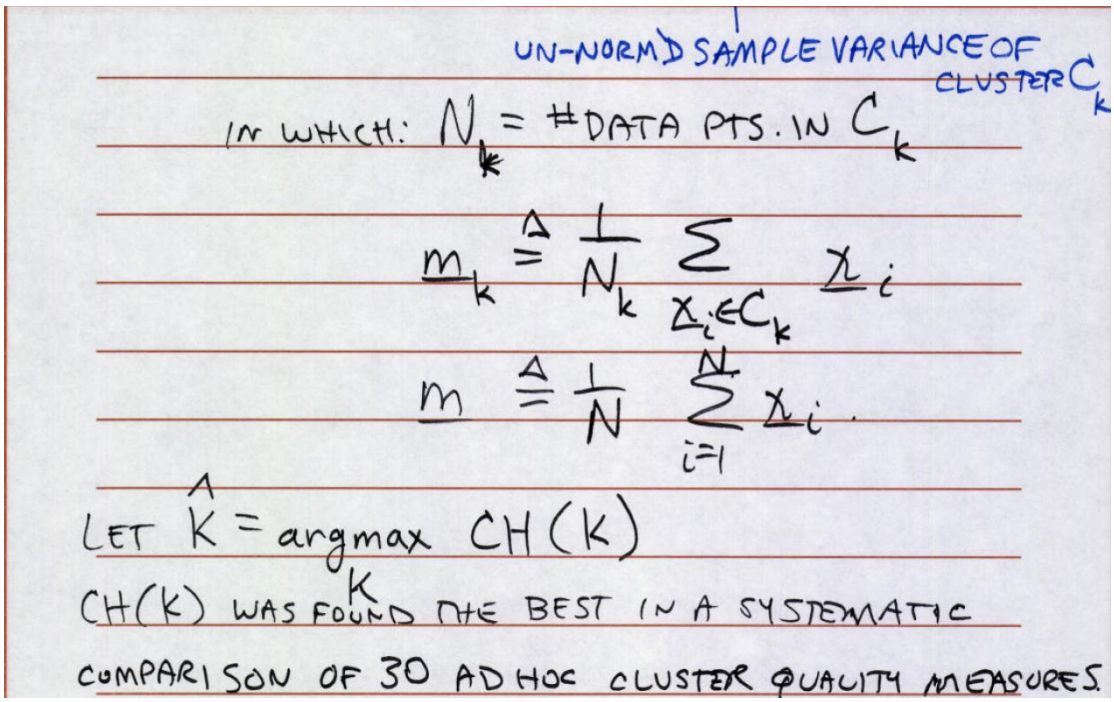
Ex: $\delta'(m) = \min_{j,k} \delta_{jk}$

IN WHICH FOR NN/SINGLE LINKAGE:

$\delta_{jk} = \delta_{min}(C_j, C_k)$

cluster quality measurment

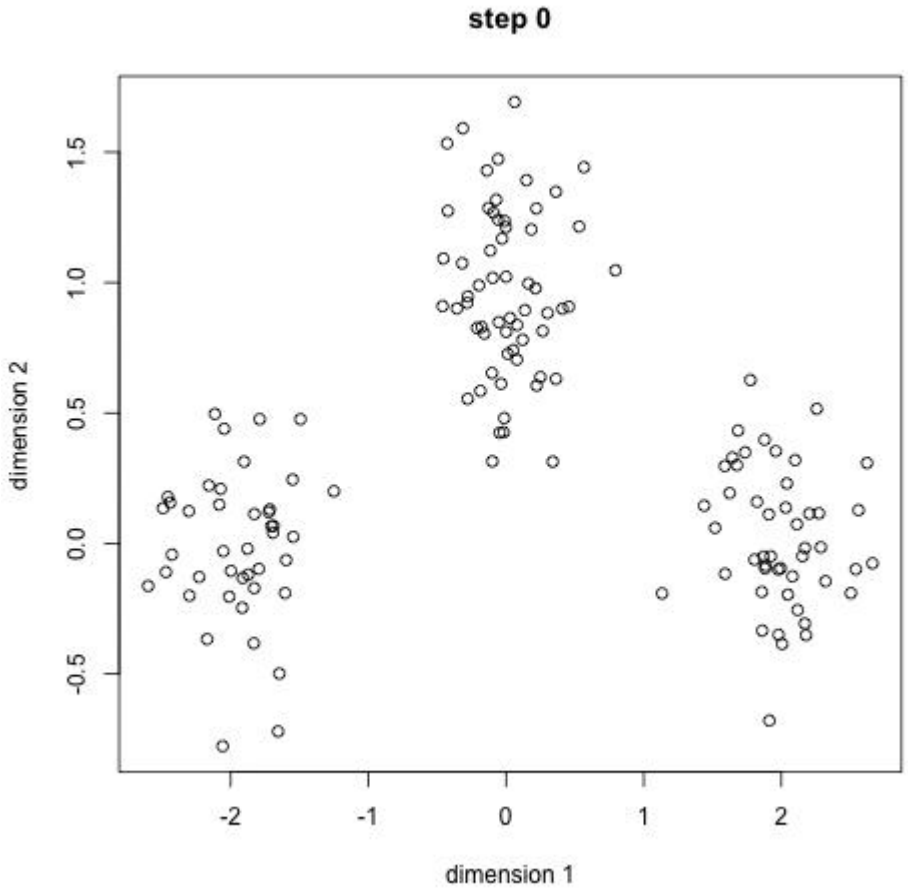




Divisive

Top down:

K Means Clustering



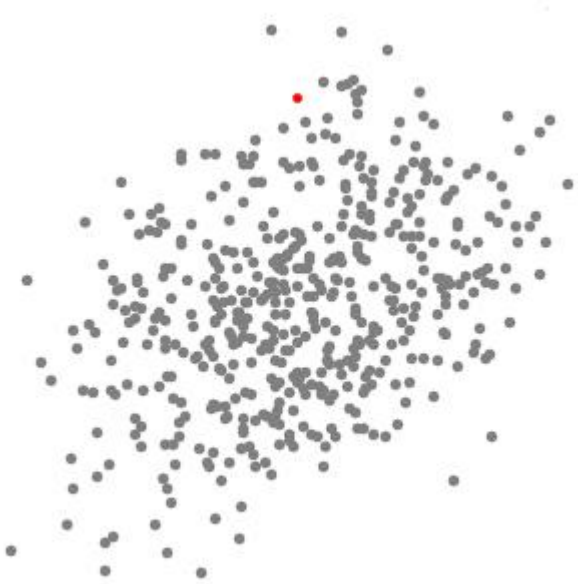
To begin, we first select a number of classes/groups to use and randomly initialize their respective center points. To figure out the number of classes to use, it's good to take a quick look at the data and try to identify any distinct groupings. The center points are vectors of the same length as each 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.

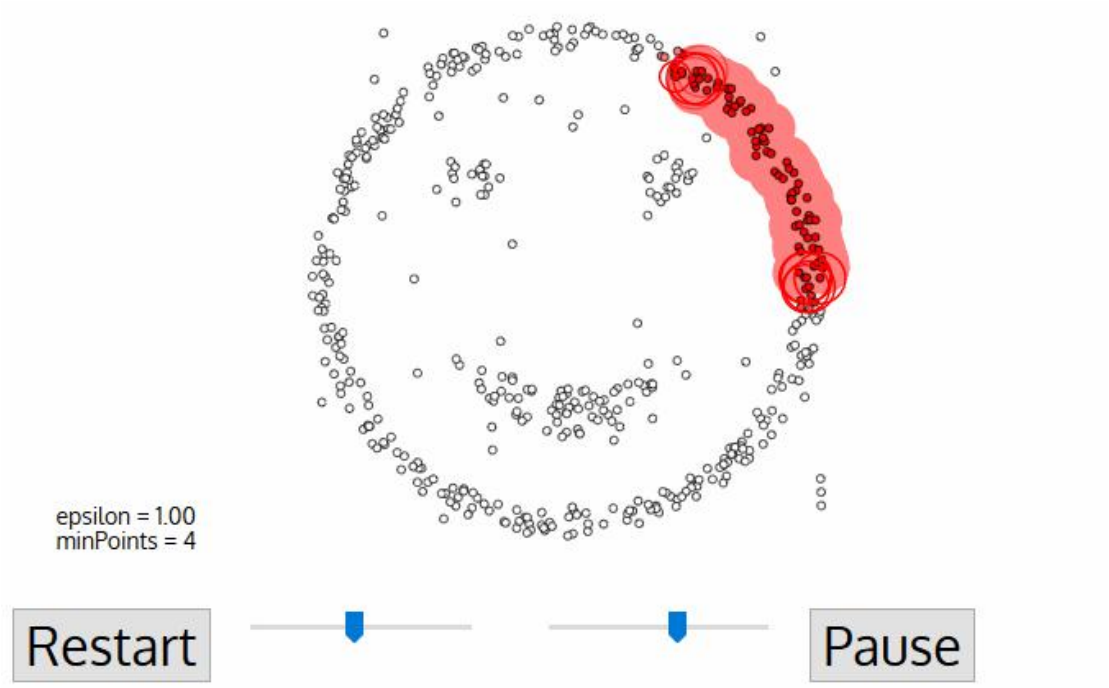
Based on these classified points, we recompute the group center by taking the mean of all the vectors in the group.

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)



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 ε (All points which are within the ε distance are neighborhood points).

If there are a sufficient number of points (according to 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 ε distance neighborhood also become part of the same cluster. This procedure of making all points in the ε 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 ε 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 well 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 pe-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 ε 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 ε becomes challenging to estimate.

Clustering Measurment

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 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 the 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

$$s(i) = 0, \text{ if } |C_i| = 1$$