

CSE 417T

Introduction to Machine Learning

Review of Exam 1

Instructor: Chien-Ju (CJ) Ho

- Homework 3: Due **Mar 19 (Friday)**.
 - Keep track of your late days
 - Utilize the office hours early. Don't wait till the last day
- Exam 1: **Mar 23 (Tuesday)**
 - Duration: 75+5 Minutes
 - Content: LFD Chapters 1 to 5
 - Time: during lecture time
 - If you have asked for exceptions, you should have heard from me already.
 - Format: Gradescope online exam + Zoom (with camera on)
 - Information access during exam:
 - Allowed: Textbook, slides, hardcopy materials (e.g., your own notes)
 - **Not allowed: search for information online during exam, talk to any other persons**
 - Other notes
 - **Follow Piazza announcements**
 - Practice questions are now on Gradescope (get familiar with interface)
 - **This Thursday lecture will be a review lecture**

More Exam Information

- Join Zoom (on mycanvas) and turn on camera before starting the exam
 - Won't be recorded
 - Students taking the exam at a different time need to do so as well
 - The Zoom + Camera policy is “softly” enforced
 - I might ask you for reasons for not following
 - No penalty as long as there are legitimate reasons
- We won't be able to answer questions during the exam
 - I don't answer individual questions even during in-person exams for fairness concerns
 - Will take into account of potential mistakes on our end
 - E.g., if there are multiple feasible answers for the multi-choice questions
 - Will give points if you choose one of the feasible ones

More Exam Information

- Please do not discuss the exam on Piazza within 24 hours of the exam
 - Some students take the exam at a different time
- Get familiar with how to submit math-heavy answers on Gradescope
 - prepare blank papers, having a smooth process of taking photos and upload file
- I'll post a meta-post about Exam 1 on Piazza by Sunday night
 - Make sure you read the post and know everything there
 - We should have already talked about important things multiple times

Plans for Today

- A summary of the content of Exam 1.
- Discussion of the practice questions.
- Discussion of any other questions you might have.

Review for Exam 1

Brief overview on the content.

Not comprehensive and not covering everything that could appear in the exam.

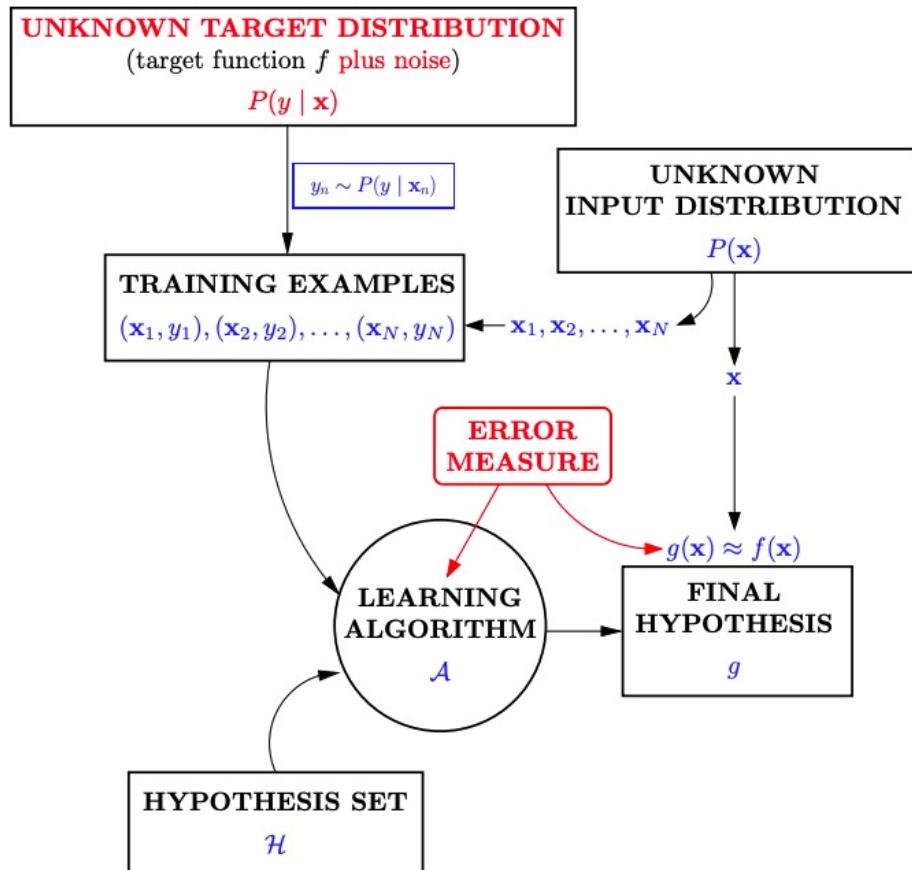
Please make sure you still study for LFD Chapter 1-5.

Let me know if you find mistakes in lecture notes.

Whenever you have doubts on the lecture notes, please use the textbook for the confirmation.

- Chap 1: Setting up the learning problem
 - Problem setup
 - probability assumptions/inferences
 - error and noise
- Chap 2: Theory of generalization (training v.s. testing)
 - Hoeffding's inequality
 - VC theory
 - Bias-variance decomposition
- Chap 3: Linear models
 - Linear classification/regression,
 - logistic regression, gradient descent,
 - nonlinear transformations
- Chap 4: Overfitting
 - Overfitting,
 - Regularization and validation
- Chap 5: Three learning principles
 - Occam's razor, sampling bias, data snooping

Setup of the Learning Problem



- Key assumption:
 - Training/testing data from the same distribution
- Define (point-wise) error measure:
 - Binary error $e(h(\vec{x}), y) = \mathbb{I}[h(\vec{x}) \neq y]$
 - Squared error $e(h(\vec{x}), y) = (h(\vec{x}) - y)^2$
 - Cost matrix

		f	
		+1	-1
h	+1	0	1
	-1	10	0

Supermarket

		f	
		+1	-1
h	+1	0	1000
	-1	1	0

CIA

Hoeffding's Inequality

- Single hypothesis bound

- Fix a hypothesis h

- $E_{in}(h) = \frac{1}{N} \sum_{n=1}^N e(h(\vec{x}_n), y_n)$ = In-sample error of h

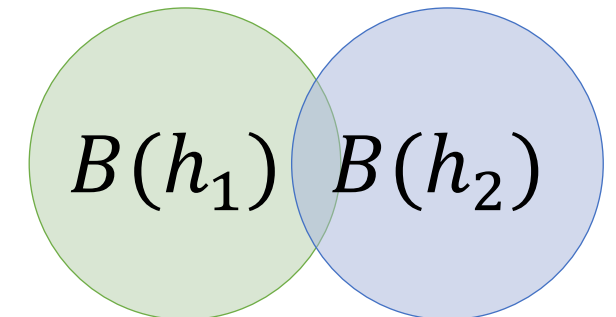
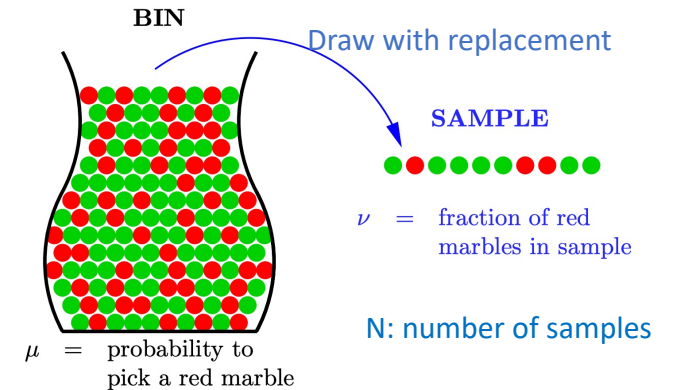
- $E_{out}(h) = \mathbb{E}_{\vec{x}}[e(h(\vec{x}), y)]$ = Out-of-sample error of h

- Hoeffding's inequality: $\Pr[|E_{out}(h) - E_{in}(h)| > \epsilon] \leq 2e^{-2\epsilon^2 N}$

- Multi-Hypothesis bound

- Learn a g from a finite hypothesis set $H = \{h_1, \dots, h_M\}$

- $\Pr[|E_{out}(g) - E_{in}(g)| > \epsilon] \leq 2Me^{-2\epsilon^2 N}$



Dealing with Infinite Hypothesis Set: $M \rightarrow \infty$

- Instead of # hypothesis, counting “effective” # hypothesis

- Dichotomy

- Informally, consider it as “data-dependent” hypothesis
 - Characterized by both H and N data points $(\vec{x}_1, \dots, \vec{x}_N)$

$$H(\vec{x}_1, \dots, \vec{x}_N) = \{h(\vec{x}_1), \dots, h(\vec{x}_N) | h \in H\}$$

- The set of possible prediction combinations $h \in H$ can induce on $\vec{x}_1, \dots, \vec{x}_N$

- Growth function

- Largest number of dichotomies H can induce across all possible data sets of size N

$$m_H(N) = \max_{(\vec{x}_1, \dots, \vec{x}_N)} |H(\vec{x}_1, \dots, \vec{x}_N)|$$

Why Growth Function?

- Finite-hypothesis Bound
With prob at least $1 - \delta$,

$$E_{out}(g) \leq E_{in}(g) + \sqrt{\frac{1}{2N} \ln \frac{2M}{\delta}}$$

- VC Generalization Bound (VC Inequality, 1971)
With prob at least $1 - \delta$

$$E_{out}(g) \leq E_{in}(g) + \sqrt{\frac{8}{N} \ln \frac{4m_H(2N)}{\delta}}$$

If we know the growth function $m_H(N)$ of H , we can obtain the learning guarantee for algorithms operating on H .

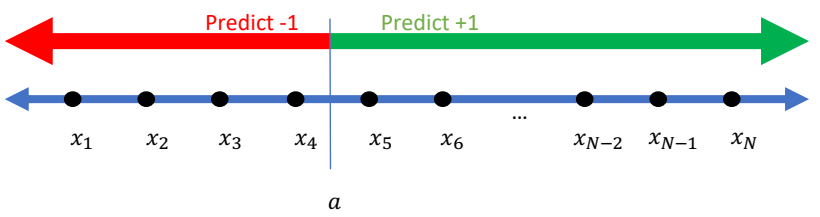
Bounding Growth Functions

- More definitions....
 - Shatter
 - H **shatters** $(\vec{x}_1, \dots, \vec{x}_N)$ if $|H(\vec{x}_1, \dots, \vec{x}_N)| = 2^N$
 - H can induce all label combinations for $(\vec{x}_1, \dots, \vec{x}_N)$
 - Break point
 - k is a **break point** for H if no data set of size k can be shattered by H
 - k is a break point for $H \leftrightarrow m_H(k) < 2^k$
- VC Dimension: $d_{vc}(H)$ or d_{vc}
 - The VC dimension of H is the largest N such that $m_H(N) = 2^N$
 - Equivalently, if k^* is the smallest break point for H , $d_{vc}(H) = k^* - 1$

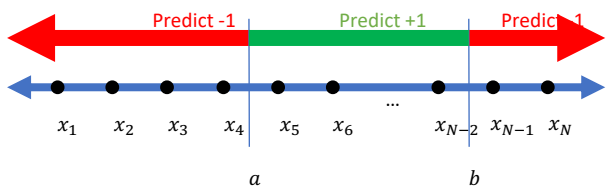
Examples

	$m_H(N)$					Break Points	VC Dimension
	N=1	N=2	N=3	N=4	N=5		
Positive Rays	2	3	4	5	6	$k = 2, 3, 4, \dots$	1
Positive Intervals	2	4	7	11	16	$k = 3, 4, 5, \dots$	2
Convex Sets	2	4	8	16	32	None	∞
2D Perceptron	2	4	8	14	?	$k = 4, 5, 6, \dots$	3

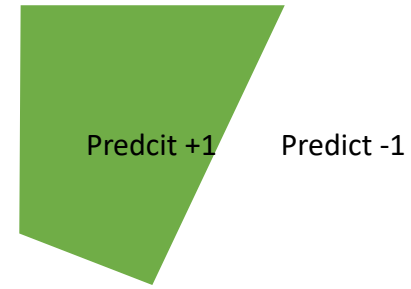
Positive Rays



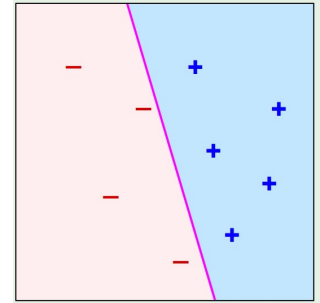
Positive Intervals



Convex Sets



2D Perceptron

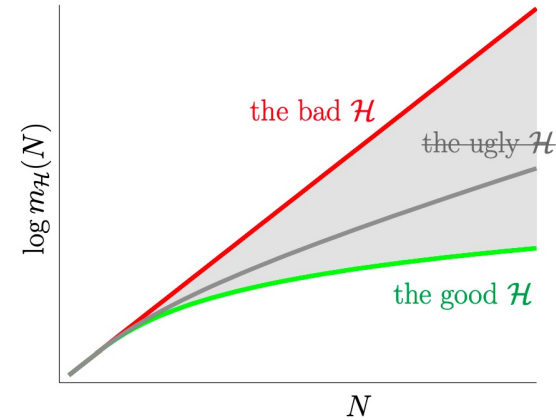


Bounding Growth Functions using Break Points

- Theorem statement:

- If there is no break point for H , then $m_H(N) = 2^N$ for all N .
- If k is a break point for H , i.e., if $m_H(k) < 2^k$ for some value k ,

$$m_H(N) \leq \sum_{i=0}^{k-1} \binom{N}{i}$$



- Rephrase the 2nd point of the above theorem

- If k is a break point for H , the following statements are true
 - $m_H(N) \leq N^{k-1} + 1$ [Can be proven using induction from above. See LFD Problem 2.5]
 - $m_H(N) = O(N^{k-1})$
 - $m_H(N)$ is polynomial in N

- If d_{vc} is the VC dimension of H , then

- $m_H(N) \leq \sum_{i=0}^{d_{vc}} \binom{N}{i}$
- $m_H(N) \leq N^{d_{vc}} + 1$
- $m_H(N) = O(N^{d_{vc}})$

If d_{vc} is the VC dimension of H ,
 $d_{vc} + 1$ is a break point for H

Vapnik–Chervonenkis (VC) Bound

- VC Generalization Bound

With prob at least $1 - \delta$

$$E_{out}(g) \leq E_{in}(g) + \sqrt{\frac{8}{N} \ln \frac{4m_H(2N)}{\delta}}$$

- Let d_{vc} be the VC dimension of H , we have $m_H(N) \leq N^{d_{vc}} + 1$. Therefore,

With prob at least $1 - \delta$

$$E_{out}(g) \leq E_{in}(g) + \sqrt{\frac{8}{N} \ln \frac{4((2N)^{d_{vc}} + 1)}{\delta}}$$

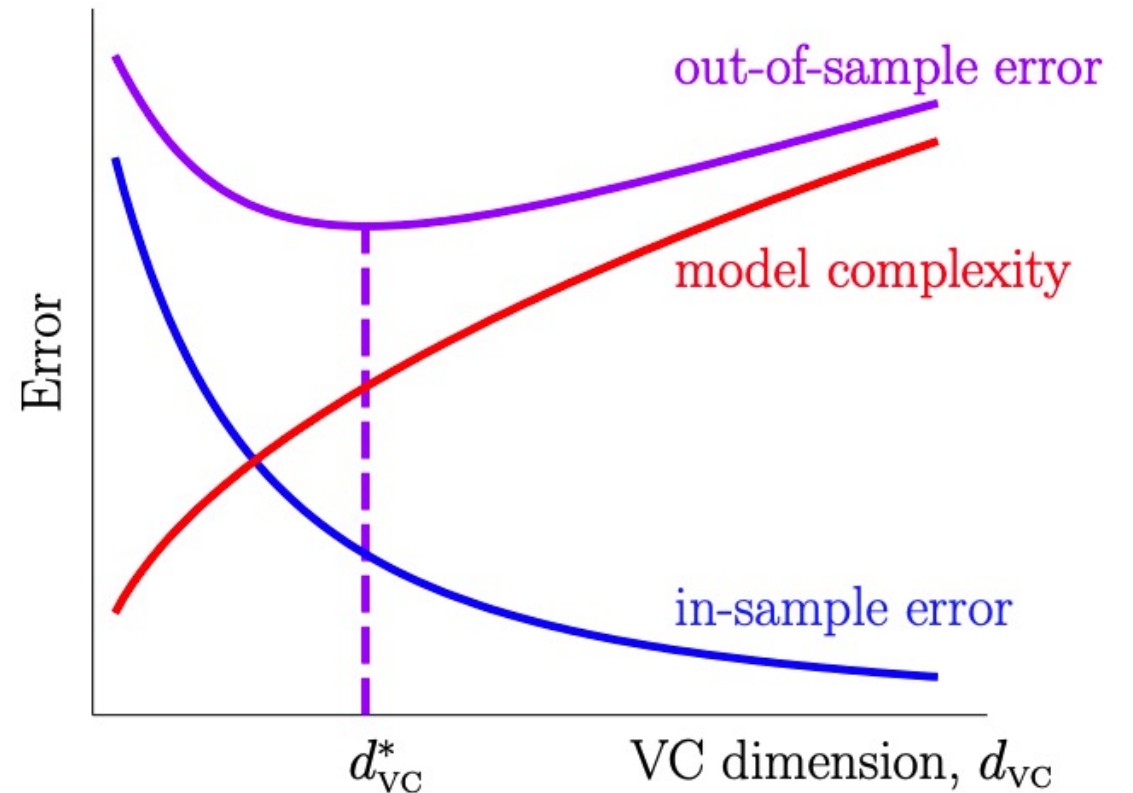
- If we treat δ as a constant, then we can say, with high probability

$$E_{out}(g) \leq E_{in}(g) + O\left(\sqrt{d_{vc} \frac{\ln N}{N}}\right)$$

Approximation-Generalization Tradeoff

- VC Dimension: A single parameter to characterize the complexity of H

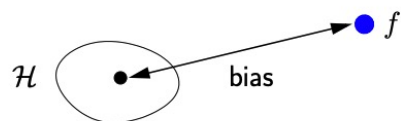
$$E_{out}(g) \leq E_{in}(g) + O\left(\sqrt{d_{vc} \frac{\ln N}{N}}\right)$$



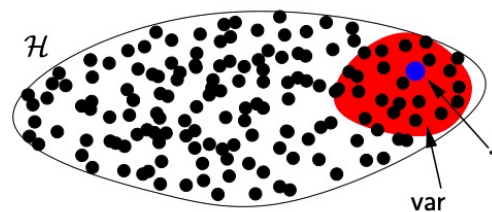
Bias-Variance Decomposition

$$\bullet \mathbb{E}_D[E_{out}(g^{(D)})] = \mathbb{E}_{\vec{x}} \left[\overset{\text{Bias}(\vec{x})}{(\bar{g}(\vec{x}) - f(\vec{x}))^2} \right] + \mathbb{E}_{\vec{x}} \left[\mathbb{E}_D \left[\overset{\text{Var}(\vec{x})}{(g^{(D)}(\vec{x}) - \bar{g}(\vec{x}))^2} \right] \right]$$

- The performance of your learning, i.e., $\mathbb{E}_D[E_{out}(g^{(D)})]$, depends on
 - How well you can fit your data using your hypothesis set (**bias**)
 - How close to the best fit you can get for a given dataset (**variance**)



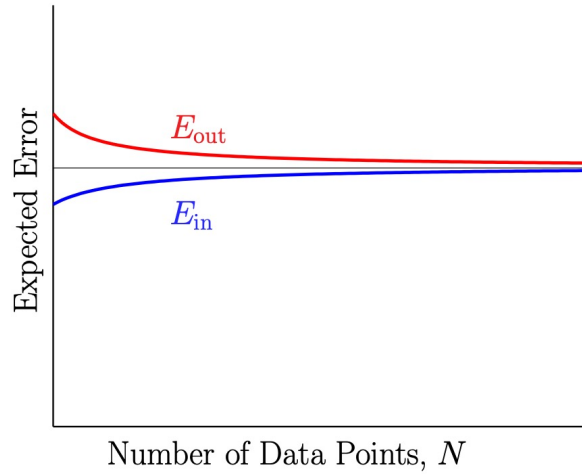
Very small model



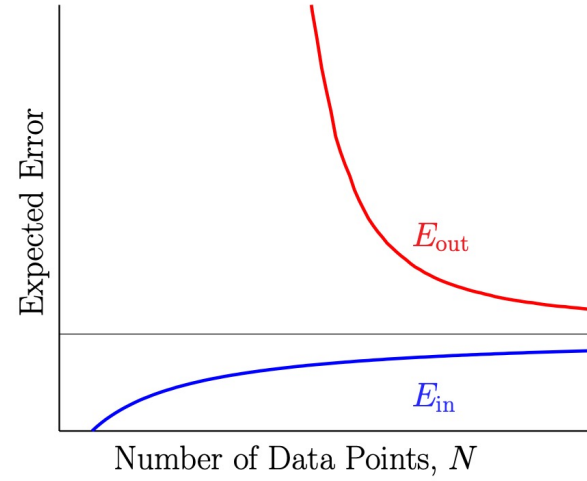
Very large model

Learning Curves

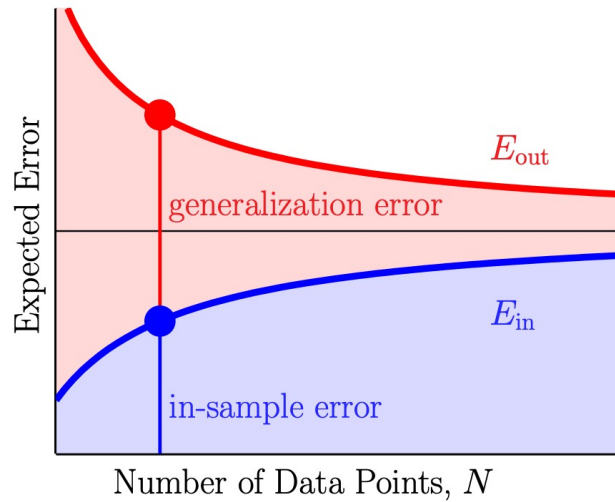
Simple Model



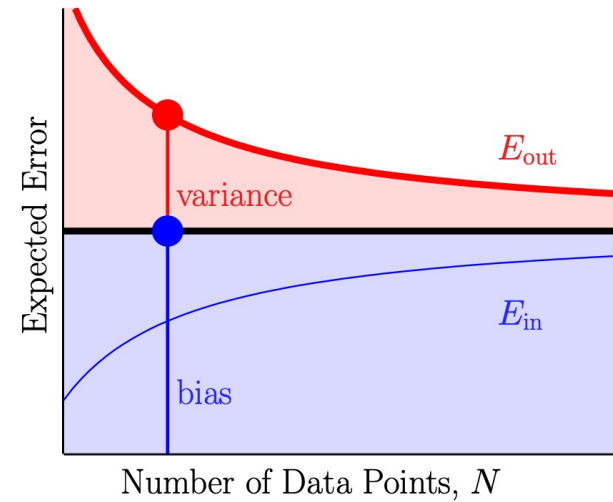
Complex Model



VC Analysis



Bias-Variance Analysis



Linear Models

This is why it's called linear models

- H contains hypothesis $h(\vec{x})$ as **some function of** $\vec{w}^T \vec{x}$

	Domain	Model
Linear Classification	$y \in \{-1, +1\}$	$H = \{h(\vec{x}) = \text{sign}(\vec{w}^T \vec{x})\}$
Linear Regression	$y \in \mathbb{R}$	$H = \{h(\vec{x}) = \vec{w}^T \vec{x}\}$
Logistic Regression	$y \in [0,1]$	$H = \{h(\vec{x}) = \theta(\vec{w}^T \vec{x})\}$

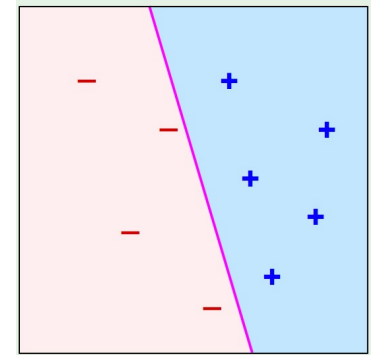
$$\theta(s) = \frac{e^s}{1 + e^s}$$

- Algorithm:
 - Focus on $g = \operatorname{argmin}_{h \in H} E_{in}(h)$

Linear Classification

- Formulation

- Hypothesis set $H = \{h(\vec{x}) = \text{sign}(\vec{w}^T \vec{x})\}$
- Error measure: binary error $e(h(\vec{x}), y) = \mathbb{I}[h(\vec{x}) \neq y]$



- Data is linearly separable

- Run PLA $\Rightarrow E_{in} = 0 \Rightarrow$ Low E_{out}

- Data is not linearly separable

- Engineering the features
- Pocket algorithm

Perceptron Learning Algorithm (PLA)

Initialize $\vec{w}(0) = \vec{0}$

For $t = 0, \dots$

Find a misclassified example $(\vec{x}(t), y(t))$ in D
that is, $\text{sign}(\vec{w}(t)^T \vec{x}(t)) \neq y(t)$

If no such sample exists

Return $\vec{w}(t)$

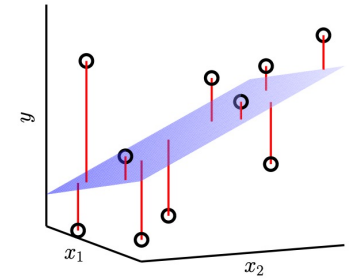
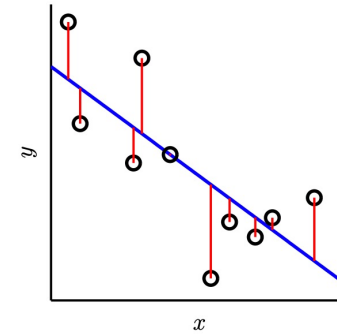
Else

$\vec{w}(t+1) \leftarrow \vec{w}(t) + y(t)\vec{x}(t)$

Linear Regression

- Formulation

- Hypothesis set $H = \{h(\vec{x}) = \vec{w}^T \vec{x}\}$
- Squared error $e(h(\vec{x}), y) = (h(\vec{x}) - y)^2$



- Linear regression algorithm (one-step learning for solving $\nabla_{\vec{w}} E_{in}(\vec{w}_{lin}) = 0$)

- Given $D = \{(\vec{x}_1, y_1), \dots, (\vec{x}_N, y_N)\}$

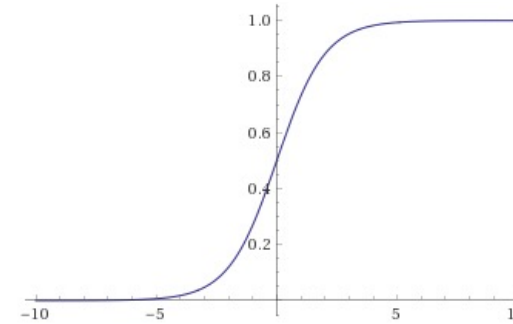
- Construct $X = \begin{bmatrix} \vec{x}_1^T \\ \vdots \\ \vec{x}_N^T \end{bmatrix} = \begin{bmatrix} x_{1,0} & x_{1,1} & \cdots & x_{1,d} \\ x_{2,0} & x_{2,1} & \cdots & x_{2,d} \\ \vdots & \vdots & \ddots & \vdots \\ x_{N,0} & x_{N,1} & \cdots & x_{N,d} \end{bmatrix}$ and $\vec{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$

- Output $\vec{w}_{lin} = (X^T X)^{-1} X^T \vec{y}$ (Assume $X^T X$ is invertible)

Logistic Regression

- Hypothesis set $H = \{h(\vec{x}) = \theta(\vec{w}^T \vec{x})\}$

- $\theta(s) = \frac{e^s}{1+e^s} = \frac{1}{1+e^{-s}}$



- Predict a probability
 - Interpreting $h(\vec{x})$ as the prob for $y = +1$ given \vec{x} when h is the target function
- Algorithm
 - Find $g = \operatorname{argmin}_{h \in H} E_{in}(h)$
- Two key questions
 - How to define $E_{in}(h)$?
 - How to perform the optimization (minimizing E_{in})?

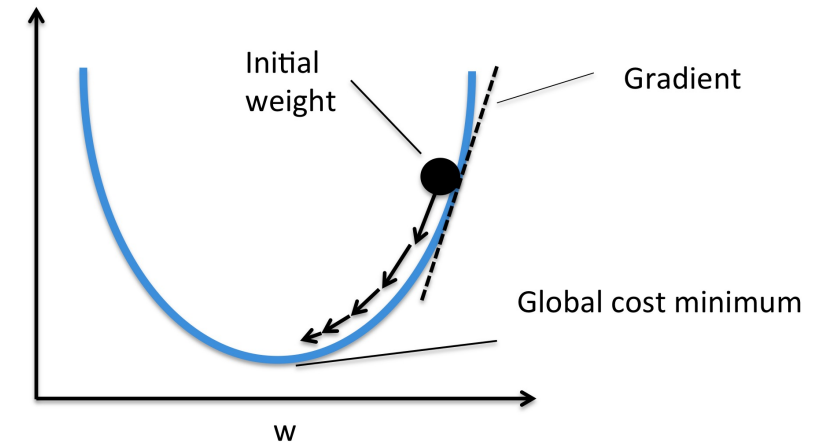
Define $E_{in}(\vec{w})$: Cross-Entropy Error

$$E_{in}(\vec{w}) = \frac{1}{N} \sum_{n=1}^N \ln(1 + e^{-y_n \vec{w}^T \vec{x}_n})$$

- Minimizing cross entropy error is the same as maximizing likelihood
- Likelihood: $\Pr(D|\vec{w})$
 - $\vec{w}^* = \operatorname{argmax}_{\vec{w}} \Pr(D|\vec{w})$ (maximizing likelihood)
 $= \operatorname{argmin}_{\vec{w}} E_{in}(\vec{w})$ (minimizing cross-entropy error)

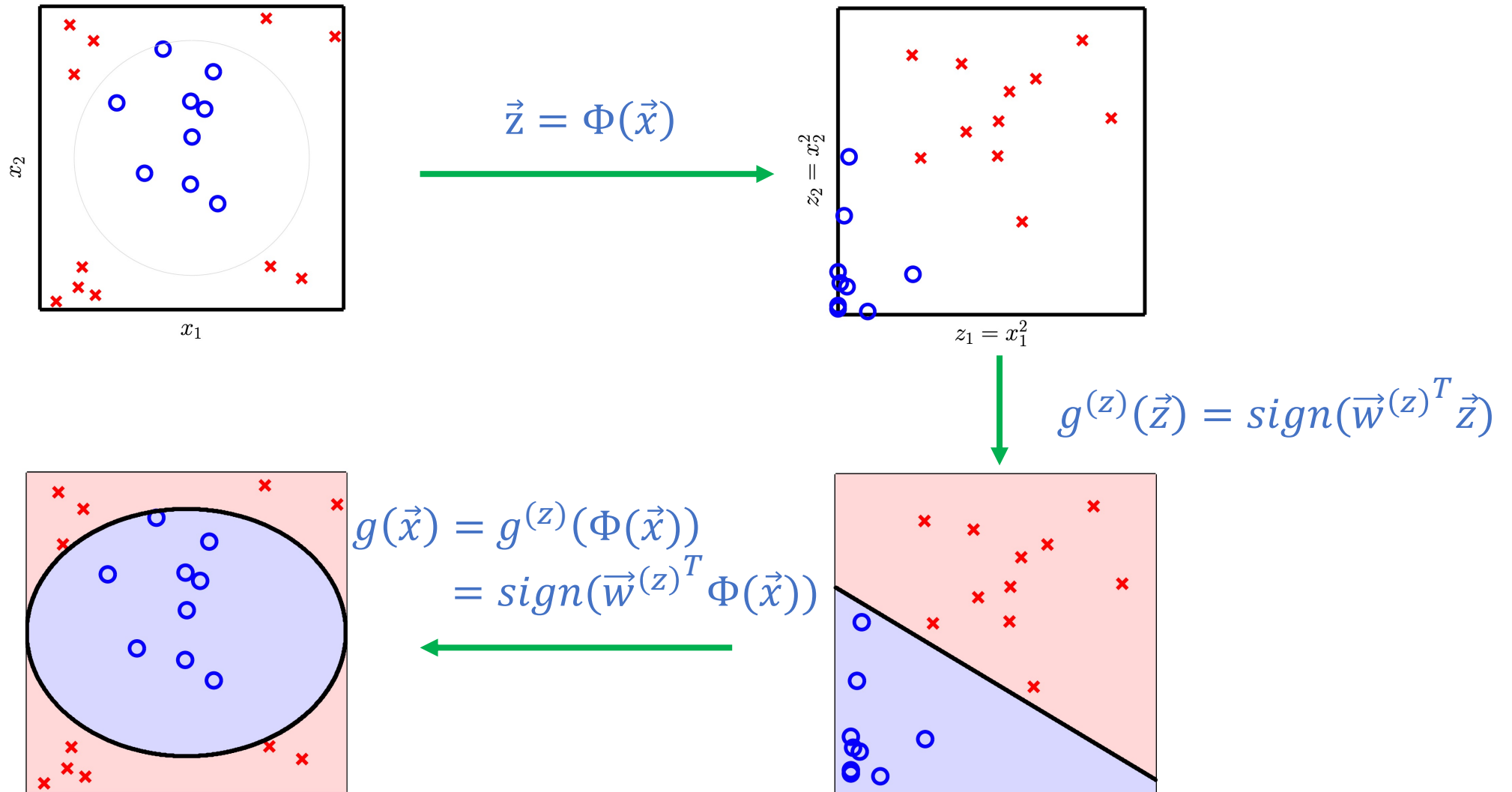
Optimizing $E_{in}(\vec{w})$: Gradient Descent

- Gradient descent algorithm
 - Initialize $\vec{w}(0)$
 - For $t = 0, \dots$
 - $\vec{w}(t+1) \leftarrow \vec{w}(t) - \eta \nabla_{\vec{w}} E_{in}(\vec{w}(t))$
 - Terminate if the stop conditions are met
 - Return the final weights
- Stochastic gradient decent
 - Replace the update step:
 - Randomly choose n from $\{1, \dots, N\}$
 - $\vec{w}(t+1) \leftarrow \vec{w}(t) - \eta \nabla_{\vec{w}} e_n(\vec{w}(t))$



Works for functions where gradient exists everywhere

Nonlinear Transformation



Must Choose Φ **BEFORE** Looking at the Data

- Rely on domain knowledge (feature engineering)
 - Handwriting digit recognition example
- Use common sets of feature transformation
 - Polynomial transformation
 - E.g., 2nd order Polynomial transformation
 - $\vec{x} = (1, x_1, x_2)$, $\Phi_2(\vec{x}) = (1, x_1, x_2, x_1x_2, x_1^2, x_2^2)$
 - Plus: more powerful (contains circle, ellipse, hyperbola, etc)
 - Minus:
 - More computation/storage
 - Worse generalization error

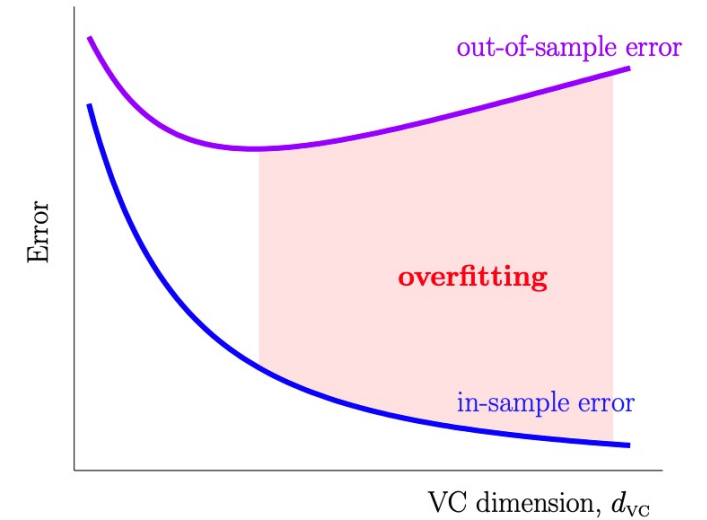
The VC dimension of d-dim perceptron is d+1

Q-th Order Polynomial Transform

- $\vec{x} = (1, x_1, \dots, x_d)$
- $\Phi_1(\vec{x}) = \vec{x}$
- $\Phi_Q(\vec{x}) = (\Phi_{Q-1}(\vec{x}), x_1^Q, x_1^{Q-1}x_2, \dots, x_d^Q)$
- Each element in $\Phi_Q(\vec{x})$ is in the form of $\sum_{i=1}^d x_i^{a_i}$
 - where $\sum_{i=1}^d a_i \leq Q$, and a_i is a non-negative integer
- Number of elements in $\Phi_Q(\vec{x})$: $\binom{Q+d}{Q}$ (including the initial 1)

Overfitting and Its Cures

- Overfitting
 - Fitting the data more than is warranted
 - Fitting the noise instead of the pattern of the data
 - Decreasing E_{in} but getting larger E_{out}
 - When H is too strong, but N is not large enough
- Regularization
 - Intuition: Constraining H to make overfitting less likely to happen
- Validation
 - Intuition: Reserve data to estimate E_{out}



Regularization

- Constraining H

- Example: Weight decay $H(C) = \{h \in H_Q \text{ and } \vec{w}^T \vec{w} \leq C\}$
- Finding $g \Rightarrow$ Constrained optimization

minimize $E_{in}(\vec{w})$
subject to $\vec{w}^T \vec{w} \leq C$

- Defining augmented error

- $E_{aug}(h, \lambda, \Omega) = E_{in}(\vec{w}) + \frac{\lambda}{N} \Omega(h)$
- Finding $g \Rightarrow$ Unconstrained optimization

minimize $E_{in}(\vec{w}) + \frac{\lambda_c}{N} \vec{w}^T \vec{w}$

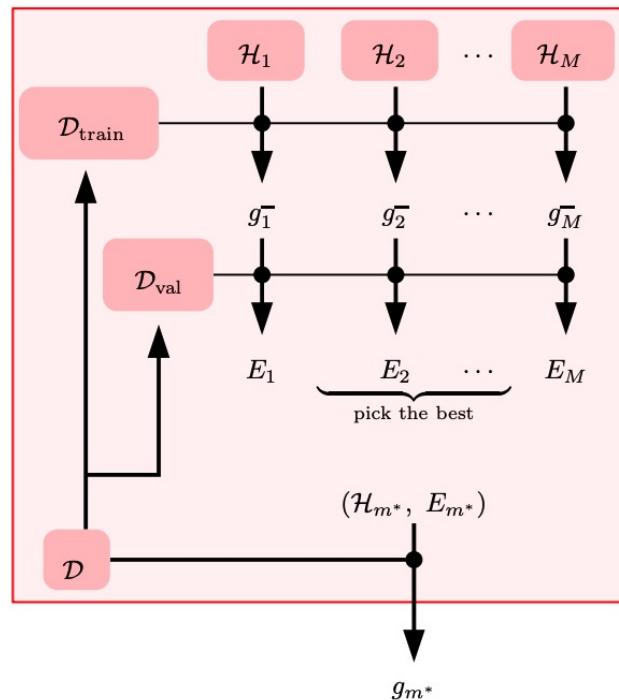
- The two interpretations are conceptually equivalent in a lot of cases.

- Understand the impacts of choosing Ω and λ

Validations

- Reserving data to estimate E_{out}

Model Selection

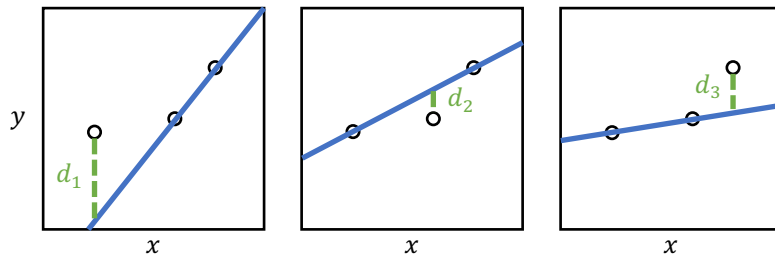
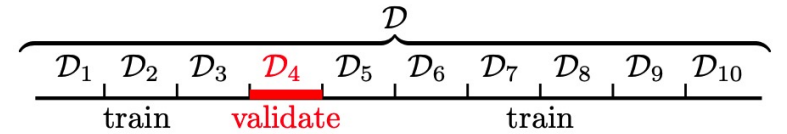


	Outlook	Relationship to E_{out}
E_{in}	Incredibly optimistic	VC-bound
E_{val} (when used for model selection)	Slightly optimistic	Hoeffding's bound (multiple hypotheses)
E_{test}	Unbiased	Hoeffding's bound (single hypothesis)

Cross Validation

- Split D into V equally sized data sets: D_1, D_2, \dots, D_V
 - Let g_i^- be the hypothesis learned using all data sets except D_i
 - Let $e_i = E_{val}(g_i^-)$ where the validation uses data set D_i

- The V -fold cross validation error is $\frac{1}{V} \sum_{i=1}^V e_i$
- Leave-One-Out Cross Validation (LOOCV): $V = N$



$$E_{cv} = \frac{1}{3}(d_1^2 + d_2^2 + d_3^2)$$

Three Learning Principles

- Occam's Razor
 - The **simplest** model that fits the data is also the most **plausible**
- Sampling Bias
 - If the data is sampled in a **biased** way, learning will produce a similarly **biased** outcome.
- Data Snooping
 - If a data set has affected any step in the learning process, its ability to assess the outcome has been compromised.

Practice Questions

Don't view these as good representations of exam questions.

But it should give you a sense of what the exam questions might look like.

What to Expect for Exam Questions

- 50 points in total
 - 5 (+/- 1) long questions, 30 points
 - Written response questions with explanations required
 - Might earn partial credits
 - 10 multiple choice questions, 20 points
 - No explanations needed
 - No partial credits
 - No penalty for choosing the wrong answer