CSE 417T
Introduction to Machine Learning

Lecture 14

Instructor: Chien-Ju (CJ) Ho

Logistics: Exam 1

- Exam 1 Date: March 3, 2020 (Tuesday)
  - In-class exam (the same time/location as the lecture)
  - Exam duration: 75 minutes
  - Planned exam content: LFD Chapter 1 to 5
    - Everything in textbook/lectures are included, except for parts labeled as “safe to skip”.

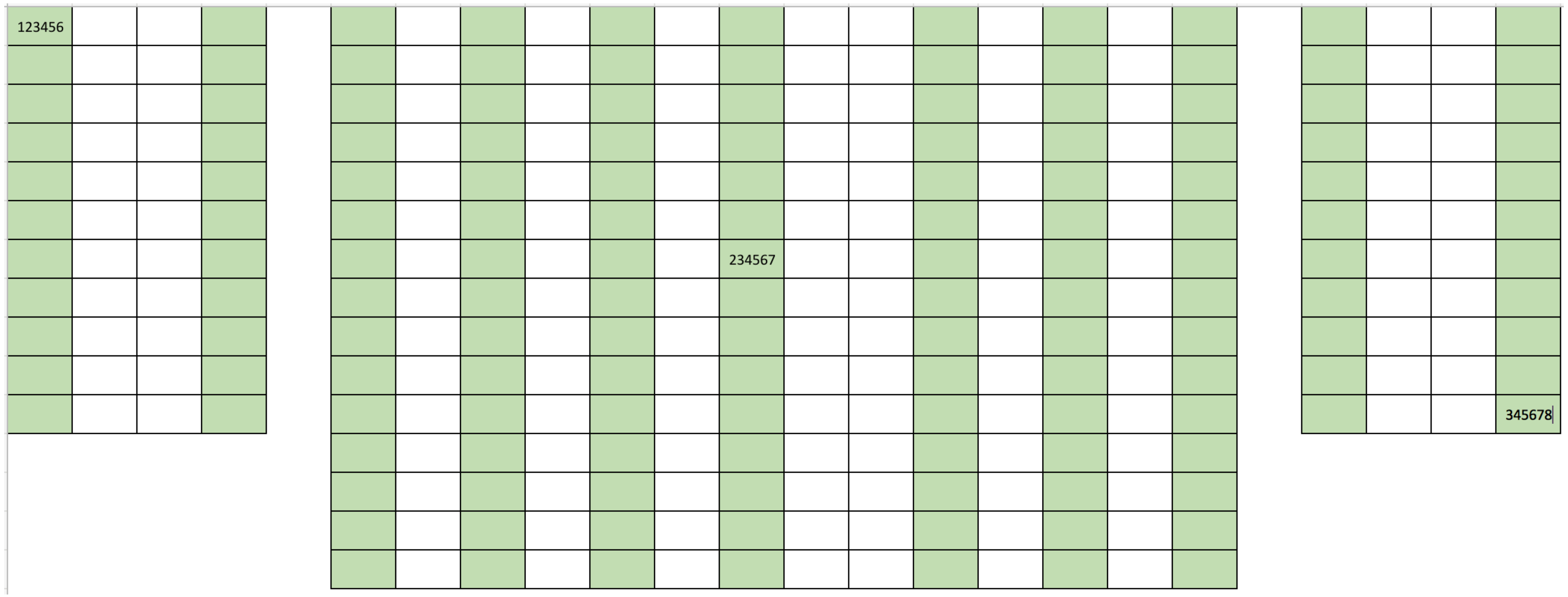
  - 2 sections
    - 5~6 long questions (written response questions with explanations required)
    - 10 multiple choice questions (no explanations needed)

  - Closed-book exam. You can bring two cheat-sheets
    - Up to letter size, front and back (up to 4 pages)
    - No format limitations (it can be typed, written, or a combination)

  - No calculators (you don’t need them)

Logistics: Exam Policies

- I plan to arrange random seat assignments
  - Will be announced on Piazza the night before the exam



Logistics: Exam Policies

- Please arrive on time. No extensions will be given if you arrive late.

- During the exam, if you have a question or if you finish before time is up:
  - **<u>Do not get up</u>**
  - Raise your hand and I will come to you
  - I most likely will not answer questions to individual students
    - But I'll give clarifications to everyone if multiple students ask the same question

- When time is called:
  - **<u>Stop writing</u>**
  - **<u>Do not get up</u>**
  - Proctors will come around and collect your exam

Plans for Today

- A summary of the content so far.

- 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.

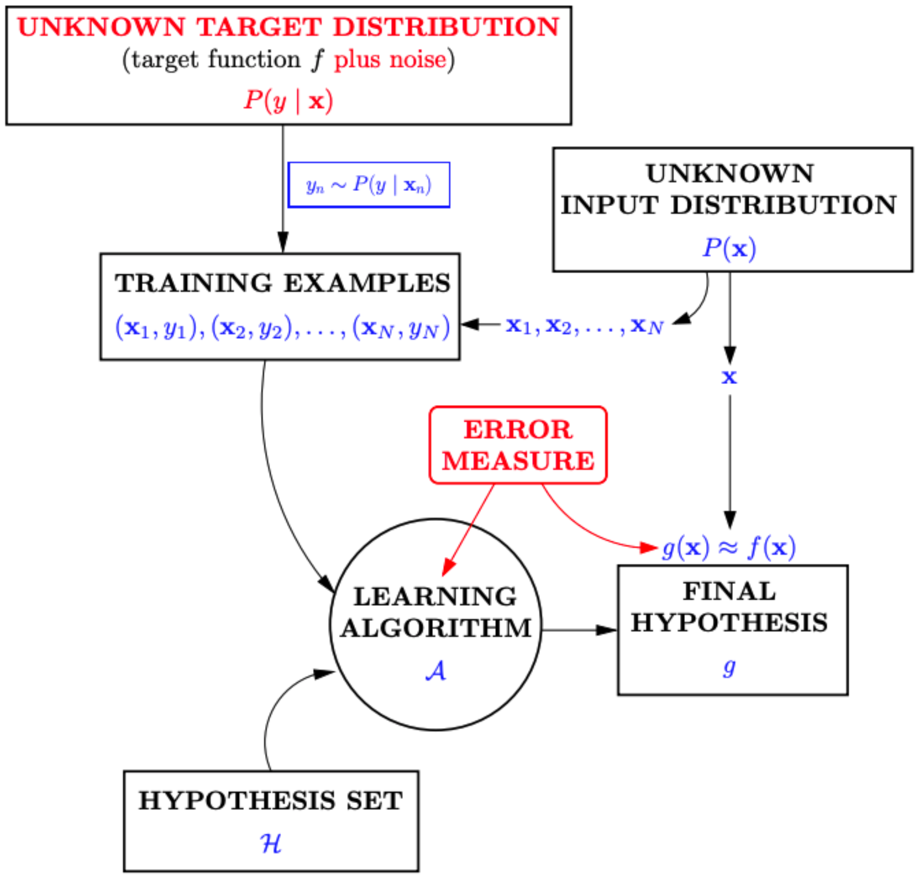
Please make sure you still study for LFD Chapter 1-5.

Let me know if you find mistakes in lecture notes.

Whenever you have doubts on the lecture notes, please resort to 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

$$\begin{array}{cc|cc} & & \multicolumn{2}{c}{f} \\ & & +1 & -1 \\ \hline h & +1 & 0 & 1 \\ & -1 & 10 & 0 \end{array}$$

Supermarket

$$\begin{array}{cc|cc} & & \multicolumn{2}{c}{f} \\ & & +1 & -1 \\ \hline h & +1 & 0 & 1000 \\ & -1 & 1 & 0 \end{array}$$

CIA

Hoeffding's Inequality

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

- 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: $M \to \infty$

- Instead of # hypothesis, counting "effective" # hypothesis
- <u>Dichotomy</u>
  - 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 $\vec{x}_1, \dots, \vec{x}_N$

- <u>Growth function</u>
  - 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 $1-\delta$,

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

- VC Generalization Bound (VC Inequality, 1971)

  With prob at least $1-\delta$

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

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

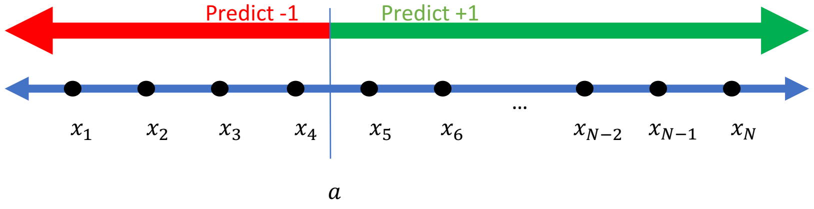
- More definitions….
  - Shatter
    - H **shatters** $(\vec{x}_1, \dots, \vec{x}_N)$ if $|H(\vec{x}_1, \dots, \vec{x}_N)| = 2^N$
    - H can induce all label combinations for $(\vec{x}_1, \dots, \vec{x}_N)$
  - Break point
    - k is a **break point** for H if no data set of size k can be shattered 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

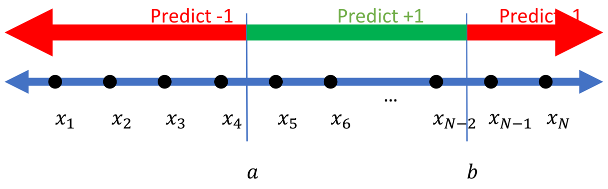
$\boldsymbol{m_H(N)}$

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

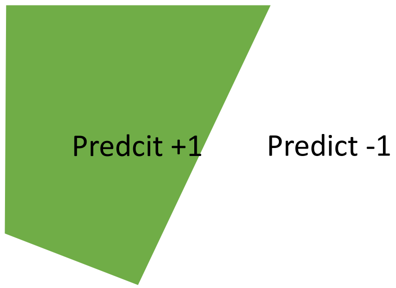
Positive Rays



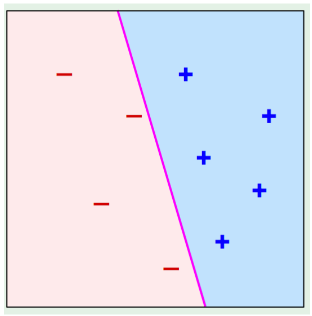
Positive Intervals



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, then

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

- Rephrase 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 $1-\delta$

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

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

  With prob at least $1-\delta$

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

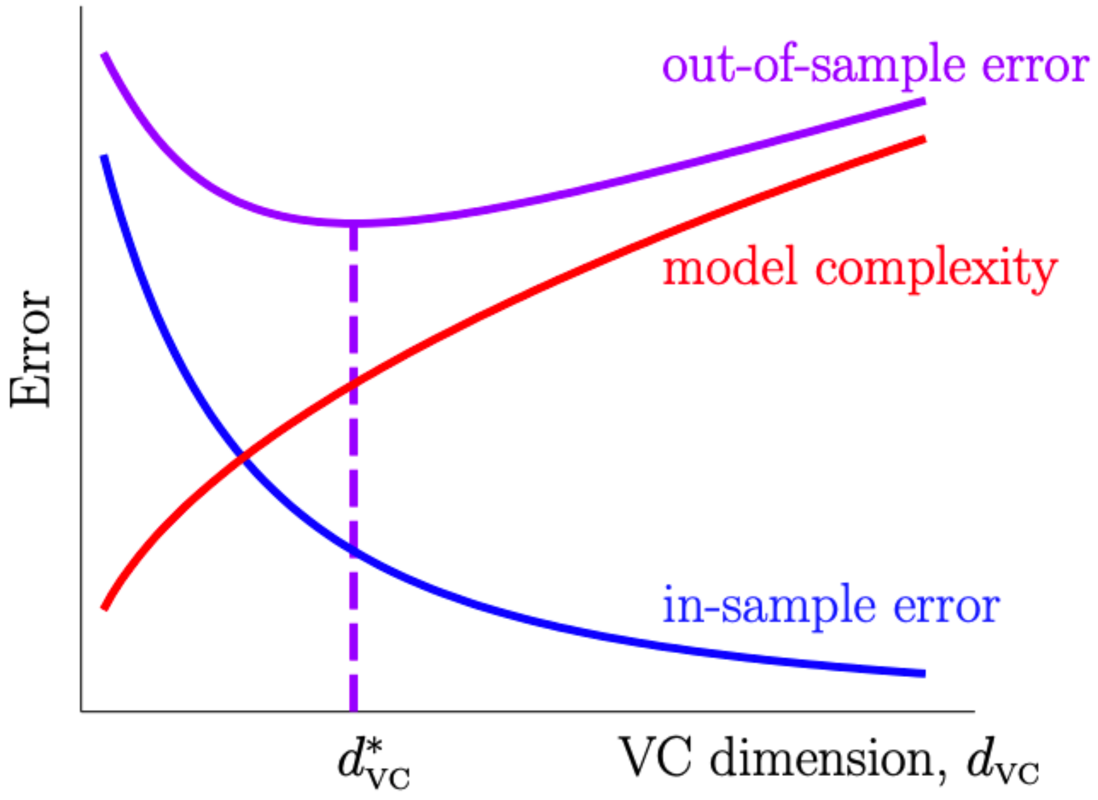
- If we <u>treat δ as a constant</u>, then we can say, with high probability

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

Approximation-Generalization Tradeoff

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

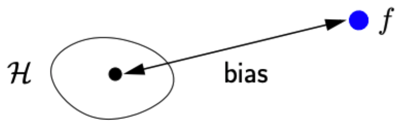
$$E_{out}(g) \leq E_{in}(g) + O\left(\sqrt{d_{vc}\frac{\ln N}{N}}\right)$$



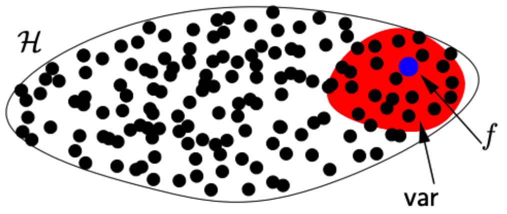
Bias-Variance Decomposition

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

- The performance of your learning, i.e.,$\mathbb{E}_D\left[E_{out}\left(g^{(D)}\right)\right]$, 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)

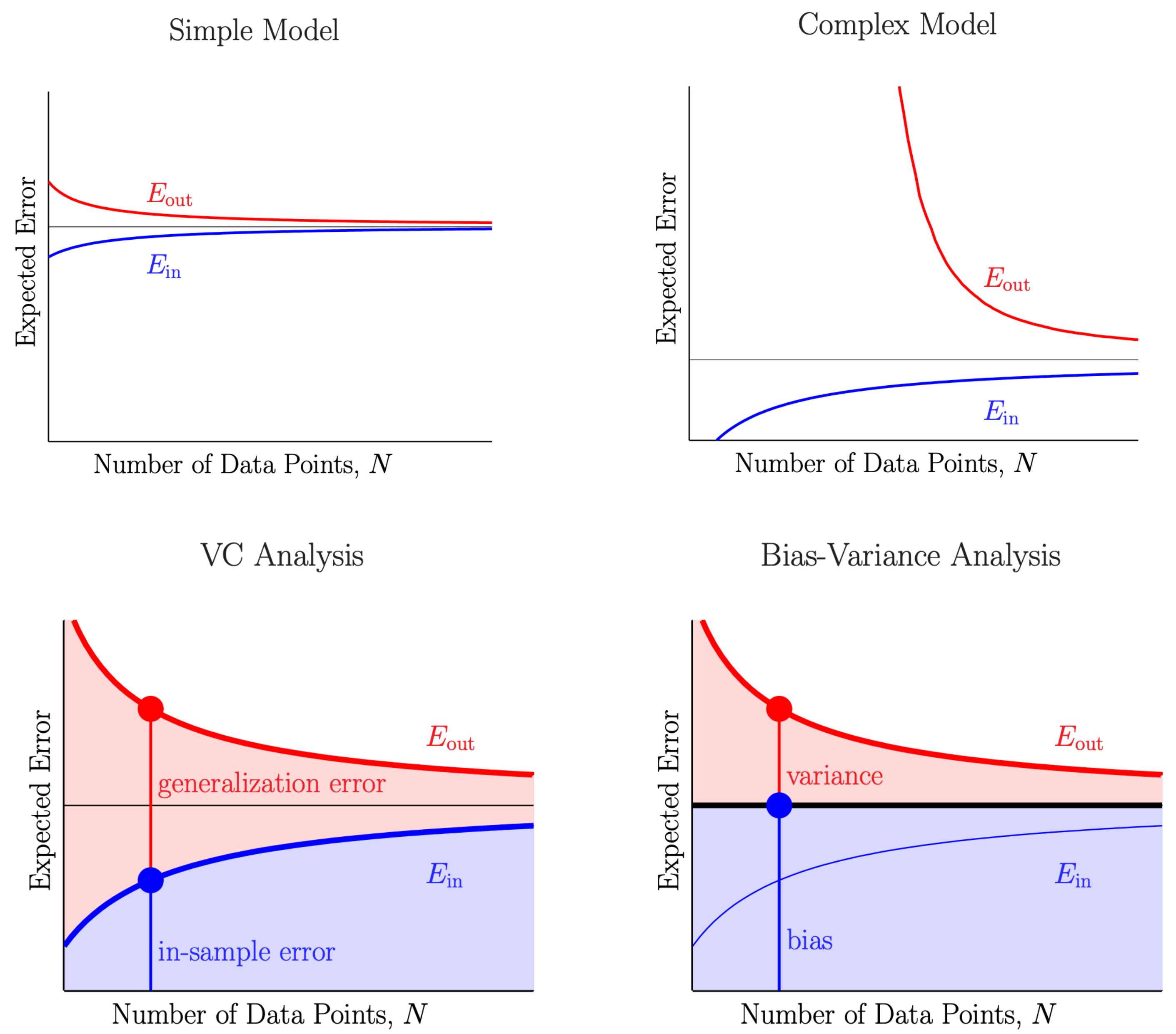


Very small model



Very large model

Learning Curves



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	$y \in \{-1, +1\}$	$H = \{h(\vec{x}) = sign(\vec{w}^T\vec{x})\}$
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 $g = argmin_{h \in H} E_{in}(h)$

Linear Classification

- Formulation
  - Hypothesis set $H = \{h(\vec{x}) = sign(\vec{w}^T \vec{x})\}$
  - Error measure: binary error $e(h(\vec{x}), y) = \mathbb{1}[h(\vec{x}) \neq y]$

- Data is linearly separable
  - Run PLA => $E_{in} = 0$ => Low E_{out}

- Data is not linearly separable
  - Engineering the features
  - Pocket algorithm

<u>Perceptron Learning Algorithm (PLA)</u>

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

For $t = 0, \ldots$

    Find a misclassified example $(\vec{x}(t), y(t))$ in D

        that is, $\mathrm{sign}(\vec{w}(t)^T \vec{x}(t)) \neq y(t)$

    If no such sample exists

        Return $\vec{w}(t)$

    Else

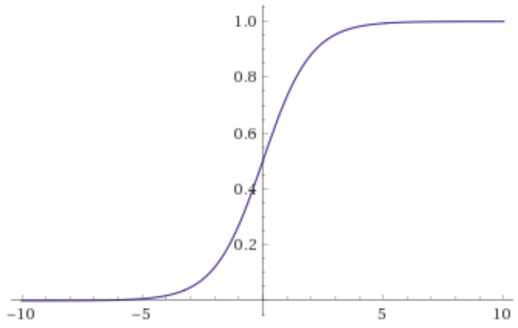
        $\vec{w}(t+1) \leftarrow \vec{w}(t) + y(t)\vec{x}(t)$

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 $X = \begin{bmatrix} \vec{x}_1^T \\ \vdots \\ \vec{x}_N^T \end{bmatrix} = \begin{bmatrix} x_{1,0} & x_{1,1} & \cdots & x_{1,d} \\ x_{2,0} & x_{2,1} & \cdots & x_{2,d} \\ \vdots & \vdots & \ddots & \vdots \\ x_{2,0} & x_{N,1} & \cdots & x_{N,d} \end{bmatrix}$ and $\vec{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$
  - Output $\vec{w}_{lin} = (X^TX)^{-1}X^T\vec{y}$ (Assume X^TX 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 $g = argmin_{h \in H} E_{in}(h)$
- Two key questions
  - How to define $E_{in}(h)$?
  - How to perform the optimization (minimizing E_{in})?

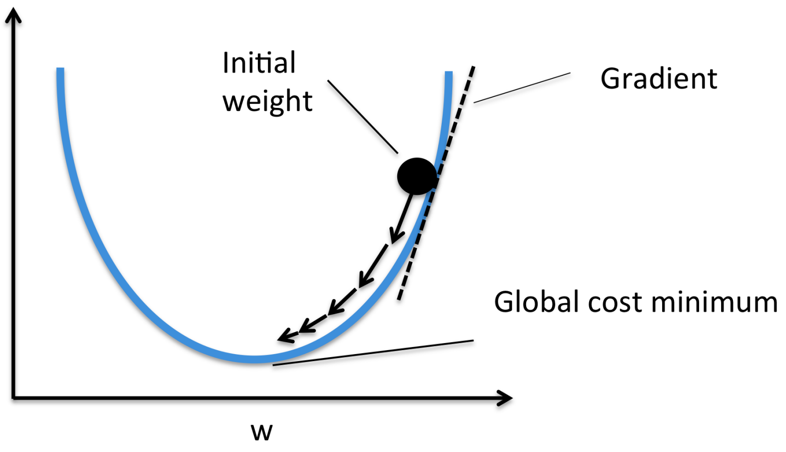
Define $E_{in}(\vec{w})$: Cross-Entropy Error

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

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

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

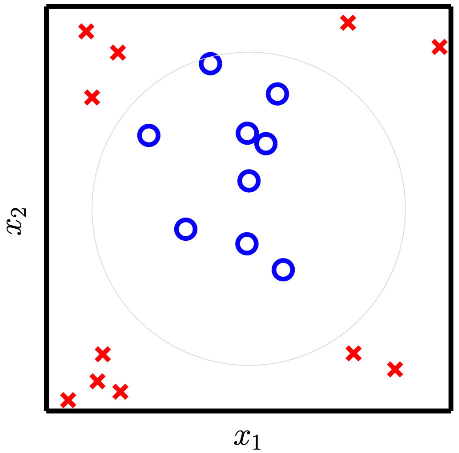
- Gradient descent algorithm
  - Initialize $\vec{w}(0)$
  - For $t = 0, \ldots$
    - $\vec{w}(t+1) \leftarrow \vec{w}(t) - \eta \nabla_{\vec{w}} E_{in}\big(\vec{w}(t)\big)$
    - Terminate if the stop conditions are met
  - Return the final weights



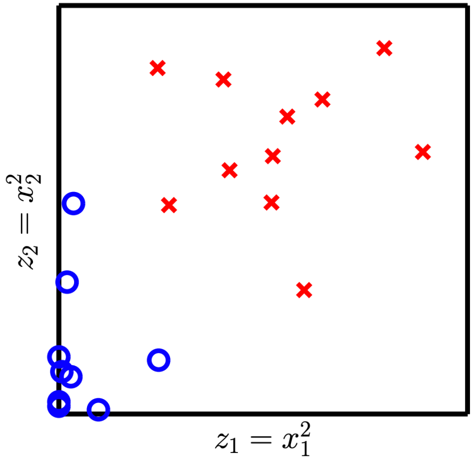
Works for functions where gradient exists everywhere

- Stochastic gradient decent
  - Replace the update step:
    - Randomly choose n from $\{1, \ldots, N\}$
    - $\vec{w}(t+1) \leftarrow \vec{w}(t) - \eta \nabla_{\vec{w}} e_n(\vec{w}(t))$

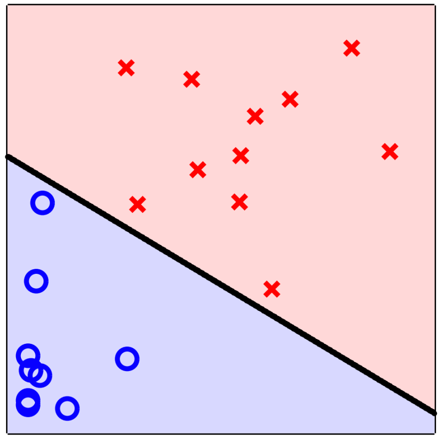
Nonlinear Transformation



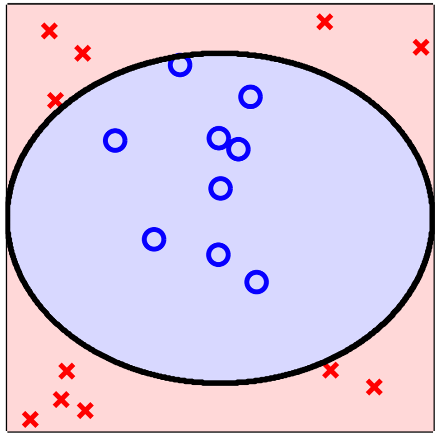
$$\vec{z} = \Phi(\vec{x})$$



$$g^{(z)}(\vec{z}) = sign({\vec{w}^{(z)}}^T \vec{z})$$



$$g(\vec{x}) = g^{(z)}(\Phi(\vec{x}))$$
$$= sign({\vec{w}^{(z)}}^T \Phi(\vec{x}))$$



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
    - $\vec{x} = (1, x_1, x_2),\ \Phi_2(\vec{x}) = (1, x_1, x_2, x_1 x_2, x_1^2, x_2^2)$
    - 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

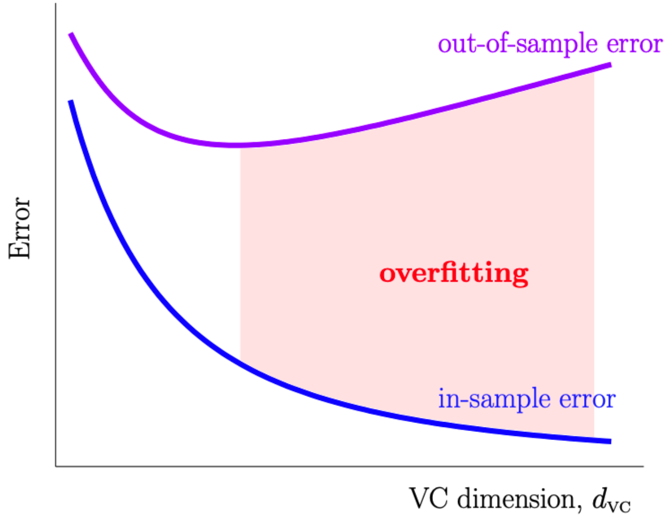
- $\vec{x} = (1, x_1, \ldots, x_d)$
- $\Phi_1(\vec{x}) = \vec{x}$
- $\Phi_Q(\vec{x}) = (\Phi_{Q-1}(\vec{x}), x_1^Q, x_1^{Q-1} x_2, \ldots, x_d^Q)$

- 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 => 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 => Unconstrained optimization

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

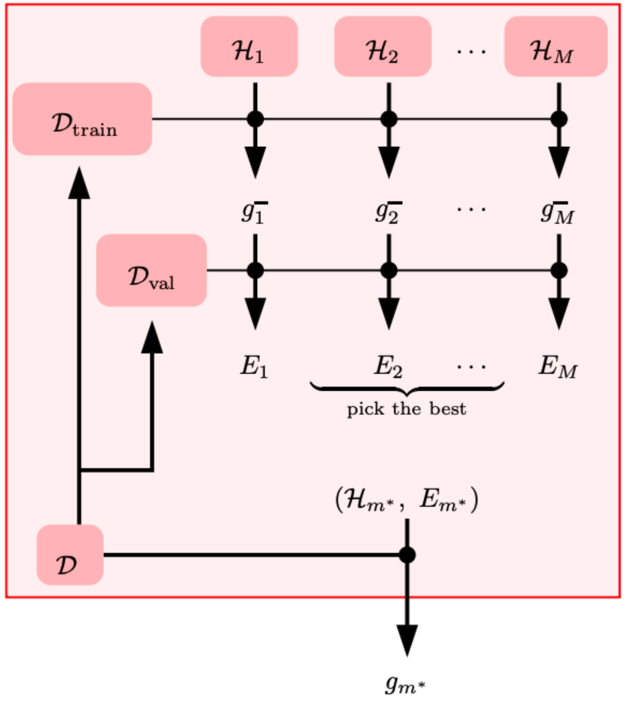
- The two interpretations are conceptually equivalent in a lot of cases.

- **Understand the impacts of choosing $\boldsymbol{\Omega}$ and $\boldsymbol{\lambda}$**

Validations

- Reserving data to estimate E_{out}

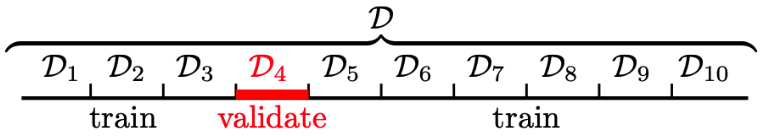
Model Selection



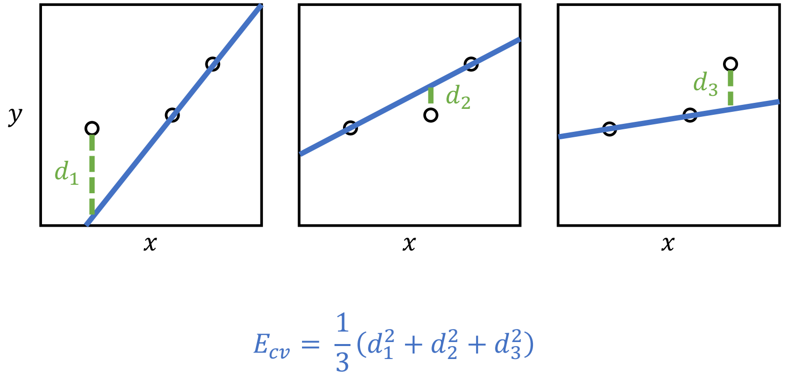
	Outlook	Relationship to E_{out}
E_{in}	Incredibly optimistic	VC-bound
E_{val}	Slightly optimistic	Hoeffding’s bound (multiple hypotheses)
E_{test}	Unbiased	Hoeffding’s bound (single hypothesis)

Cross Validation

- Split D into V equally sized data sets: $D_1, D_2, \ldots, D_V$
  - 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): $V = N$



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