CS 461: Machine Learning Principles

Class 15: Oct. 24

Learning Theory

and Quiz2

Instructor: Diana Kim

Learning Theory

+ computing the bounds for # data and size of hypothesis to get small generalization error with high probability

Theoretical bounds to compute,

(1) how many data samples are required given the size of $|H|$

to achieve |train error – true error| $< \epsilon$ with probability at least (1-δ)?

(2) how large should the hypothesis space be given the data size of m

to ensure that |train error – true error| $< \epsilon$ with probability at least (1-δ)?

+the population averaged error of binary classification is the probability of occurrence error.
because the distance between 1 or 0 (Bernoulli R.V) E[X] = p

Empirical vs. Population Binary Classification Error

$$R(w) = \int \frac{1}{2}|y - f(x;w)| f(x,y)\,dx\,dy$$

$$R_{emp}(w) = \frac{1}{2N}\sum_{n=1}^{N} |y_n - f(x_n, y)|$$, where $y \in \{-1,1\}$

The bounds about # data and hypothesis dimension
, we are going to see,
are for the case of binary classification.

[Haussler's 88 Bound] $f(x;w)$ Q: What is the meaning of this?

- P[a hypothesis true error $> \epsilon$ but consistent with N data samples] $\leq (1-\epsilon)^N$
  * a hypothesis true error $> \epsilon$ implies that error (h) $> \epsilon$
  where error (h) is the probability that h misclassify a new sample.

+computing a loose bound

- P[any learned hypothesis true error $> \epsilon$ but consistent with N data samples]
  $\leq |H|\ (1-\epsilon)^N$ $\rightarrow$ a finite |H|

- P[any consistent hypothesis true error $> \epsilon$]
  $\leq |H|\ (1-\epsilon)^N \leq |H|e^{-N\epsilon} \leq \delta$ [*use the bound*: $1 - x \leq exp^{-x}$]
- With prob at least 1- δ, any consistent hypothesis's true error $\leq \epsilon$
  when

$$N \geq \frac{\ln|H| + \ln\frac{1}{\delta}}{\epsilon}$$

But in reality, training error may not be consistent with training data.
Training error is not zero.
We are interested in P[|train error – true error| $< \epsilon$]

[Hoeffding's Bound] (https://cs229.stanford.edu/extra-notes/hoeffding.pdf)

- P[|train error (h) – true error (h) | $\geq \epsilon$] $\leq 2e^{-2N\epsilon^2}$
- P[any learned hypothesis |train error (h) – true error (h) | $\geq \epsilon$]
  $\leq 2|H|e^{-2N\epsilon^2}$

- true error $(h) \leq$ train error (h) + $\sqrt{\dfrac{\ln|H| + \ln\dfrac{2}{\delta}}{2m}}$, with prob at least $1-\delta$.

- With prob at least 1- δ, |train error (h) – true error (h)| $< \epsilon$
  when

$$N \geq \frac{1}{2\epsilon^2}(\ln|H| + \ln\frac{2}{\delta})$$

In practice, hypothesis space dimension |H| can be infinite and continuous.
We need another bound.

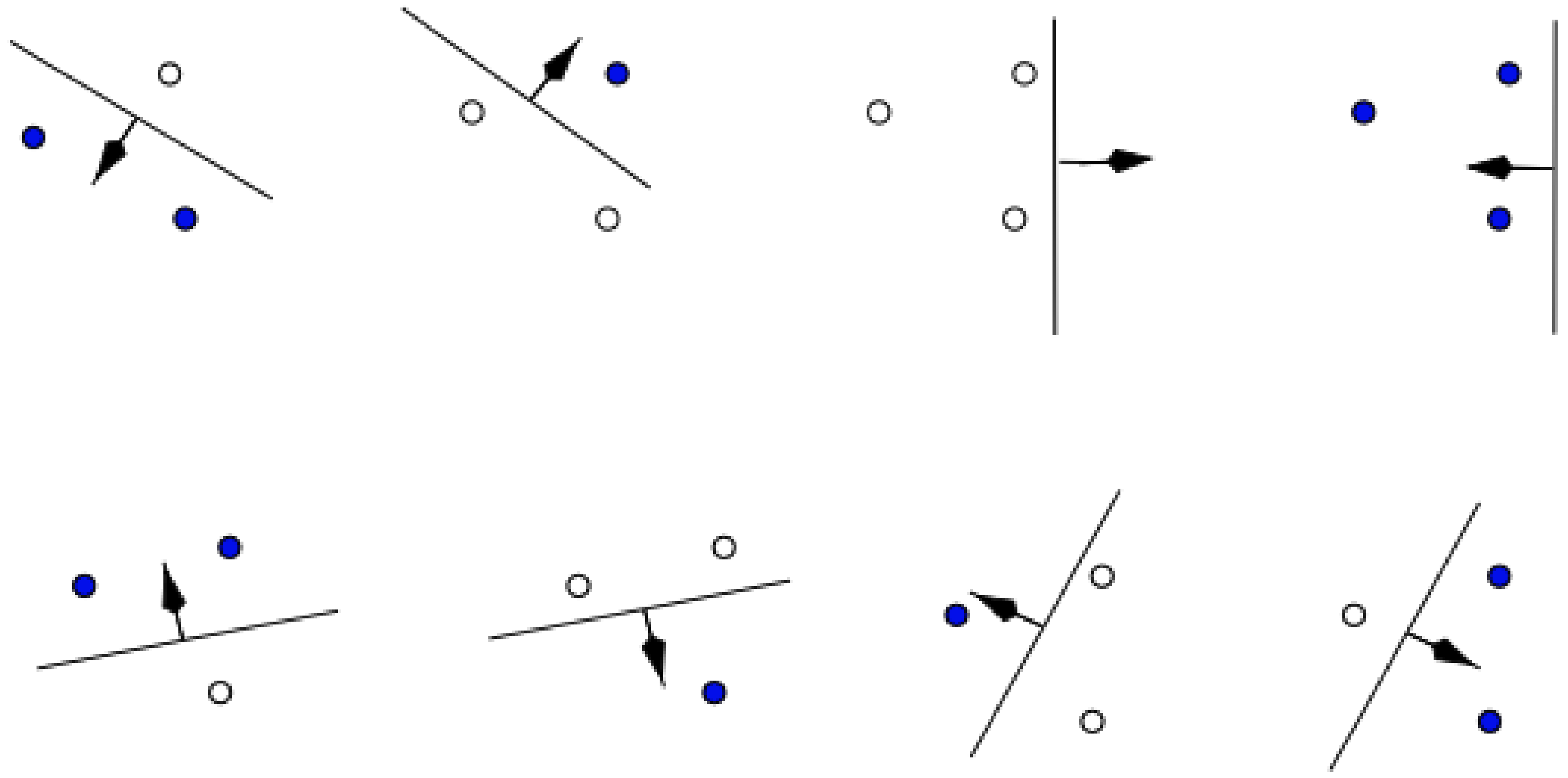
[VC (Vapnik–Chervonenkis) Bound]

The bound is hold with the probability at least $1 - \delta$.

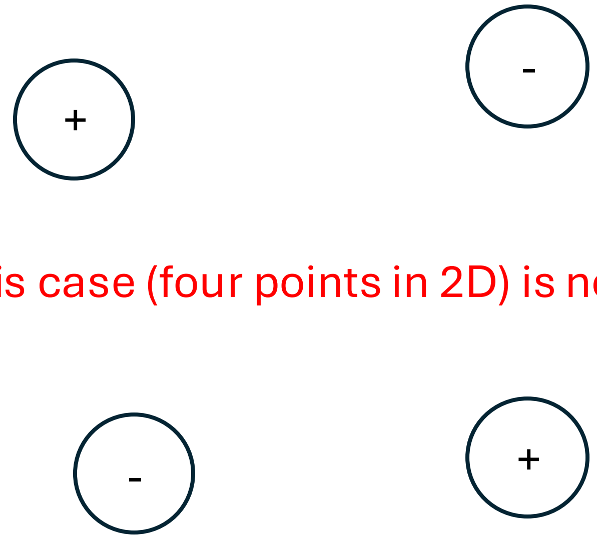
- true error $(h) \leq$ train error (h) + $\sqrt{\underbrace{\frac{VC(H)(\ln(2N/VC(H))+1)+\ln 4/\delta}{N}}_{\text{[The right term called “VC confidence”]}}}$

VC dimension for the set of functions $\boldsymbol{f(x;w)}$:
the maximum number N of training points $(x_1, x_2, \ldots, x_N)$
that can be separated into two classes
in all 2^N possible labeling configurations by $f(x;w)$.

The Example: **VC dimension of the hyperplanes in 2D**.
The three data points in any labeling cases are shattered by the hyperplane.



How about four data points?
What labeling case is not separable by the hyperplane?



+ this case (four points in 2D) is not separable by a hyperplane.

Theorem1]
Consider a set of m points in $\mathcal{R}^n$. Choose any one of points as origin. The the m points can be shattered by oriented hyperplane if and only if the position vector of the remaining points are linearly independent.

Corollary]
VC dimensions of the set of hyperplanes in $R^N is\ N + 1$.

$\mapsto$ we can have N independent vectors in R^N.
Then we can add one vector for the origin. Then the N+1 vectors will be shattered by the oriented hyperplanes by the theorem1.

- VC Dimensions for SVM with P-Polynomial Kernels

SVM with P-polynomial finds <u>a hyperplane in the P-polynomial feature space.</u>
We need to compute the dimensionality of the feature space
for the P-polynomials case to compute the VC dimension.

Let L be the original data dimension then
we can compute the dimensionality of the feature space
by computing # all possible solutions for the e$quation$: $X_1 + X_2 + \ldots X_L = $ p
where p =0,1,…P
The the dimensionality of the feature space is
$\sum_{p=0}^{P} \binom{p+L-1}{p}$(choose P from different L elements with repetition)

Hence VC dimension for SVM with P-Polynomial Kernels is $\sum_{p=0}^{P} \binom{p+L-1}{p}$+1

Hence VC dimension for SVM with P-Polynomial Kernels is $\sum_{p=0}^{P} \binom{p+L-1}{p} + 1$

$\sum_{p=0}^{P} \binom{p+L-1}{p} + 1 = \binom{P+L}{P} + 1$

- VC Dimensions for SVM with *Gaussian Kernel*

SVM with Gaussian kernels finds a hyperplane in infinite dimensional feature space. In the space, we always can find any finite data points that are independent.

$$\kappa(x_1, x_2) = \exp \frac{-||x_1 - x_2||^2}{2\sigma^2}$$

How?

+ we can always pick two data points far enough making the kernel value zero.
  Then, two points are independent (inner product is zero, orthogonal) .
  The number of data points can be any finite. Hence, VC for svm is infinite.

The independent data points can be defined as infinitely many.
Hence, VC dimension for SVM with *Gaussian Kernel* will be infinite.

Computing the Lower Bounds for # Data Samples using VC Dimension

- true error $(h) \leq$ train error (h) + $\sqrt{\dfrac{VC(H)(\ln(2N/VC(H))+1)+\ln 4/\delta}{N}}$

$$\epsilon \geq \sqrt{\frac{VC(H)(\ln(2N/VC(H))+1)+\ln 4/\delta}{N}}$$

$$\epsilon \geq \sqrt{\frac{VC(H)(1-\frac{VC(H)}{2N}+1)+\ln 4/\delta}{N}}$$

$$N\epsilon^2 \geq VC(H)(1-\frac{VC(H)}{2N}+1)+\ln 4/\delta$$

$$N\epsilon^2 \geq 2VC(H)-\frac{VC(H)^2}{2N}+\ln 4/\delta$$

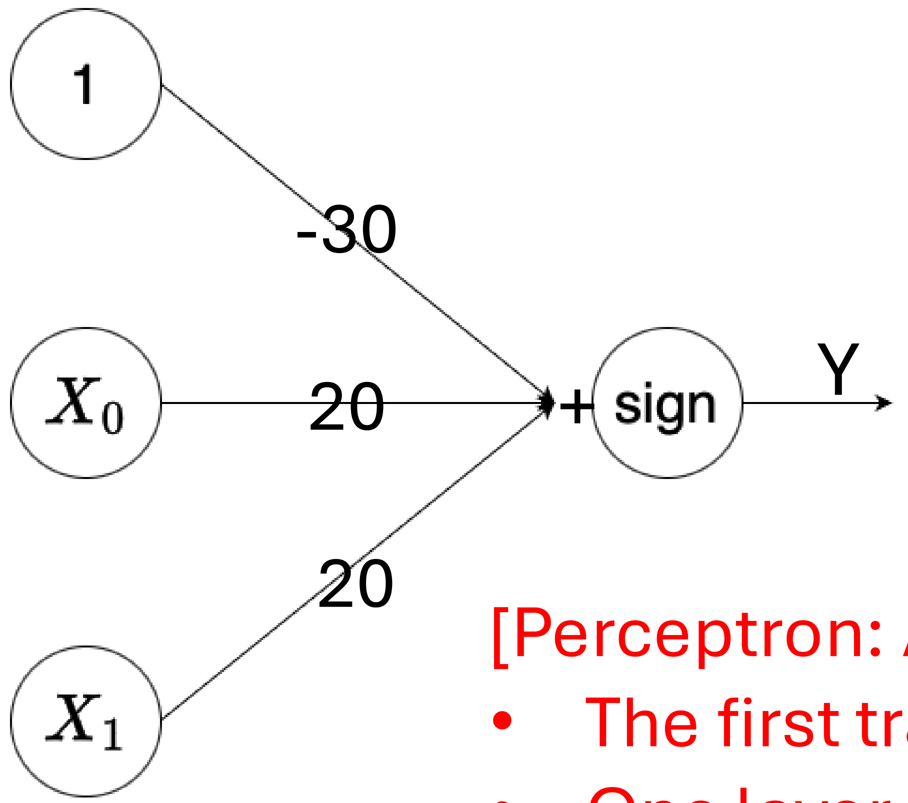
+ use the bound: $1-1/x \leq \ln(x)$]

$$2\epsilon^2 N^2 - 2N(2VC(H)+\ln 4/\delta)+VC(H)^2 \geq 0$$

Q: How many data points are needed to ensure that true error true error $(h) -$ train error (h) <0.01 $with\ the\ probability\ at\ least$ 0.9?

Deep **N**eural **N**etwork (DNN)

(**M**ulti**l**ayer **P**erceptron : MLP)



[Perceptron: A Founding Elements of Neural Net]

- The first trainable network
- One layer

Perceptron is a building block for the neural net but had not had attention since the work of Rosenblatt in spite of the convergence proof. The problem is that Perceptron has no way to represent the knowledge (feature) required for solving certain problems. It started to work right as having multiple-layers.

By the 1980s, however, researcher had developed algorithms for modifying neural nets' weights and threshold that were efficient enough for networks with more than one layer, removing many of the limitations addressed in the one layer perceptron.
(https://news.mit.edu/2017/explained-neural-networks-deep-learning-0414)

- **Neocognitron** (Fukushima 1980) : very similar to the architecture of modern deep neural net, alternating convolutional and down sampling layers. However, it did not use backpropagation.
- **LetNet Series** (Yann LeCun 1989 – 1998): Neocognitron like network and backpropagation is applied.

(https://www.rctn.org/bruno/public/papers/Fukushima1980.pdf)
(http://vision.stanford.edu/cs598_spring07/papers/Lecun98.pdf)

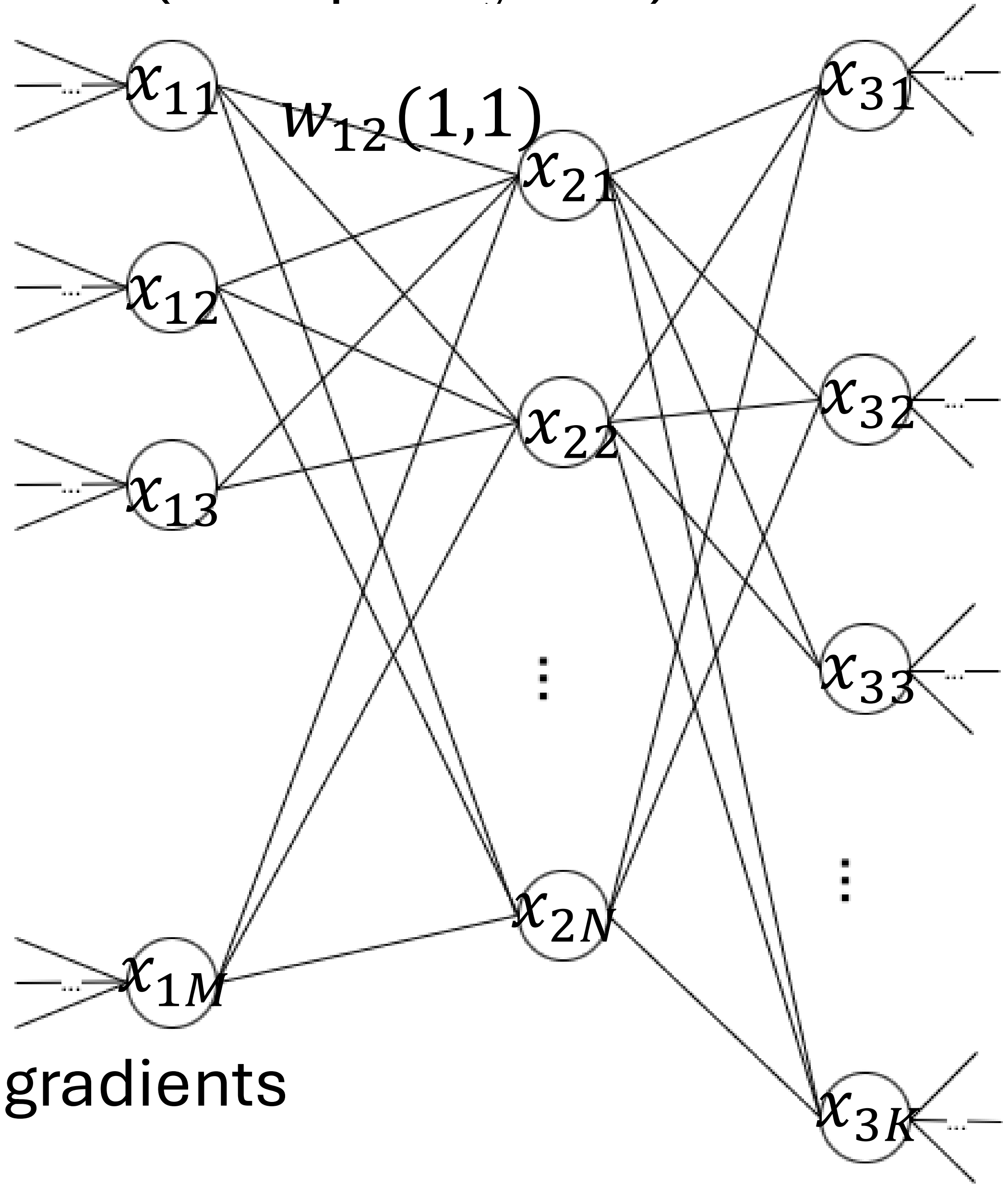
Feedfoward network defines a mapping/ function.

$$y = f(x; W)$$

The function is a heuristic to perform a task such as regression / classification.

The heuristic function is represented by composing together many different functions (multiple layered)

- x_{ij}: *i layer and j unit*
- $w_{kl}(i,j)$: weight from unit x_{ki} to unit x_{lj}
- depth : the number of overall layers
- width: the number of units at a layer
- Two directional data flow in a unit
  - Feedforward: for prediction
  - Error backpropagation: for computing gradients



The architecture of a deep neural net
defines the connectivity among the computational units.

- Input (Preprocessing and Convolution Layers)
- Middle Layer (Fully Connected Layers)
- Output (Right before computing objective / Cost / Loss function)