FSAN/ELEG815: Statistical Learning

Gonzalo R. Arce

Department of Electrical and Computer Engineering
University of Delaware

5. Training vs Testing

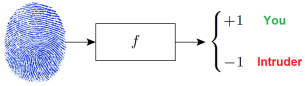
Outline of the Course

1. Review of Probability
2. Stationary processes
3. Eigen Analysis, Singular Value Decomposition (SVD) and Principal Component Analysis (PCA)
4. The Learning Problem
5. Training vs Testing
6. Estimation theory: Maximum likelihood and Bayes estimation
7. The Wiener Filter
8. Adaptive Optimization: Steepest descent and the LMS algorithm
9. Least Squares (LS) and Recursive Least Squares (RLS) algorithm
10. Overfitting and Regularization
11. Logistic, Ridge and Lasso regression.
12. Neural Networks
13. Matrix Completion

Review

- **Error measures:**

User specified $\mathsf{e}(h(\mathbf{x}), f(\mathbf{x}))$



In-sample:

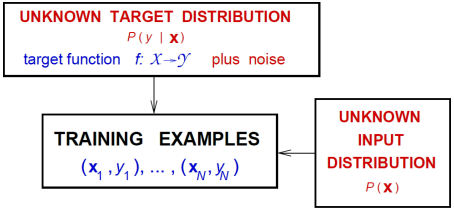
$$E_{in}(h) = \frac{1}{N}\sum_{n=1}^{N} \mathsf{e}(h(\mathbf{x}_n), f(\mathbf{x}_n))$$

Out-of-sample:

$$E_{out}(h) = \mathbb{E}_x[\mathsf{e}(h(\mathbf{x}), f(\mathbf{x}))]$$

- **Noisy targets:**

$$y = f(\mathbf{x}) \rightarrow y \sim P(y|\mathbf{x})$$



$(\mathbf{x}_1, y_1), \cdots, (\mathbf{x}_N, y_N)$ generated by

$$P(\mathbf{x}, y) = P(\mathbf{x})P(y|\mathbf{x})$$

$E_{out}(h)$ is now $\mathbb{E}_{x,y}[e(h(\mathbf{x}), y)]$

Outline

- From training to testing
- Illustrative examples
- Key notion: break point
- Puzzle

Example - The Final Exam

Before the final exam, a professor may hand out practice problems and solutions to the class (training set).

Why not to give out the exam problems?

The goal is for the students to learn the course material (small E_{out}), not to memorize the practice problems (small E_{in}).

Having memorized all the practice problems (small E_{in}) does not guarantee to learn the course material (small E_{out}).

The Final Exam

Testing:

- The hypothesis is fixed (you already prepare for the test).
- The hypothesis is tested over unseen data (the test does not include the same practice problems) i.e. E_{in} is computed using the hypothesis set.

$$\mathbb{P}[|E_{in} - E_{out}| > \epsilon] \leq 2e^{-2\epsilon^2 N}$$

- For a large N (number of questions), E_{in} tracks E_{out} (your performance gauges how well you learned).

The Final Exam

Training: Performance on practice problems.

- The hypothesis is adjusted (since you know the answers, you repeat a problem until getting it right).

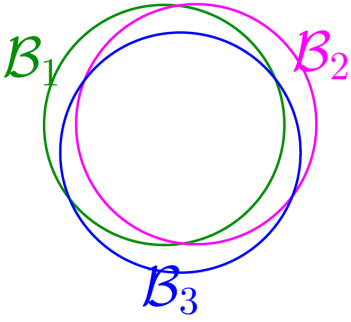
$$\mathbb{P}[|E_{in} - E_{out}| > \epsilon] \leq 2Me^{-2\epsilon^2 N}$$

- E_{in} is computed using the practice set.
- Small $E_{in} \rightarrow$ not necessarily small E_{out}.
  You may have not learned and have memorized the problems solutions.
- M is the number of hypotheses to explore.
  Depending on the times you repeat a problem, your performance may no longer accurately gauge how well you learned.

Goal: We want to replace M by another quantity that is not infinity.

Where did the M Come from?

Venn Diagram of $\mathcal{B}$ad events



The $\mathcal{B}$ad events $\mathcal{B}_m$ are

$$|E_{in}(h_m) - E_{out}(h_m)| > \epsilon$$

The union bound consider $\mathcal{B}_m$ as disjoint events:

$$\mathbb{P}[\mathcal{B}_1 \text{ or } \mathcal{B}_2 \text{ or } \cdots \text{ or } \mathcal{B}_M] \leq \mathbb{P}[\mathcal{B}_1] + \mathbb{P}[\mathcal{B}_2] + \cdots \mathbb{P}[\mathcal{B}_M]$$

It is a poor bound when there is overlap.

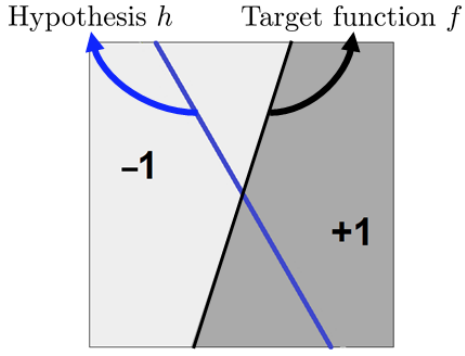
Can we Improve on M ?

Yes, bad events are very overlapping

Remember the perceptron:

$$h(\mathbf{x}) = \begin{cases} 1 & \text{if 'approved'} \\ -1 & \text{if 'deny credit'} \end{cases}$$

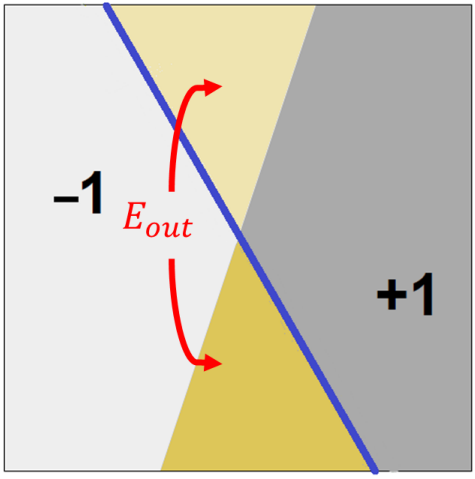
$$h(\mathbf{x}) = \text{sign}(\mathbf{w}^T\mathbf{x})$$

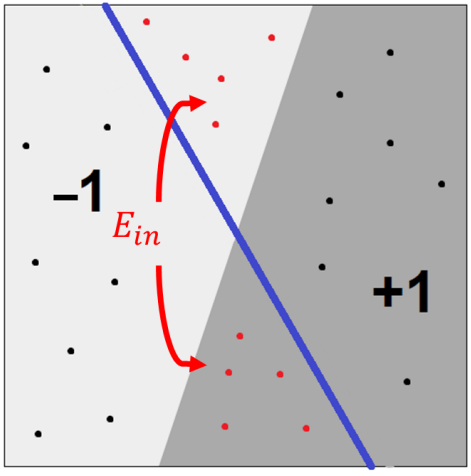


For any perceptron ($\mathbf{w}$): The line $w_0 + w_1x_1 + w_2x_2 = 0$ splits the plane into $+1$ and -1

Can we Improve on M ?

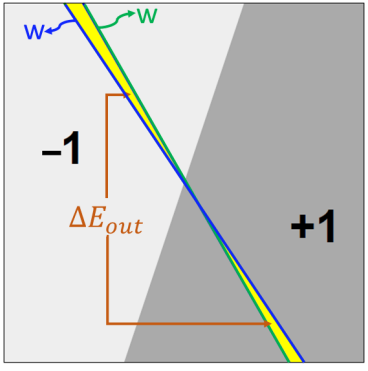
For the given perceptron (**w**) , consider the out-of-sample error E_{out} and the in-sample error E_{in}:

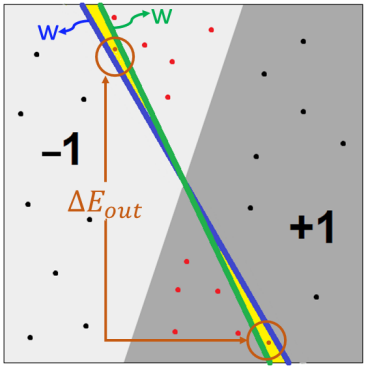




Can we Improve on M ?

Consider a different perceptron **w**:

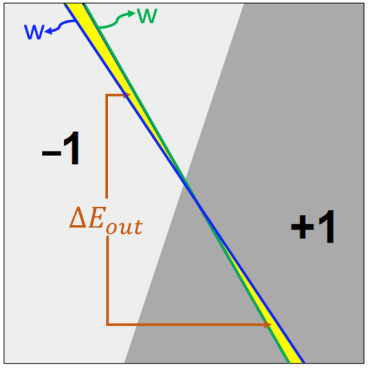


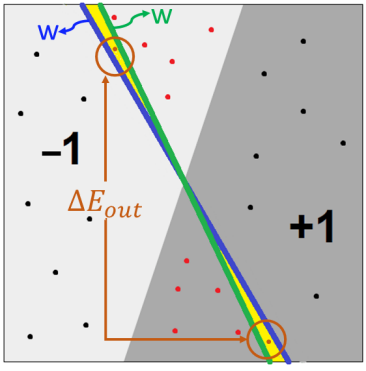


$\triangle E_{out}$ and $\triangle E_{in}$ move in the same direction

Area of yellow part increases $\rightarrow$ probability of data points falling in yellow part increases.

Can we Improve on M ?





$$|E_{in}(h_1) - E_{out}(h_1)| \approx |E_{in}(h_2) - E_{out}(h_2)| \qquad \text{(Both exceed } \epsilon\text{)}$$

Many hypotheses are similar. In PLA, if we slowly vary **w**, we get infinitely many hypotheses that differ from each other infinitesimally.

What can we Replace M with?

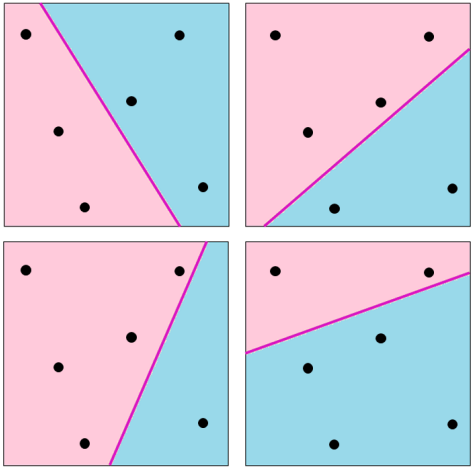
Since the input space $\mathcal{X}$ is infinity, the possible hypotheses are infinity.

Instead of counting the hypotheses over the whole input space, consider a finite set of input points.

On a finite set of input points, how many different 'hypotheses' can I get?

Classification by the four perceptrons is different in at least one data point, so we have four different 'hypotheses'.

Four different perceptrons:



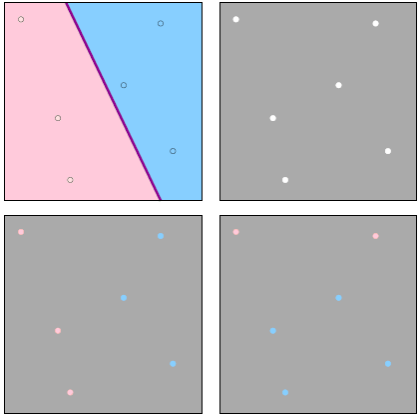
What can we Replace M with?

Define *dichotomy* as different 'hypotheses' over the finite set of N input points.

Definition: Let $\mathbf{x}_1, \cdots, \mathbf{x}_N \in \mathcal{X}$. The *dichotomies* generated by $\mathcal{H}$ are

$$\mathcal{H}(\mathbf{x}_1, \cdots, \mathbf{x}_N) = \{(h(\mathbf{x}_1), \cdots, h(\mathbf{x}_N)) | h \in \mathcal{H}\}$$

Hypotheses are seen through the eyes of N points only



Vary perceptron until the line crosses one of the points $\rightarrow$ different *dichotomy*.

Dichotomies: Mini-Hypotheses

A hypotheses $h : \mathcal{X} \to \{-1, +1\}$

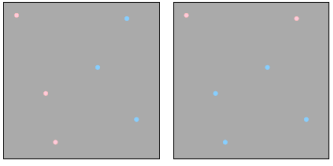
A dichotomy $h : \{\mathbf{x}_1, \mathbf{x}_2, , \cdots, \mathbf{x}_N, \} \to \{-1, +1\}$

Number of hypotheses $|\mathcal{H}|$ can be infinite.

Number of dichotomies $|\mathcal{H}(\mathbf{x}_1, \mathbf{x}_2, , \cdots, \mathbf{x}_N)|$ is at most 2^N

Candidate for replacing M.

Ex: The two *dichotomies* in the picture could be: $[-1, -1, -1, +1, +1, +1]$, $[-1, -1, +1, +1, +1, +1]$.



The Growth Function

The growth function counts the most dichotomies on any N points

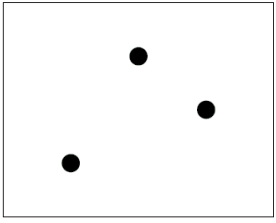
$$m_{\mathcal{H}}(N) = \max_{\mathbf{x}_1,\mathbf{x}_2,\cdots,\mathbf{x}_N \in \mathcal{X}} |\mathcal{H}(\mathbf{x}_1,\mathbf{x}_2,\cdots,\mathbf{x}_N)|$$

The value of $m_{\mathcal{H}}(N)$ is at most $|\{-1,+1\}^N|$. Hence, the growth function satisfies:

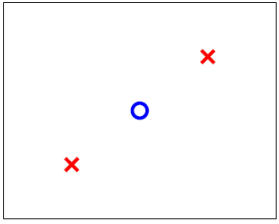
$$m_{\mathcal{H}}(N) \leq 2^N$$

Let's apply the definition.

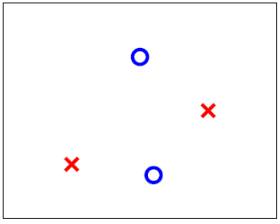
Applying $m_{\mathcal{H}}(N)$ Definition - 2D Perceptrons



Maximum 8 dichotomies with three points.



Dichotomy on 3 colinear points cannot be generated ($N = 4$)



Dichotomy here cannot be generated

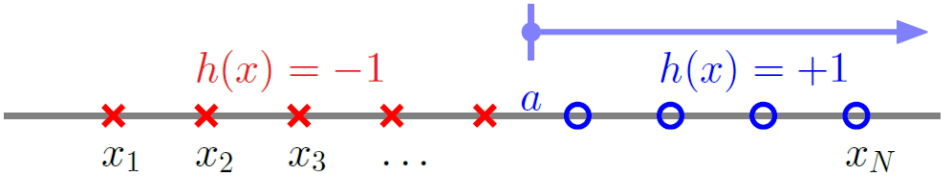
$$m_{\mathcal{H}}(3) = 8 \qquad m_{\mathcal{H}}(4) = 14$$

Note: At most 14 out of the possible 16 dichotomies on any 4 points can be generated.

Outline

- From training to testing
- Illustrative examples
  These examples confirm the intuition that $m_{\mathcal{H}}(N)$ grows faster when $\mathcal{H}$ becomes more complex.
- Key notion: break point
- Puzzle

Example 1: Positive Rays

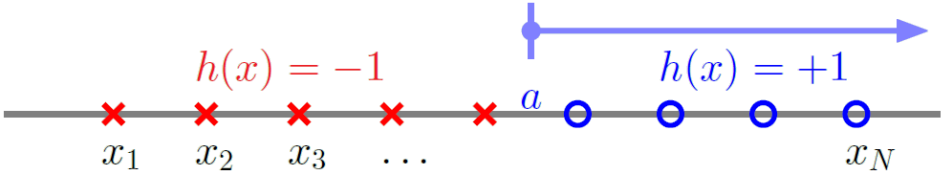


$\mathcal{H}$ is set of $h : \mathbb{R} \to \{-1, +1\}$

$$h(x) = \text{sign}(x - a)$$

Hypotheses are defined on a one-dimensional input space, and they return -1 to the left of a and $+1$ to the right of a.

Example 1: Positive Rays

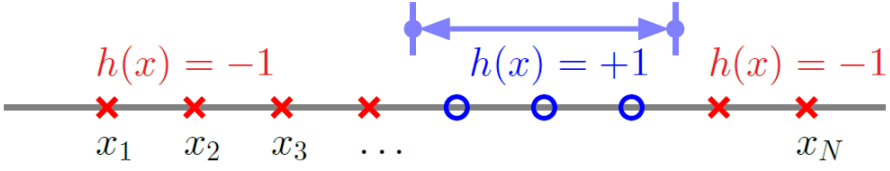


N points, split line into $N+1$ regions. As we vary a we get different dichotomies.

The growth function: $m_{\mathcal{H}}(N) = N+1$

At most $N+1$ dichotomies given any N points.

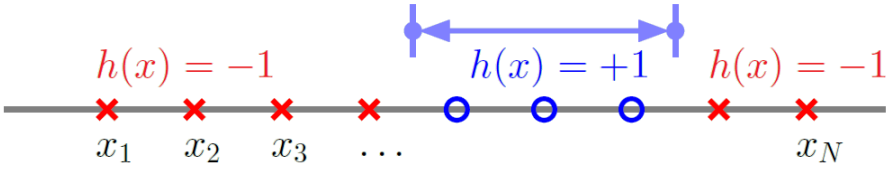
Example 2: Positive Intervals



$\mathcal{H}$ is set of $h : \mathbb{R} \to \{-1, +1\}$

Hypotheses defined on a one-dimensional input space, and they return $+1$ over some interval and -1 otherwise.

Example 2: Positive Intervals



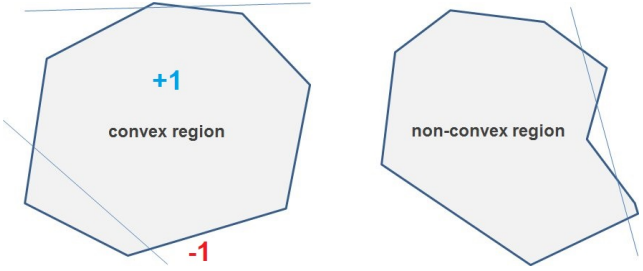
N points, split line into $N+1$ regions.

$$m_{\mathcal{H}}(N) = \binom{N+1}{2} + 1 = \frac{1}{2}N^2 + \frac{1}{2}N + 1$$

Dichotomies are decided by end values of interval, we have $\binom{N+1}{2}$ possibilities. Add the case in which both end values fall in the same region.

Example 3: Convex Sets

A set is **convex** if a line segment connecting any two points in the set lies entirely within the set

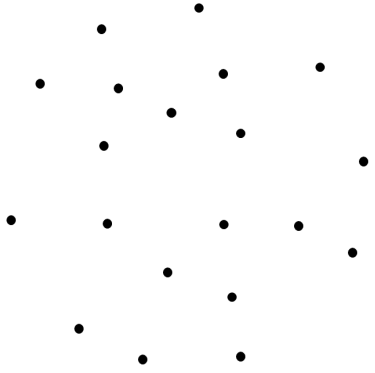


$\mathcal{H}$ consists of all hypotheses in two dimensions that are positive inside some convex set and negative elsewhere

$\mathcal{H}$ is set of $h : \mathbb{R}^2 \to \{-1, +1\}$ $\quad$ $h(\mathbf{x}) = +1$ is convex

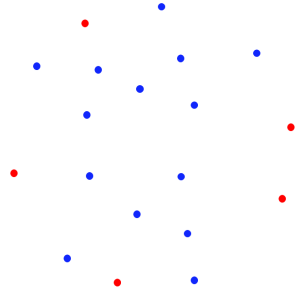
Example 3: Convex Sets

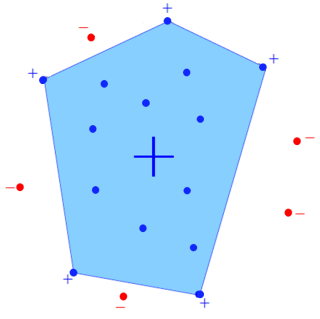
How many patterns can I get out of these data points using convex regions?



Example 3: Convex Sets

How many patterns can I get out of these data points using convex regions?

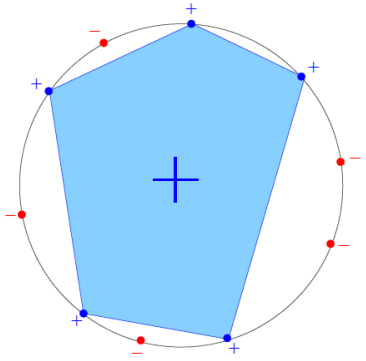




If we consider some outer points to be $+1$, then all interior points are $+1$ (not many dichotomies).

Example 3: Convex Sets

Find another distribution of points to get all possible dichotomies using convex regions?



Place N points over the perimeter of the circle. We get all possible combinations (maximum number of dichotomies).

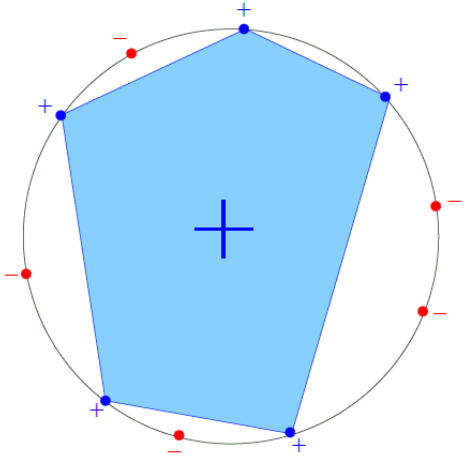
Example 3: Convex Sets

$$m_{\mathcal{H}}(N) = 2^N$$

Any dichotomy on these N points can be realized using a convex hypothesis.

The N points are 'shattered' by convex sets.

Note: $m_{\mathcal{H}}(N)$ is an upper bound. The number of possible dichotomies for given data points may be less than 2^N because of interior points.



The hypothesis shatters all points

The 3 Growth Functions

- $\mathcal{H}$ is positive rays:

$$m_{\mathcal{H}}(N) = N + 1$$

- $\mathcal{H}$ is positive intervals:

$$m_{\mathcal{H}}(N) = \frac{1}{2}N^2 + \frac{1}{2}N + 1$$

- $\mathcal{H}$ is convex sets:

$$m_{\mathcal{H}}(N) = 2^N$$

$m_{\mathcal{H}}(N)$ grows faster when $\mathcal{H}$ becomes more complex.

Back to the Big Picture

Remember this inequality?

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

What happens if $m_{\mathcal{H}}(N)$ replaces M?

$m_{\mathcal{H}}(N)$ polynomial $\implies$ Good

If $m_{\mathcal{H}}(N)$ can be bounded by any polynomial, the generalization error will go to zero as $N \to \infty$ $\implies$ Learning is feasible.

Just prove that $m_{\mathcal{H}}(N)$ can be bounded by a polynomial?

Outline

- From training to testing
- Illustrative examples
- Key notion: break point
  It would enable us to proof that $m_{\mathcal{H}}(N)$ can be bounded by a polynomial
- Puzzle

Break Point of $\mathcal{H}$

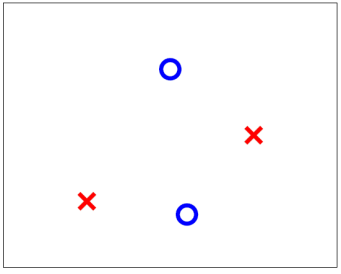
Definition:
If data set of size k cannot be shattered by $\mathcal{H}$, then k is a break point for $\mathcal{H}$

$$m_{\mathcal{H}}(k) < 2^k$$

The break point k is the number of data points at which we fail to get all possible dichotomies.

A bigger data set cannot be shattered either.

Remember the 2D perceptrons



At most 14 out of 16 dichotomies on any 4 points can be generated.

$$k = 4$$

Break Point - the 3 Examples

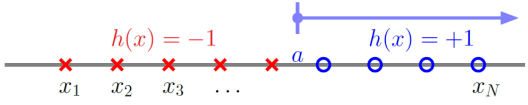
$$m_{\mathcal{H}}(k) < 2^k$$

- Positive rays $m_{\mathcal{H}}(N) = N + 1$

$k = 1$ $m_{\mathcal{H}}(1) = 2 \not< 2^1$

$k = 2$ $m_{\mathcal{H}}(2) = 3 < 2^2$ $\rightarrow$ **break point**

Intuitively, remember the positive rays:



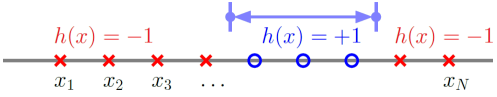
There is no way for the positive ray to generate: ● ●

Break Point - the 3 Examples

- Positive intervals $m_{\mathcal{H}}(N) = \frac{1}{2}N^2 + \frac{1}{2}N + 1$

$k = 1$ $m_{\mathcal{H}}(1) = 2 \not< 2^1$
$k = 2$ $m_{\mathcal{H}}(2) = 4 \not< 2^2$
$k = 3$ $m_{\mathcal{H}}(3) = 7 < 2^3$ $\rightarrow$ **break point**

Intuitively, remember the positive intervals:



There is no way to generate:

- Convex sets $m_{\mathcal{H}}(N) = 2^N$

break point $k = \text{`}\infty\text{'}$

Main Result

We observe how the break point increases with the complexity of the model.

No break point $\rightarrow \quad m_{\mathcal{H}}(N) = 2^N$

Any break point $\rightarrow$ Use k to bound $quadm_{\mathcal{H}}(N)$ by a polynomial in N

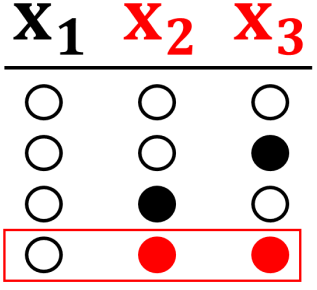
Remember: If $m_{\mathcal{H}}(N)$ can be bounded by any polynomial, the generalization error will go to zero as $N \rightarrow \infty \implies$ Learning is feasible.

To consider learning feasible, all that we need to know now is that there exist a break point.

Puzzle

Let's consider 3 data points and a break point $k = 2$, i.e. we cannot get 4 dichotomies out of any pair of points. How many dichotomies can we get on these 3 data points?

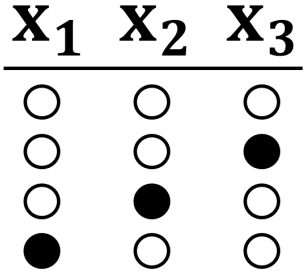
We start generating the possible dichotomies.



We stop when we get all possible combinations out of two points.
We cannot include this last dichotomy!

Puzzle

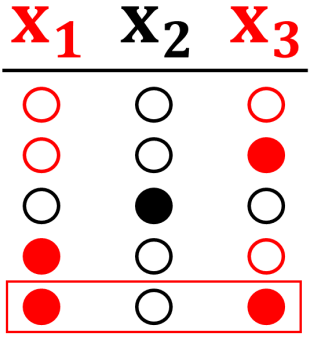
We tried another one:



We can add this one!

Puzzle

Let's continue!



We stop again when we get all possible combinations out of two points.
We cannot include this last dichotomy either!

Puzzle

If we continue trying, we'll see that none of the other dichotomies work.

x_1	x_2	x_3
○	○	○
○	○	●
○	●	○
●	○	○

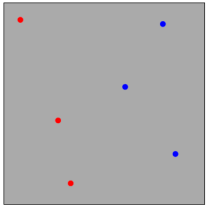
At most 4 dichotomies out of 8.

If we start different, are we going to be able to achieve more? No!

Note: Knowing only N and k, we can determine the maximum number of dichotomies (complexity).

Review

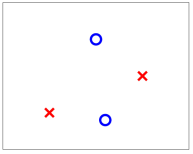
- **Dichotomies:**



- **Growth Function:**

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

- **Break Point** k :



At most 14 out of the possible 16 dichotomies on any 4 points can be generated. $k = 4$

- **Maximum # of dichotomies**

$\mathbf{x_1}$	$\mathbf{x_2}$	$\mathbf{x_3}$
○	○	○
○	○	●
○	●	○
●	○	○

Bounding the Growth Function

For a given $\mathcal{H}$, if the break point k is fixed, $m_{\mathcal{H}}(N)$ can be bounded by a polynomial$^{(*)}$:

Theorem:
If $m_{\mathcal{H}}(k) < 2^k$ for some value k, then

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

for all N. The RHS is polynomial of degree $k-1$.

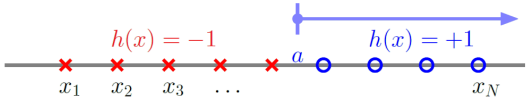
Note: This ensures good generalization on the Hoeffding's Inequality.

$^{(*)}$ Proof can be found on the book: Learning from Data, Yaser S. Abu-Mostafa, Malik Magdon-Ismail and Hsuan-Tien Lin, AMLbook 2012.

Three examples

Let's take the hypothesis sets for which we compute the growth function:

- $\mathcal{H}$ is positive rays:



We compute before:

$$m_{\mathcal{H}}(N) = N + 1$$

No need to know anything about the hypothesis set just that break point $k = 2$

$$m_{\mathcal{H}}(N) \leq \sum_{i=0}^{1} \binom{N}{i} = N + 1$$

Three examples

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

- $\mathcal{H}$ is positive intervals: (break point $k = 3$)

$$m_{\mathcal{H}}(N) = \frac{1}{2}N^2 + \frac{1}{2}N + 1 \quad \leq \quad \sum_{i=0}^{2} \binom{N}{i} = \frac{1}{2}N^2 + \frac{1}{2}N + 1$$

- $\mathcal{H}$ is 2D perceptrons: (break point $k = 4$)

$$m_{\mathcal{H}}(N) = \quad ? \quad \leq \quad \sum_{i=0}^{3} \binom{N}{i} = \frac{1}{6}N^3 + \frac{5}{6}N + 1$$

What we Want

Instead of:

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

We want:

$$\mathbb{P}[|E_{in}(g) - E_{out}(g)| > \epsilon] \leq 2 \quad m_{\mathcal{H}}(N) \quad e^{-2\epsilon^2 N}$$

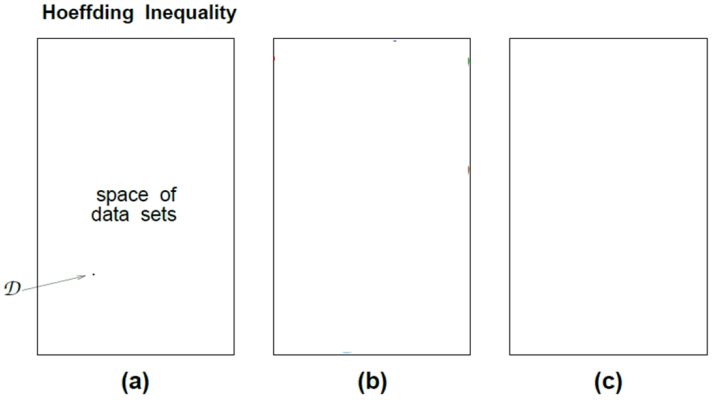
Let's consider a pictorial proof:

Pictorial Proof

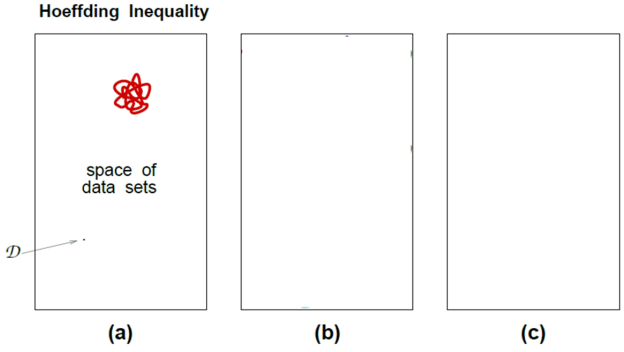
- How does $m_{\mathcal{H}}(N)$ relate to overlaps?
- What to do about E_{out}?
- Putting it together

How does $m_{\mathcal{H}}(N)$ relate to overlaps?

The 'canvas' represents space of all possible data sets, with areas corresponding to probabilities. Each data set $\mathcal{D}$ is a point on the canvas. The total area of the canvas is 1.



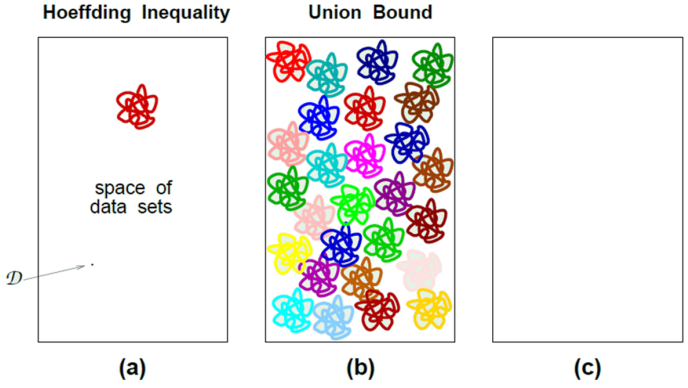
How does $m_{\mathcal{H}}(N)$ relate to overlaps?



(a) For a given hypothesis $h \in \mathcal{H}$, colored points correspond to data sets where E_{in} does not generalize well to E_{out} ("$|E_{in}(h) - E_{out}(h)| > \epsilon$").

The Hoeffding Inequality guarantees a small colored area.

How does $m_{\mathcal{H}}(N)$ relate to overlaps?



(b) Considering different hypothesis.
The event "$|E_{in}(h) - E_{out}(h)| > \epsilon$" may contain different points (painted with different color).
The union bound assumes no overlap, colored area is large.

How does $m_{\mathcal{H}}(N)$ relate to overlaps?

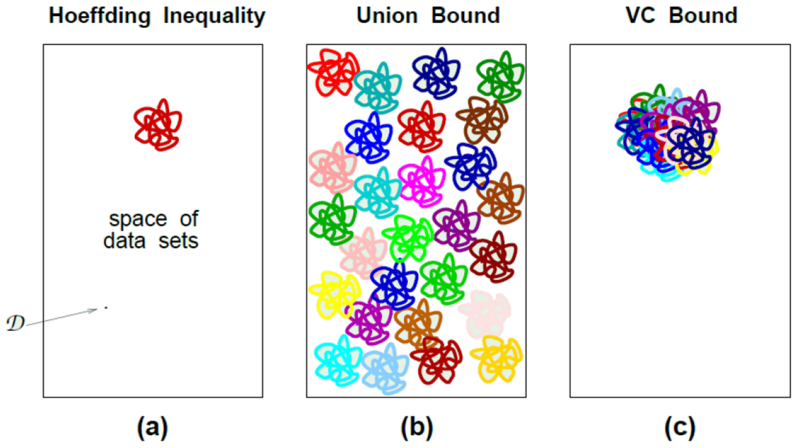
How the growth function is going to account for the overlaps?

Assume a hypothesis set $\mathcal{H}$ that colors each point on the canvas 100 times (because of 100 different h's). The total colored area is now $\frac{1}{100}$ of what it would have been without any overlap.

Many hypotheses have same dichotomy on a given $\mathcal{D}$.

If a hypothesis paints a given point, similar hypotheses (same dichotomy) will do too.

How does $m_{\mathcal{H}}(N)$ relate to overlaps?



(c) The VC bound keeps track of overlaps.
It estimates the total area of bad generalization to be relatively small.

Learning is Feasible!

Pictorial Proof

- How does $m_{\mathcal{H}}(N)$ relate to overlaps?

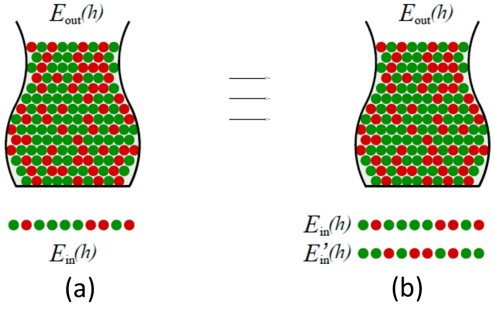
  The point being colored (event "$|E_{in}(h) - E_{out}(h)| > \epsilon$") depends not only on $\mathcal{D}$, but also on the entire $\mathcal{X}$ because $E_{out}(h)$ is based on $\mathcal{X}$.

- What to do about E_{out}?

- Putting it together

What to do about E_{out}

To remedy this, consider the artificial event "$|E_{in}(h) - E'_{in}(h)| > \epsilon$" instead, where E_{in} and E'_{in} are based on two samples $\mathcal{D}$ and $\mathcal{D}'$ each of size N.



(a) For multiple hypotheses, $E_{in}(h)$ may sometimes deviate from $E_{out}(h)$.

(b) $E_{in}(h)$ and $E'_{in}(h)$ track $E_{out}(h)$. Thus, they track each other. For multiple hypotheses the behavior reflects the **same** as in (a), $E_{in}(h)$ may sometimes deviate from $E'_{in}(h)$.

Putting it Together

Instead of:

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

We wanted:

$$\mathbb{P}[|E_{in}(g) - E_{out}(g)| > \epsilon] \leq 2 \quad m_{\mathcal{H}}(N) \quad e^{-2\epsilon^2 N}$$

but rather, we get:

$$\mathbb{P}[|E_{in}(g) - E_{out}(g)| > \epsilon] \leq 4 \quad m_{\mathcal{H}}(2N) \quad e^{-\frac{1}{8}\epsilon^2 N}$$

The Vapnik-Chervonenkis Inequality

Outline

- The definition
- VC dimension of perceptrons
- Interpreting the VC dimension
- Generalization bounds

Definition of VC Dimension

The Vapnik-Chervonenkis (VC) dimension of a hypothesis set $\mathcal{H}$ denoted by $d_{\text{VC}}(\mathcal{H})$, is

the largest value of N for which $m_{\mathcal{H}}(N) = 2^N$

" the maximum number of points $\mathcal{H}$ can shatter"

$N \leq d_{\text{VC}}(\mathcal{H}) \implies \mathcal{H}$ can shatter N points

$k > d_{\text{VC}}(\mathcal{H}) \implies k$ is a break point for $\mathcal{H}$

The Growth Function

In terms of a break point k:

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

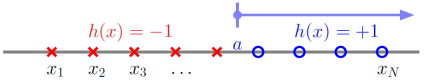
In terms of the d_{VC}:

$$m_{\mathcal{H}}(N) \leq \sum_{i=0}^{d_{\text{VC}}} \binom{N}{i}$$

Maximum power is $N^{d_{\text{VC}}}$

Examples

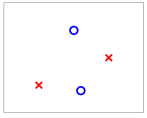
- $\mathcal{H}$ is positive rays: $d_{\text{VC}} = 1$



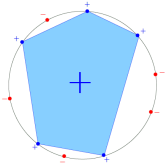
if $N = 2$, we cannot have

- $\mathcal{H}$ is 2D perceptrons: $d_{\text{VC}} = 3$

if $N = 4$, we cannot have



- $\mathcal{H}$ is convex sets: $d_{\text{VC}} = \infty$

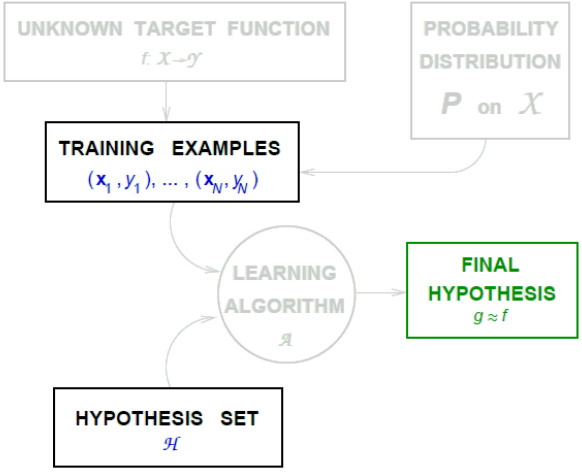


VC Dimension and Learning

Result: If $d_{\text{VC}}(\mathcal{H})$ is finite, $g \in \mathcal{H}$ will generalize.

This statement is true independently of:

- **Learning algorithm**
- **Input distribution**
- **Target function**

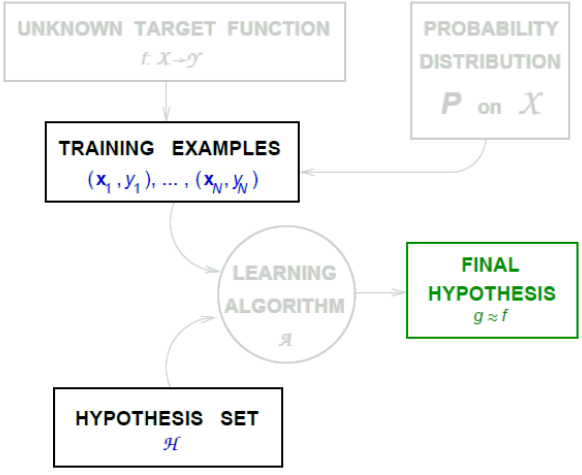


VC Dimension and Learning

Result: If $d_{\text{VC}}(\mathcal{H})$ is finite, $g \in \mathcal{H}$ will generalize.

This statement depends on:

- **Final hypothesis**
- **Hypothesis set**
  VC dimension depends only on the hypothesis set.
- **Training samples**
  Exist a small chance of having a data set that won't allow generalization.



VC Dimension of Perceptrons

Consider the 2D perceptron:

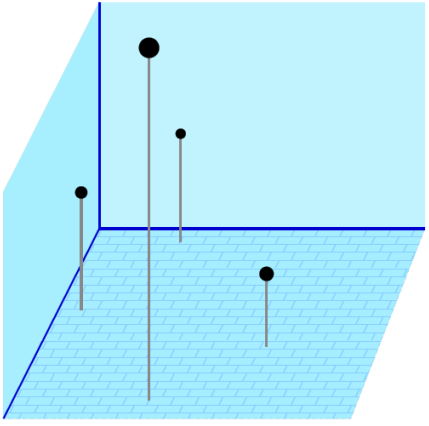
$d = 2$, $d_{VC} = 3$

In general, for a d-dimensional perceptron:

$$d_{VC} = d + 1$$

To prove this, we are going to show that:

$$\begin{aligned} d_{VC} &\leq d+1 \\ d_{VC} &\geq d+1 \end{aligned}$$



VC Dimension of Perceptrons

Consider a set of $N = d+1$ points in $\mathbb{R}^d$ shattered by the perceptron:

Let's choose input points such as:

$$\mathbf{X} = \begin{bmatrix} 1 & x_{11} & x_{12} & \dots & x_{1d} \\ 1 & x_{21} & x_{22} & \dots & x_{2d} \\ 1 & x_{31} & x_{32} & \dots & x_{3d} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_{N1} & x_{N2} & \dots & x_{Nd} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & \dots & 0 \\ 1 & 1 & 0 & \dots & 0 \\ 1 & 0 & 1 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & 0 & \dots & 0 & 1 \end{bmatrix}$$

- $\mathbf{X} \in \mathbb{R}^{(d+1)\times(d+1)}$
- $\mathbf{X}$ is invertible $(\det(\mathbf{X}) = 1)$.

This would allow us to shatter the data set.

Can we Shatter this Data Set?

In vector form, dichotomies are:

$$\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_{d+1} \end{bmatrix} = \begin{bmatrix} \pm 1 \\ \pm 1 \\ \vdots \\ \pm 1 \end{bmatrix}, \quad \text{and considering the perceptron: } \mathbf{y} = \text{sign}(\mathbf{X}\mathbf{w})$$

Since $\mathbf{X}$ is invertible, for any $\mathbf{y}$, we can find a vector $\mathbf{w}$ satisfying:

$$\begin{aligned} \text{sign}(\mathbf{X}\mathbf{w}) &= \mathbf{y} \\ \mathbf{X}\mathbf{w} &= \mathbf{y} \\ \mathbf{w} &= \mathbf{X}^{-1}\mathbf{y} \end{aligned}$$

Note: There exist a perceptron $\mathbf{w}$ that can generate all possible dichotomies $\mathbf{y}$.

Quiz

This result implies what?

(a) $d_{\text{VC}} = d + 1$

(b) $d_{\text{VC}} \geq d + 1$

(c) $d_{\text{VC}} \leq d + 1$

(d) No conclusion

Quiz

This result implies what?

(a) $d_{\text{VC}} = d + 1$

(b) $d_{\text{VC}} \geq d + 1$

(c) $d_{\text{VC}} \leq d + 1$

(d) No conclusion

Answer: (b) $d_{\text{VC}} \geq d + 1$

Quiz

Now, to demonstrate that $d_{\text{VC}} \leq d+1$, we need to show that:

(a) There are $d+1$ points we cannot shatter

(b) There are $d+2$ points we cannot shatter

(c) We cannot shatter any set of $d+1$ points

(d) We cannot shatter any set of $d+2$ points

Quiz

Now, to demonstrate that $d_{\text{VC}} \leq d+1$, we need to show that:

(a) There are $d+1$ points we cannot shatter

(b) There are $d+2$ points we cannot shatter

(c) We cannot shatter any set of $d+1$ points

(d) We cannot shatter any set of $d+2$ points

Answer: (d) We cannot shatter any set of $d+2$ points

For any $d+2$ points,

$$\mathbf{x}_1, \cdots, \mathbf{x}_{d+1}, \mathbf{x}_{d+2}$$

More points than dimensions ($\mathbf{x} \in \mathbb{R}^d$) $\implies$ the vectors must be linearly dependent and

$$\mathbf{x}_j = \sum_{i \neq j} a_i \mathbf{x}_i$$

where not all the a_i's are zeros.

$$\mathbf{x}_j = \sum_{i \neq j} a_i \mathbf{x}_i$$

Focus on $\mathbf{x}_i$'s with non-zero a_i and construct a dichotomy that cannot be implemented by a perceptron:

$\mathbf{x}_i$'s with non-zero a_i get $y_i = \text{sign}(a_i)$, $\mathbf{x}_j$ gets $y_j = -1$ and let others either $+1$ or -1.

No perceptron can implement such dichotomy!

Why?

The perceptron:

$$\mathbf{x}_j = \sum_{i \neq j} a_i \mathbf{x}_i \implies \mathbf{w}^T \mathbf{x}_j = \sum_{i \neq j} a_i \mathbf{w}^T \mathbf{x}_i$$

If $y_i = \text{sign}(\mathbf{w}^T \mathbf{x}_i) = \text{sign}(a_i)$, then $a_i \mathbf{w}^T \mathbf{x}_i > 0$

This forces

$$\mathbf{w}^T \mathbf{x}_j = \sum_{i \neq j} a_i \mathbf{w}^T \mathbf{x}_i > 0$$

Therefore, $y_j = \text{sign}(w^T \mathbf{x}_j) = +1$ (impossible to get -1).

Conclusion: we cannot shatter any set of $d+2$ points $\implies d_{\text{VC}} \leq d+1$

Putting it Together

We proved $d_{VC} \leq d+1$ and $d_{VC} \geq d+1$. Thus,

$$d_{VC} = d+1$$

What is $d+1$ in the perceptron?

It is the number of parameters $w_0, w_1, ..., w_d$,

Note: The more parameters a model has, the more diverse its hypothesis set is, which is reflected in a larger value of the growth function.

Outline

- The definition
- VC dimension of perceptrons
- Interpreting the VC dimension
  - What does it signify?
  - How apply it in practice?
- Generalization bounds

Degrees of Freedom

Parameters create degrees of freedom

# of parameters: **analog** degrees of freedom

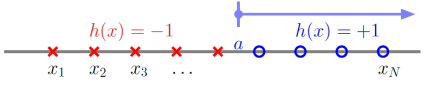
d_{VC}: translates to degrees of freedom.

Parameters are consider as knobs

The Usual Suspects

Let's see if the correspondence between degrees of freedom and VC dimension holds.

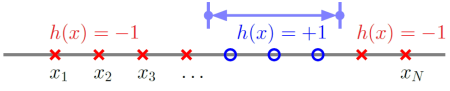
- Positive rays ($d_{\text{VC}} = 1$):



we cannot have

Each hypothesis is specified by the parameter a (one degree of freedom).

- Positive Intervals ($d_{\text{VC}} = 2$)



we cannot have

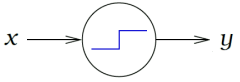
Each hypothesis is specified by the two end values of the interval (two degrees of freedom).

Not Just Parameters

Parameters may not contribute degrees of freedom:

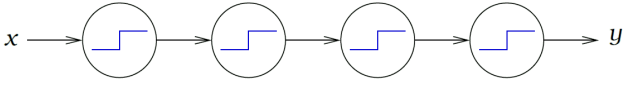
Example: consider a one-dimensional perceptron $h(x) = \text{sign}(w_0 + w_1 x)$ where w_0 is a threshold.

$$y = h(x) = \begin{cases} 1 & \text{if } w_1 x > -w_0 \\ -1 & \text{if } w_1 x < -w_0 \end{cases}$$



2 parameters and 2 degrees of freedom.

Creating a cascade of perceptrons:



Eight parameters in this model and still two degrees of freedom.

d_{VC} measures the **effective** number of parameters.

Number of Data Points Needed

Two small quantities in the VC inequality:

$$\mathbb{P}[|E_{in}(g) - E_{out}(g)| > \epsilon] \leq \underbrace{4m_{\mathcal{H}}(2N)e^{-\frac{1}{8}\epsilon^2 N}}_{\delta}$$

If we want certain ϵ and δ, how does N depend on d_{VC}

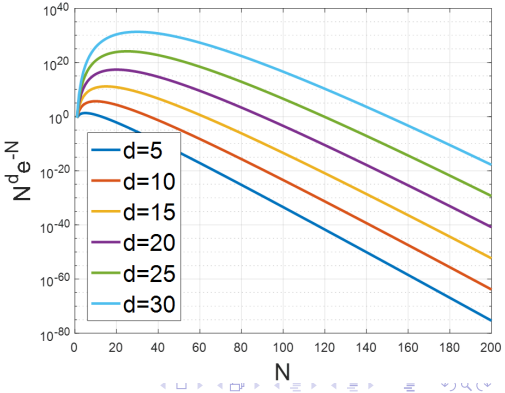
Let us look at $N^d e^{-N}$

Fix $N^d e^{-N}$ = small value

How does N change with d?
It is basically proportional.

Rule of thumb:

$$N \geq 10 d_{\text{VC}}$$



Outline

- The definition
- VC dimension of perceptrons
- Interpreting the VC dimension
- Generalization bounds

Rearranging Things

Start from the VC inequality:

$$\mathbb{P}[|E_{in}(g) - E_{out}(g)| > \epsilon] \leq \underbrace{4m_{\mathcal{H}}(2N)e^{-\frac{1}{8}\epsilon^2 N}}_{\delta}$$

The performance is specified by these two parameters:

- ϵ determines the allowed generalization error
- δ determines how often the error tolerance is violated (confidence).

Get ϵ in terms of δ:

$$\delta = 4m_{\mathcal{H}}(2N)e^{-\frac{1}{8}\epsilon^2 N} \implies \epsilon = \underbrace{\sqrt{\frac{8}{N}\ln\frac{4m_{\mathcal{H}}(2N)}{\delta}}}_{\Omega}$$

With probability $\geq 1 - \delta$, $|E_{out} - E_{in}| \leq \Omega(N, \mathcal{H}, \delta)$

Generalization Bound

With probability $\geq 1-\delta$, $|E_{out} - E_{in}| \leq \Omega(N, \mathcal{H}, \delta)$

Since we minimize E_{in}, in general, $E_{in} \leq E_{out}$. Thus,

$$\text{With probability } \geq 1-\delta, \quad E_{out} - E_{in} \leq \Omega$$

$$\Longrightarrow$$

$$\text{With probability } \geq 1-\delta, \quad E_{out} \leq E_{in} + \Omega$$

We know and we have control over the RHS quantities.

Tradeoff: bigger hypothesis set is good for $\downarrow E_{in}$ but bad for generalization $\uparrow \Omega$.

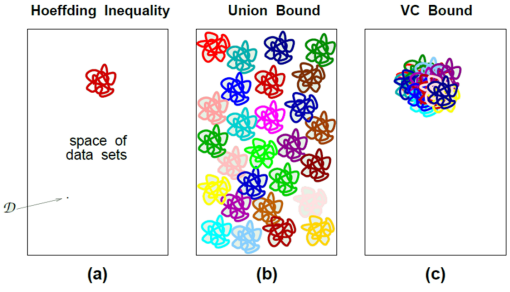
Review

- $m_{\mathcal{H}}(N)$ is polynomial if $\mathcal{H}$ has a break point k

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

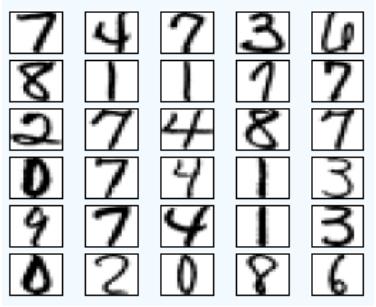
Maximum power is N^{k-1}

- The VC Inequality:



$$\begin{array}{lcccc} \mathbb{P}[|E_{in}(g) - E_{out}(g)| > \epsilon] & \leq & 2 & M & e^{-2\epsilon^2 N} \\ & & \downarrow & \downarrow & \downarrow \\ \mathbb{P}[|E_{in}(g) - E_{out}(g)| > \epsilon] & \leq & 4 & m_{\mathcal{H}}(2N) & e^{-\frac{1}{8}\epsilon^2 N} \end{array}$$

A real data set



16x16 pixels gray-scale images of digits from the US Postal Service Zip Code Database. Goal: recognize the digit in each image.

Not a trivial task (even for a human). Typical human error E_{out} is 2.5% due to common confusions between $\{4, 9\}$ and $\{2, 7\}$.

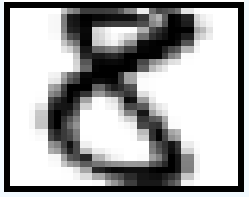
Machine Learning tries to achieve or beat this error.

Input Representation

Since the images are 16×16 pixels:

- 'raw' input
  $\mathbf{x}_r = (x_0, x_1, x_2, \cdots, x_{256})$
- Linear model:
  $(w_0, w_1, w_2, \cdots, w_{256})$

Too many many parameters.
A better representation needed.



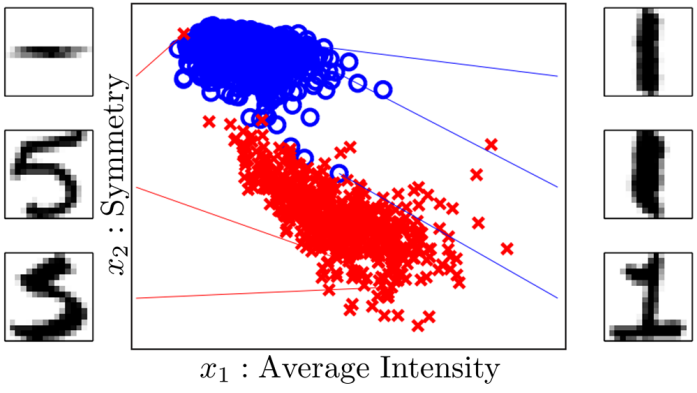
Features: Extract useful information, e.g.,

- Average intensity and symmetry
  $\mathbf{x} = (x_0, x_1, x_2)$
- Linear model: (w_0, w_1, w_2)

The descriptors must be representative of the data.

Illustration of Features

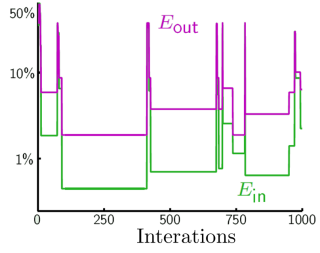
$$\mathbf{x} = (x_0, x_1, x_2) \quad x_0 = 1$$



Almost linearly separable. However, it is impossible to have them all right.

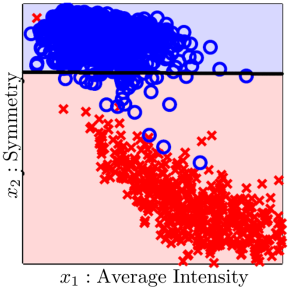
What Perceptron Learning Algorithm does?

Evolution of in-sample error E_{in} and out-of-sample error E_{out} as a function of iterations of PLA



- Assume we know E_{out} .
- E_{in} tracks E_{out}. PLA generalizes well!

- It would never converge (data not linearly separable).
- **Stopping criteria:** Max. number of iterations.



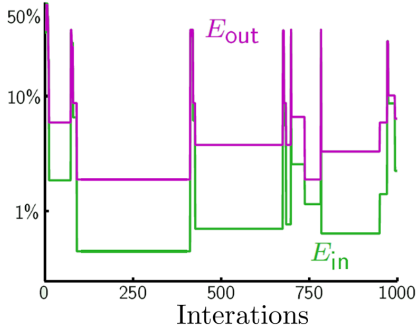
Final perceptron boundary

We can do better...

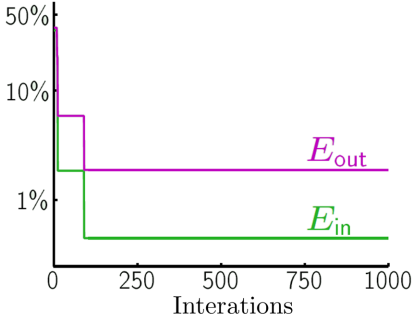
The 'pocket' algorithm

Keeps 'in its pocket' the best weight vector encountered up to the current iteration t in PLA.

PLA

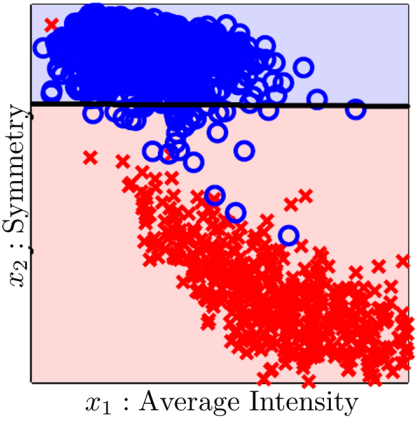


Pocket

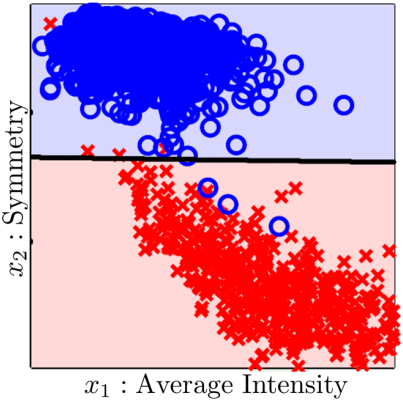


Classification boundary - PLA versus Poket

PLA



Pocket



Linear Regression - Credit Example

Regression $\equiv$ Real-valued output

Classification: Credit approval (yes/no)
Regression: Credit line (dollar amount)

Input: $\mathbf{x} =$

age	23 years
gender	male
annual salary	\$30,000
years in residence	1 year
years in job	1 year
current debt	\$15,000
...	...

Linear regression output: $h(\mathbf{x}) = \sum_{i=0}^{d} w_i x_i = \mathbf{w}^T \mathbf{x}$

Credit Example Again - The data set

Input: $\mathbf{x} =$

age	23 years
gender	male
annual salary	$30,000
years in residence	1 year
years in job	1 year
current debt	$15,000
...	...

Output:

$$h(\mathbf{x}) = \sum_{i=0}^{d} w_i x_i = \mathbf{w}^T \mathbf{x}$$

Credit officers decide on credit lines:

$$(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \cdots, (\mathbf{x}_N, y_N)$$

$y_n \in \mathbb{R}$ is the credit for customer $\mathbf{x}_n$.

Linear regression wants to automate this task, trying to replicate human experts decisions.

How to Measure the Error?

How well does $h(\mathbf{x}) = \mathbf{w}^T\mathbf{x}$ approximate $y = f(\mathbf{x})$?

Linear regression algorithm is based on minimizing the squared error:

$$E_{out}(h) = \mathbb{E}[(h(\mathbf{x}) - y)^2]$$

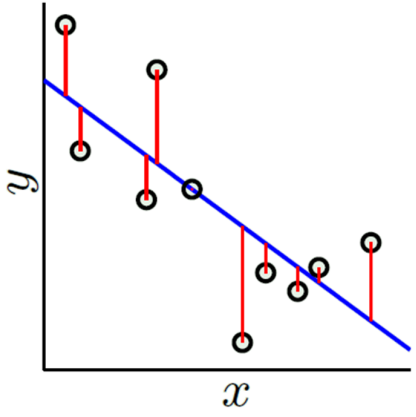
where $\mathbb{E}[\cdot]$ is taken with respect to $P(\mathbf{x}, y)$ that is unknown. Thus, we resort to minimize the in-sample error:

$$E_{in}(h) = \frac{1}{N}\sum_{n=1}^{N}(h(\mathbf{x}_n) - y_n)^2$$

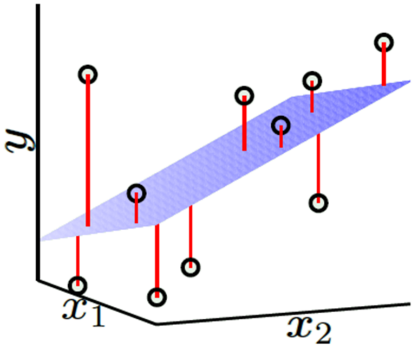
The goal is to find a hypothesis that achieves a small E_{in}.

Illustration of Linear Regression

The solution hypothesis (in blue) of the linear regression algorithm in one and two dimensions input. The sum of square error is minimized.



One dimension (line)



Two dimensions (hyperplane)

Linear Regression - The Expression for E_{in}

$$\mathbf{y} = w_0 + w_1\mathbf{x}_1 + w_2\mathbf{x}_2 + \ldots + w_p\mathbf{x}_d + \epsilon.$$

$$\underbrace{\begin{bmatrix} y_1 \\ \vdots \\ y_N \end{bmatrix}}_{\mathbf{y}} = \underbrace{\begin{bmatrix} 1 & x_{11} & x_{12} & \cdots & x_{1d} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_{N1} & x_{N2} & \cdots & x_{Nd} \end{bmatrix}}_{\mathbf{X}} \cdot \underbrace{\begin{bmatrix} w_0 \\ w_1 \\ \vdots \\ w_d \end{bmatrix}}_{\mathbf{w}} + \begin{bmatrix} \epsilon \\ \vdots \\ \epsilon \end{bmatrix}$$

$$\begin{aligned} E_{in} &= \frac{1}{N}\sum_{n=1}^{N}(\mathbf{w}^T\mathbf{x}_n - y_n)^2 \qquad \mathbf{X} \in \mathbb{R}^{N\times(d+1)} \\ &= \frac{1}{N}||\mathbf{X}\mathbf{w} - \mathbf{y}||_2^2 = \frac{1}{N}(\mathbf{X}\mathbf{w} - \mathbf{y})^T(\mathbf{X}\mathbf{w} - \mathbf{y}) \\ &= \frac{1}{N}(\mathbf{w}^T\mathbf{X}^T\mathbf{X}\mathbf{w} - \mathbf{y}^T\mathbf{X}\mathbf{w} - \mathbf{w}^T\mathbf{X}^T\mathbf{y} + \mathbf{y}^T\mathbf{y}) \\ &= \frac{1}{N}(\mathbf{w}^T\mathbf{X}^T\mathbf{X}\mathbf{w} - 2\mathbf{w}^T\mathbf{X}^T\mathbf{y} + \mathbf{y}^T\mathbf{y}) \end{aligned}$$

Learning Algorithm - Minimizing E_{in}

$$\begin{aligned}\hat{\mathbf{w}} &= \arg\min_{\mathbf{w}\in\mathbb{R}^d} \frac{1}{N}||\mathbf{X}\mathbf{w}-\mathbf{y}||_2^2 \\ &= \arg\min_{\mathbf{w}\in\mathbb{R}^d} \frac{1}{N}(\mathbf{w}^T\mathbf{X}^T\mathbf{X}\mathbf{w}-2\mathbf{w}^T\mathbf{X}^T\mathbf{y}+\mathbf{y}^T\mathbf{y})\end{aligned}$$

Observation: The error is a quadratic function of $\mathbf{w}$

Consequences: The error is an d–dimensional bowl–shaped function of $\mathbf{w}$ with a unique minimum

Result: The optimal weight vector, $\hat{\mathbf{w}}$, is determined by differentiating $E_{in}(\mathbf{w})$ and setting the result to zero

$$\nabla_{\mathbf{w}} E_{in}(\mathbf{w}) = 0$$

- A closed form solution exists

Example

Consider a two dimensional case, i.e., a $d = 2$. Plot the error surface and error contours.

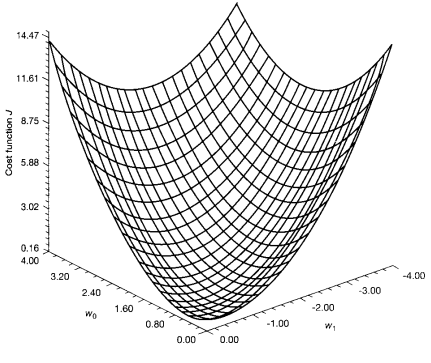


Figure 5.6 Error-performance surface of the two-tap transversal filter described in the numerical example.

Error Surface

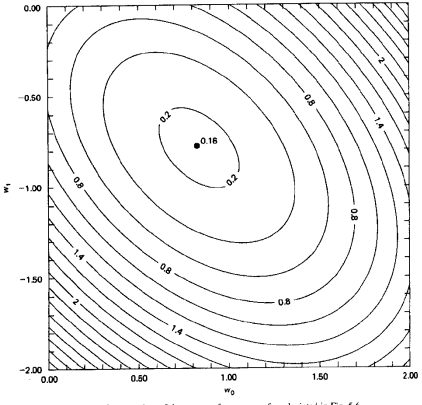


Figure 5.7 Contour plots of the error-performance surface depicted in Fig. 5.6.

Error Contours

Aside (Matrix Differentiation):

Let $\mathbf{w} \in \mathbb{R}^{(d+1)}$ and let $f : \mathbb{R}^{(d+1)} \to \mathbb{R}$. The derivative of f (called gradient of f) with respect to $\mathbf{w}$ is:

$$\nabla_{\mathbf{w}}(f) = \frac{\partial f}{\partial \mathbf{w}} = \begin{bmatrix} \nabla_0(f) \\ \nabla_1(f) \\ \vdots \\ \nabla_d(f) \end{bmatrix} = \begin{bmatrix} \frac{\partial f}{\partial w_0} \\ \frac{\partial f}{\partial w_1} \\ \vdots \\ \frac{\partial f}{\partial w_d} \end{bmatrix}$$

Thus,

$$\nabla_k(f) = \frac{\partial f}{\partial w_k}, \qquad k = 0, 1, \cdots, d$$

Example

Now suppose $f = \mathbf{c}^T\mathbf{w}$. Find $\nabla_{\mathbf{w}}(f)$

In this case,

$$f = \mathbf{c}^T\mathbf{w} = \sum_{k=0}^{d} w_k c_k$$

and

$$\nabla_k(f) = \frac{\partial f}{\partial w_k} = c_k, \qquad k = 0, 1, \cdots, d$$

Result: For $f = \mathbf{c}^T\mathbf{w}$

$$\nabla_{\mathbf{w}}(g) = \begin{bmatrix} \nabla_0(g) \\ \nabla_1(g) \\ \vdots \\ \nabla_{\mathrm{d}}(g) \end{bmatrix} = \begin{bmatrix} c_0 \\ c_1 \\ \vdots \\ c_d \end{bmatrix} = \mathbf{c}$$

Same for $f = \mathbf{w}^T\mathbf{c}$.

Example

Lastly, suppose $f = \mathbf{w}^T\mathbf{Q}\mathbf{w}$. Where $\mathbf{Q} \in \mathbb{R}^{(d+1)\times(d+1)}$ and $\mathbf{w} \in \mathbb{R}^{d+1}$. Find $\nabla_{\mathbf{w}}(f)$

In this case, using the product rule:

$$\begin{aligned}\nabla_{\mathbf{w}} f &= \frac{\partial \mathbf{w}^T(\mathbf{Q}\bar{\mathbf{w}})}{\partial \mathbf{w}} + \frac{\partial(\bar{\mathbf{w}}^T\mathbf{Q})\mathbf{w}}{\partial \mathbf{w}} \\ &= \frac{\partial \mathbf{w}^T\mathbf{u}_1}{\partial \mathbf{w}} + \frac{\partial \mathbf{u}_2^T\mathbf{w}}{\partial \mathbf{w}}\end{aligned}$$

Using previous result, $\frac{\partial \mathbf{c}^T\mathbf{w}}{\partial \mathbf{w}} = \frac{\partial \mathbf{w}^T\mathbf{c}}{\partial \mathbf{w}} = \mathbf{c}$,

$$\begin{aligned}\nabla_{\mathbf{w}} f &= \mathbf{u}_1 + \mathbf{u}_2, \\ &= \mathbf{Q}\mathbf{w} + \mathbf{Q}^T\mathbf{w} = (\mathbf{Q} + \mathbf{Q}^T)\mathbf{w}, \qquad \text{if } \mathbf{Q} \text{ symmetric, } \mathbf{Q}^T = \mathbf{Q} \\ &= 2\mathbf{Q}\mathbf{w}\end{aligned}$$

Returning to the MSE performance criteria

$$E_{in}(\mathbf{w}) = \left[\frac{1}{N}(\mathbf{w}^T\mathbf{X}^T\mathbf{X}\mathbf{w} - 2\mathbf{w}^T\mathbf{X}^T\mathbf{y} + \mathbf{y}^T\mathbf{y})\right]$$

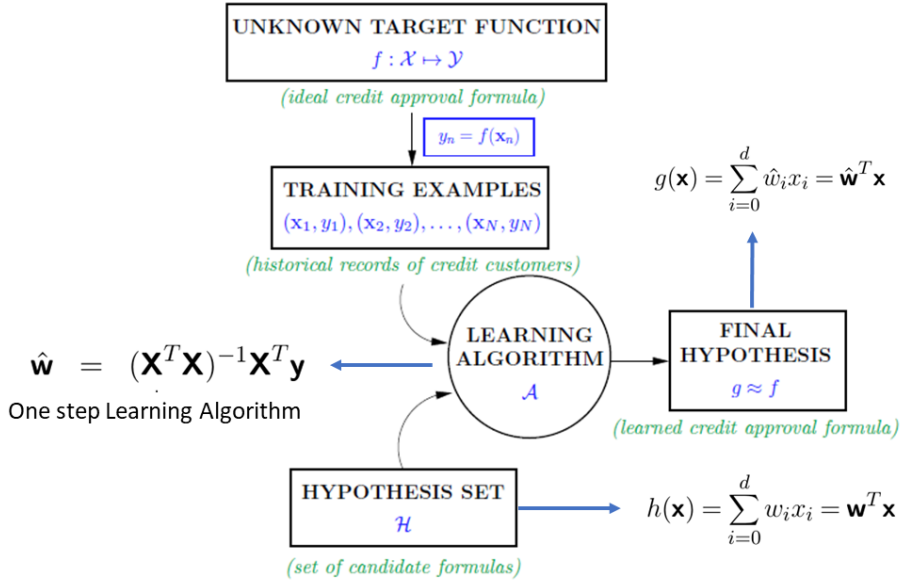
Differentiating with respect to $\mathbf{w}$ and setting equal to zero, we obtain,

$$\begin{aligned} \bigtriangledown E_{in}(\mathbf{w}) &= \frac{1}{N}(2\mathbf{X}^T\mathbf{X}\mathbf{w} - 2\mathbf{X}^T\mathbf{y} + 0) \\ &= \frac{2}{N}\mathbf{X}^T\mathbf{X}\mathbf{w} - \frac{2}{N}\mathbf{X}^T\mathbf{y} = 0 \end{aligned}$$

$$\begin{aligned} \mathbf{X}^T\mathbf{X}\mathbf{w} &= \mathbf{X}^T\mathbf{y} \\ \hat{\mathbf{w}} &= (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y} \\ &= \mathbf{X}^\dagger\mathbf{y} \end{aligned}$$

where $\mathbf{X}^\dagger = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T$ is the pseudo-inverse of $\mathbf{X}$.

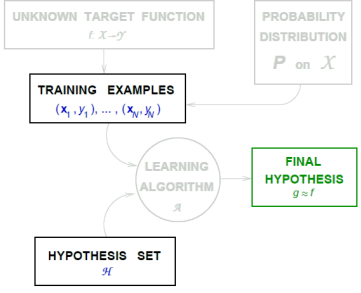
Learning Diagram - Linear Regression



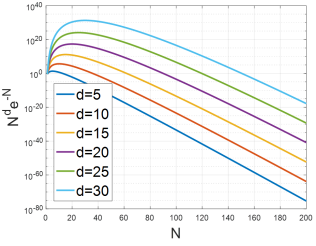
Review

- **VC dimension** $d_{\text{VC}}(\mathcal{H})$
  most points $\mathcal{H}$ can shatter.

- **Scope of VC analysis**



- **Utility of VC dimension:**



$$N \propto d_{\text{VC}}$$

Rule of thumb: $N \geq 10 d_{\text{VC}}$

- **Generalization bound**

$$E_{out} \leq E_{in} + \Omega$$

Approximation- Generalization Tradeoff

Balance between approximating f in the training data and generalizing on new data.

Goal: small $E_{out} \rightarrow$ good approximation of f out of sample.

More complex $\mathcal{H} \implies$ better chance of **approximating** f

Less complex $\mathcal{H} \implies$ better chance of **generalizing** out of sample

A more complex $\mathcal{H}$ better approximates f, however, it might be more difficult for the algorithm to zoom in on the right hypothesis.

The ideal $\mathcal{H}$ is a singleton hypothesis set containing only the target function.

$$\mathcal{H} = \{f\} \equiv \text{Wining the lottery!}$$

Approximation-Generalization Tradeoff

Two different approaches:

- VC analysis (binary error): $E_{out} \leq E_{in} + \Omega$.
  - $E_{in} \rightarrow$ Approximation
  - $\Omega \rightarrow$ Generalization

  The optimal model is a compromise that minimizes a combination of the two terms.

- Bias-variance analysis (squared error): decomposing E_{out} into
  1. How well $\mathcal{H}$ can approximate f
  2. How well we can zoom in on a good $h \in \mathcal{H}$

  We apply this analysis to **real-valued targets** and use **squared error** (linear regression).

Start with E_{out}

$$E_{out}(g^{(\mathcal{D})}) = \mathbb{E}_{\mathbf{x}}[(g^{(\mathcal{D})}(\mathbf{x}) - f(\mathbf{x}))^2]$$

where $\mathbb{E}_{\mathbf{x}}$ denotes the expected value with respect to $\mathbf{x}$ (based on P on $\mathcal{X}$).

Rid of the dependence on a particular data set by taking the expectation with respect to all data sets:

$$\begin{aligned}\mathbb{E}_{\mathcal{D}}\left[E_{out}(g^{(\mathcal{D})})\right] &= \mathbb{E}_{\mathcal{D}}\left[\mathbb{E}_{\mathbf{x}}[(g^{(\mathcal{D})}(\mathbf{x}) - f(\mathbf{x}))^2]\right] \\ &= \mathbb{E}_{\mathbf{x}}\left[\mathbb{E}_{\mathcal{D}}[(g^{(\mathcal{D})}(\mathbf{x}) - f(\mathbf{x}))^2]\right]\end{aligned}$$

Now, let us focus on:

$$\mathbb{E}_{\mathcal{D}}[(g^{(\mathcal{D})}(\mathbf{x}) - f(\mathbf{x}))^2]$$

The Average Hypothesis

To evaluate $\mathbb{E}_{\mathcal{D}}[(g^{(\mathcal{D})}(\mathbf{x}) - f(\mathbf{x}))^2]$:

We define the 'average' hypothesis $\bar{g}(\mathbf{x})$:

$$\bar{g}(\mathbf{x}) = \mathbb{E}_{\mathcal{D}}[g^{(\mathcal{D})}(\mathbf{x})]$$

Imagine we generate **many** data sets $\mathcal{D}_1, \mathcal{D}_1, \cdots, \mathcal{D}_K$. We can estimate an average function for any $\mathbf{x}$ by

$$\bar{g}(\mathbf{x}) \approx \frac{1}{K} \sum_{k=1}^{K} g^{(\mathcal{D}_k)}(\mathbf{x})$$

$g(\mathbf{x})$ is seen as a RV, with the randomness coming from the randomness in the data set.

For a particular $\mathbf{x}$, $\bar{g}(\mathbf{x})$ is the expectation of this RV.

Using $\bar{g}(\mathbf{x})$

$$\begin{aligned}\mathbb{E}_{\mathcal{D}}[(g^{(\mathcal{D})}(\mathbf{x}) - f(\mathbf{x}))^2] &= \mathbb{E}_{\mathcal{D}}[(g^{(\mathcal{D})}(\mathbf{x}) - \bar{g}(\mathbf{x}) + \bar{g}(\mathbf{x}) - f(\mathbf{x}))^2] \\ &= \mathbb{E}_{\mathcal{D}}[(g^{(\mathcal{D})}(\mathbf{x}) - \bar{g}(\mathbf{x}))^2 + (\bar{g}(\mathbf{x}) - f(\mathbf{x}))^2 \\ &\quad +2(g^{(\mathcal{D})}(\mathbf{x}) - \bar{g}(\mathbf{x}))(\bar{g}(\mathbf{x}) - f(\mathbf{x}))]\end{aligned}$$

Since $\mathbb{E}_{\mathcal{D}}[g^{(\mathcal{D})}(\mathbf{x})] = \bar{g}(\mathbf{x})$, cross term cancels.

$$= \mathbb{E}_{\mathcal{D}}[(g^{(\mathcal{D})}(\mathbf{x}) - \bar{g}(\mathbf{x}))^2] + (\bar{g}(\mathbf{x}) - f(\mathbf{x}))^2$$

Bias and Variance

$$\mathbb{E}_{\mathcal{D}}[(g^{(\mathcal{D})}(\mathbf{x}) - f(\mathbf{x}))^2] = \underbrace{\mathbb{E}_{\mathcal{D}}[(g^{(\mathcal{D})}(\mathbf{x}) - \bar{g}(\mathbf{x}))^2]}_{\text{var}(\mathbf{x})} + \underbrace{(\bar{g}(\mathbf{x}) - f(\mathbf{x}))^2}_{\text{bias}(\mathbf{x})}$$

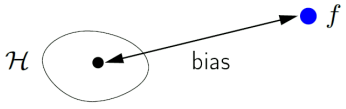
var($\mathbf{x}$) is the variance of the RV $g^{(\mathcal{D})}(\mathbf{x})$ and measures the variation in the final hypothesis depending on the data set.

bias($\mathbf{x}$) measures how much the average function that we would learn using different data sets $\mathcal{D}$ deviates from the target function.

Therefore,

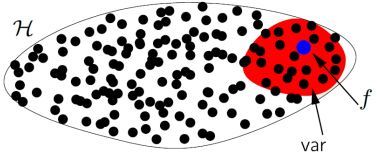
$$\begin{aligned}
\mathbb{E}_{\mathcal{D}}\left[E_{out}(g^{(\mathcal{D})})\right] &= \mathbb{E}_{\mathbf{x}}\left[\mathbb{E}_{\mathcal{D}}[(g^{(\mathcal{D})}(\mathbf{x}) - f(\mathbf{x}))^2]\right] \\
&= \mathbb{E}_{\mathbf{x}}\left[\text{bias}(\mathbf{x}) + \text{var}(\mathbf{x})\right] \\
&= \textbf{bias} + \textbf{var}
\end{aligned}$$

$$\textbf{bias} = \mathbb{E}_{\mathbf{x}}\left[(\bar{g}(\mathbf{x}) - f(\mathbf{x}))^2\right]$$



Very small model (one hypothesis). The final hypothesis $g^{(\mathcal{D})}$ will be the same as $\bar{g}$, for any data set $\rightarrow$ **var** $= 0$. The **bias** will depend solely on how well this single hypothesis approximates the target f, and unless we are extremely lucky, we expect a large **bias**.

$$\textbf{var} = \mathbb{E}_{\mathbf{x}}\left[\mathbb{E}_{\mathcal{D}}[(g^{(\mathcal{D})}(\mathbf{x}) - \bar{g}(\mathbf{x}))^2]\right]$$



Very large model (all hypothesis). $f \in \mathcal{H}$. Different data sets will lead to different hypotheses that agree with f on the data set, and are spread around f in the red region. Thus, **bias**≈ 0 because $\bar{g}$ is likely to be close to f. The **var** is large (represented by the size of the red region in the figure).

The Tradeoff:

$\mathcal{H}\uparrow$ **bias**$\downarrow$ **var**$\uparrow$

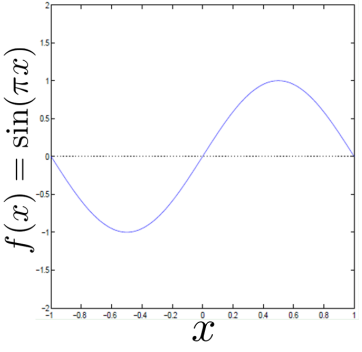
Example: Sine Target

$f : [-1,1] \to \mathbb{R} \quad f(x) = \sin(\pi x)$ **unknown**

We sample x uniformly in $[-1,1]$ to generate two training samples ($N = 2$)

Two models used for learning:

$$\mathcal{H}_0 : \quad h(x) = b$$
$$\mathcal{H}_1 : \quad h(x) = ax + b$$

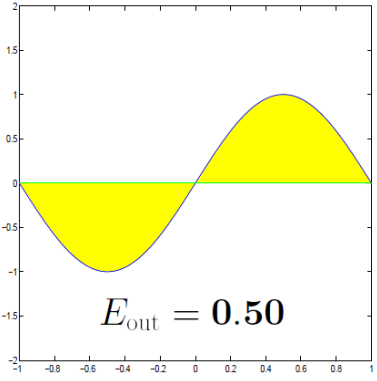


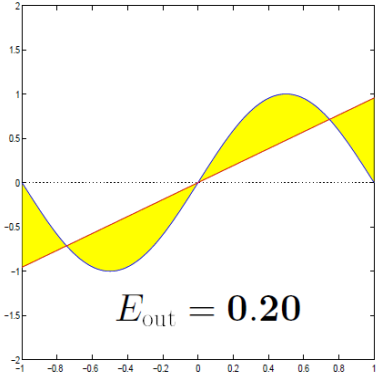
Which is better, $\mathcal{H}_0$ or $\mathcal{H}_1$?

Approximation - $\mathcal{H}_0$ versus $\mathcal{H}_1$

Based on the two models and assuming we know f, try to find the two functions that minimize the squared error:

$$\mathcal{H}_0 : h(x) = b \qquad \mathcal{H}_1 : h(x) = ax + b$$



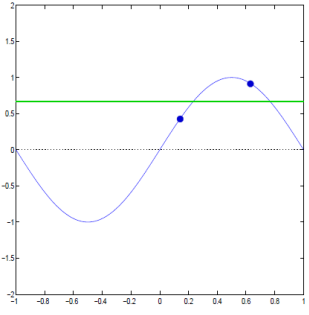


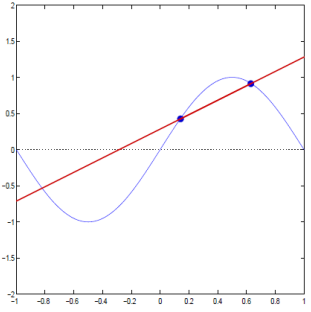
Learning - $\mathcal{H}_0$ versus $\mathcal{H}_1$

In learning, we do not know f. We use the two examples (x_1, y_1),(x_2, y_2) to learn the two functions that best fits the data.

$\mathcal{H}_0$: midpoint $\left(b = \frac{y_1+y2}{2}\right)$

$\mathcal{H}_1$: line passes through the two points

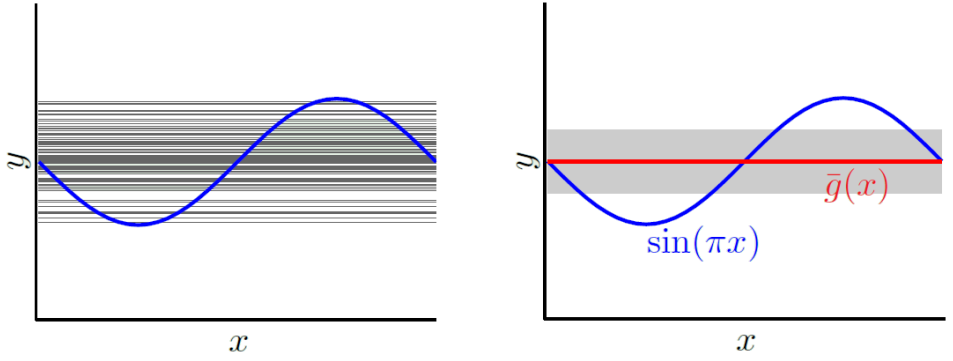




The result varies depending on the data points. We need bias-variance analysis to evaluate our result (considering other possible data sets).

Bias and Variance - $\mathcal{H}_0$

Repeating the process with many data sets, we can estimate the bias and the variance.



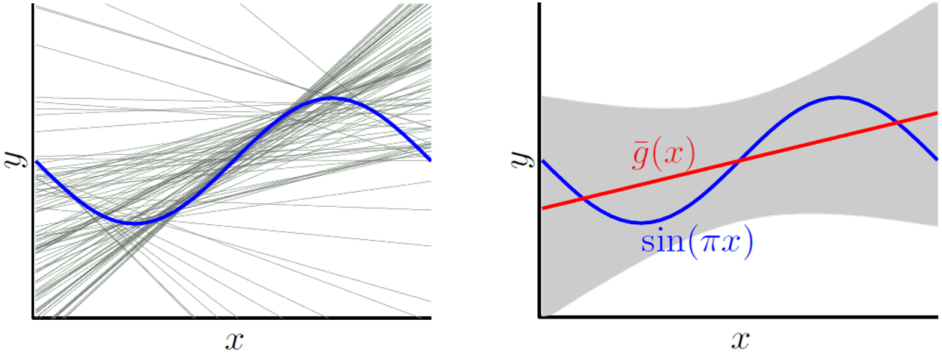
Average hypothesis $\bar{g}(x)$. In this case $\bar{g}(x) \approx 0$ that is close to the best approximation computed using f.

bias: difference between red function $\bar{g}(x)$ and blue function f.

var(x) is indicated by the gray shaded region that is $\bar{g}(x) \pm \sqrt{\textbf{var}(x)}$

Bias and Variance - $\mathcal{H}_1$

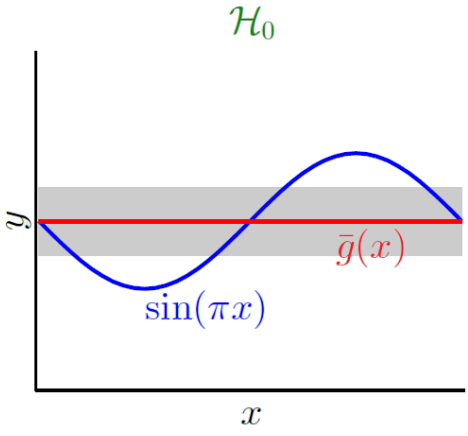
Using the same data sets as before, for the second model we get

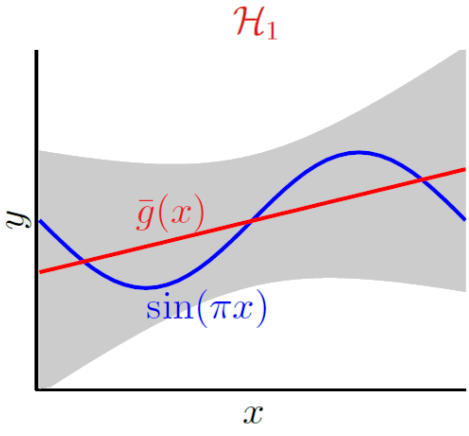


bias: difference between red function $\bar{g}(x)$ and blue function f.

var(x) is indicated by the gray shaded region that is $\bar{g}(x) \pm \sqrt{\textbf{var}(x)}$

The Winner is ...





bias = 0.50 var=0.25 bias=0.21 var=1.69

The simpler model wins by significantly decreasing the **var** at the expense of a smaller increase in **bias**

Lesson Learned

However, the **var** term decreases as N increases, so if we get a bigger data set, the **bias** term will be dominant in E_{out}, and $\mathcal{H}_1$ will win.

Match the '**model complexity**'

to the **data resources**, not to the **target complexity**

Outline

- Bias and Variance
- Learning Curves

Expected E_{out} and E_{in}

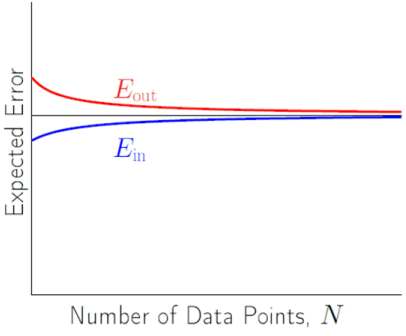
Consider learning with a data set $\mathcal{D}$ of size N,

the final hypothesis has a expected out-of-sample error $\mathbb{E}_{\mathcal{D}}\left[E_{out}(g^{(\mathcal{D})})\right]$ and

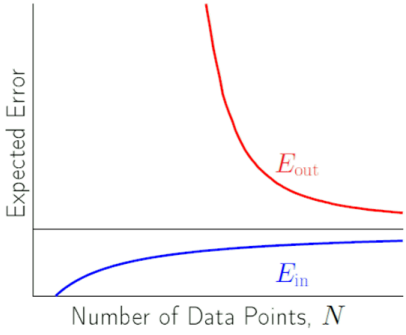
expected in-sample error $\mathbb{E}_{\mathcal{D}}\left[E_{in}(g^{(\mathcal{D})})\right]$

How do they vary with N?

The Curves



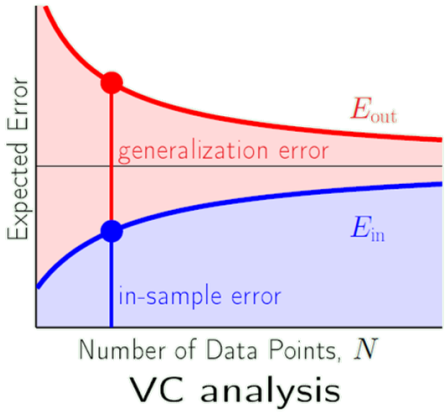
Simple Model

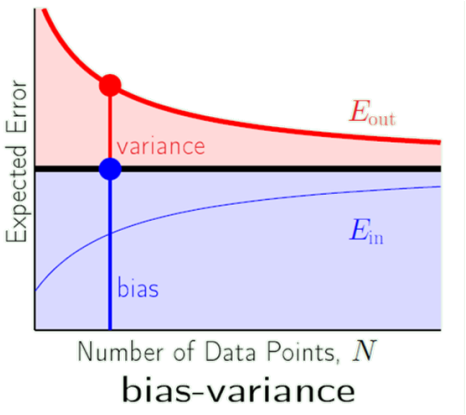


Complex Model

Note: the simple model converges more quickly but to a higher error. In both models, E_{out} decreases while E_{in} increases toward the smallest error the learning model can achieve in approximating f.

VC versus Bias-Variance





In the VC analysis, $E_{out} \leq E_{in} + \Omega$. In the **bias-variance**, it is assumed that, for every N, $\bar{g}$ has the same performance as the best approximation to f in the learning model.

Both capture the tradeoff: Approximation-Generalization

Example - Linear Regression Case

Noisy target $y = f(\mathbf{x}) + \epsilon = \mathbf{w}^T\mathbf{x} + \epsilon$
where ϵ represents noise with zero mean and variance σ^2.

Data set $\mathcal{D} = \{(\mathbf{x}_1, y_1), \cdots, (\mathbf{x}_N, y_N)\}$

Linear regression solution: $\mathbf{w} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$

In sample error vector $= \mathbf{X}\mathbf{w} - \mathbf{y}$

Out-of-sample error vector $= \mathbf{X}\mathbf{w} - \mathbf{y}$'

where $\mathbf{y}$' correspond to the output of the target function to the same inputs $\mathbf{x}$ but with a different realization of the noise. $y' = f(\mathbf{x}) + \epsilon'$

Learning Curves for Linear Regression

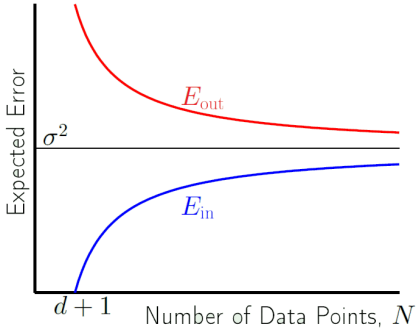
Best approximation error $= \sigma^2$

Expected in-sample error $= \sigma^2\left(1 - \frac{d+1}{N}\right)$

Expected out-of-sample error $= \sigma^2\left(1 + \frac{d+1}{N}\right)$

Expected generalization error $= 2\sigma^2\left(\frac{d+1}{N}\right)$

$d+1 \rightarrow$ VC dimension in perceptron
$d+1 \rightarrow$ 'degrees of freedom' in regression.



Conclusion: the generalization error is a compromise between the 'degrees of freedom' (complexity of the model) and the size of the dataset.