6.036 Introduction to Machine Learning

(meets with 6.862)

Generalization, Complexity, VC-dimension (Chapter 7 in notes)

Administrivia

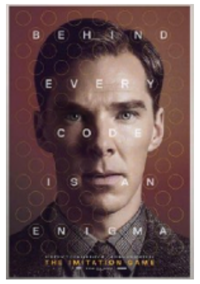
Project #2 due Friday 4/7 @ 9AM.

As always:

- Check LMOD/Piazza for announcements.
- To contact staff, use Piazza (6036-staff@lists.csail.mit.edu for exceptions only)

So far...

- **Tasks:**
  - Binary or multi-class classification

$h\Big($  $\Big) = -1$ $\qquad h : \mathcal{X} \to \{-1, +1\}$

  - Regression

$h\Big($ $\Big) = \$1{,}349{,}000$ $\qquad h : \mathcal{X} \to \mathbb{R}$

- **Formulations:** hyperplane separation, empirical risk minimization, regularization, neural networks,…

- **Algorithms:** perceptron, stochastic gradient, backprop,…

Generalization

Key notion in Machine Learning

- Find classifier that does well on **training set**
- Expect (hope?) that also does well on **test set** (validation set, population)

The real goal is to perform well on the **test set** (population), not just on the **training set**

How to ensure generalization?

Today, we'll study:

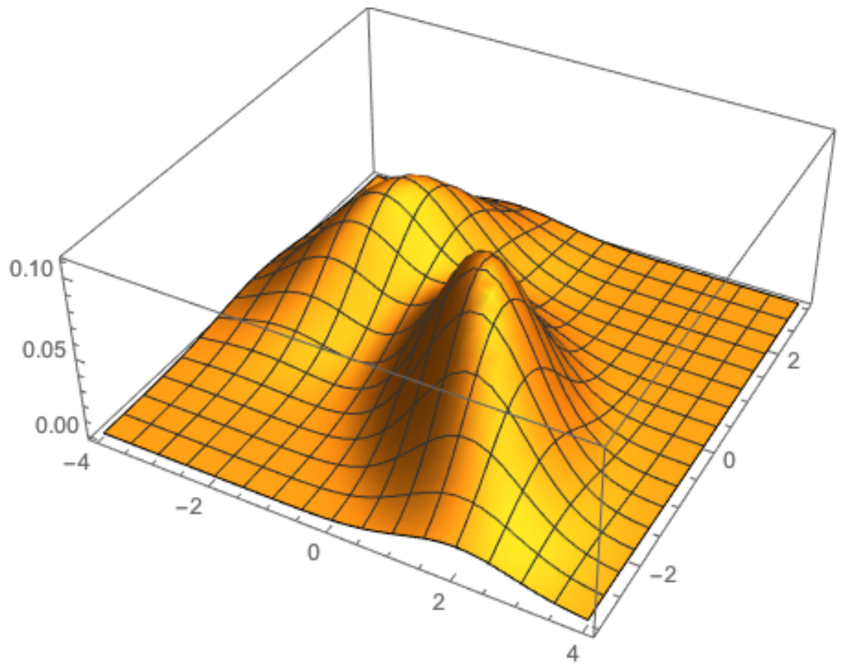
- How to formalize learning/generalization
- Role of parameters (what controls generalization)?
- How to measure the size of a model class

Our setting

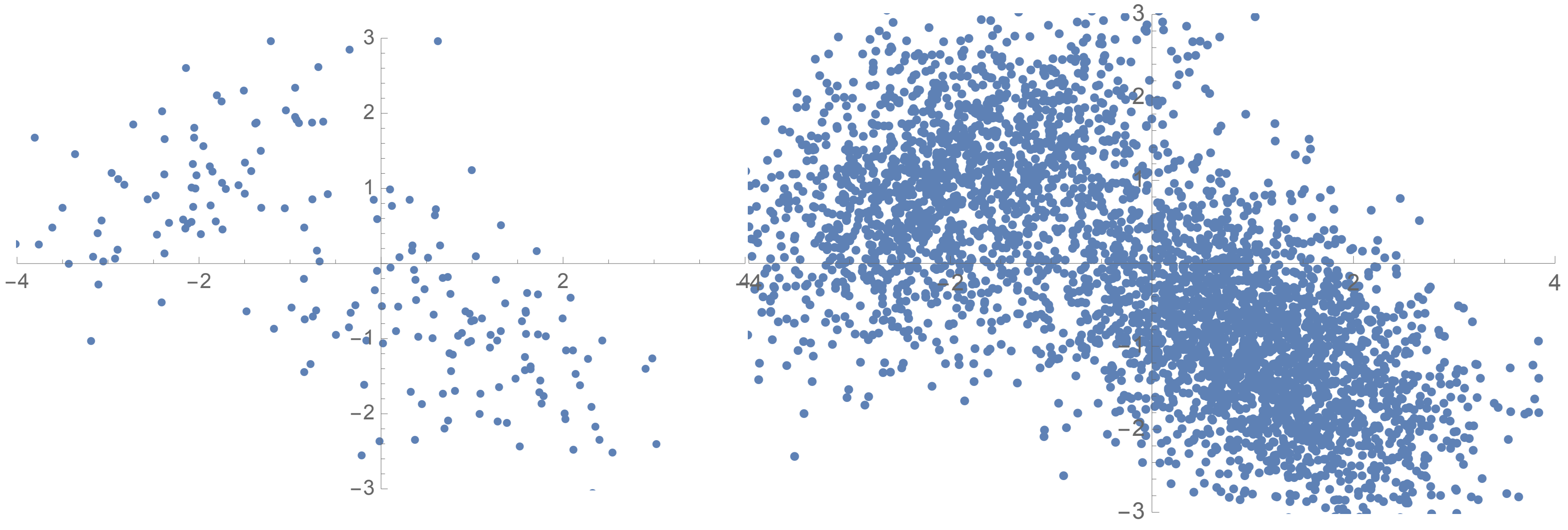
Assume a probabilistic model, where **both** training and validation sets come from the **same** distribution **P***

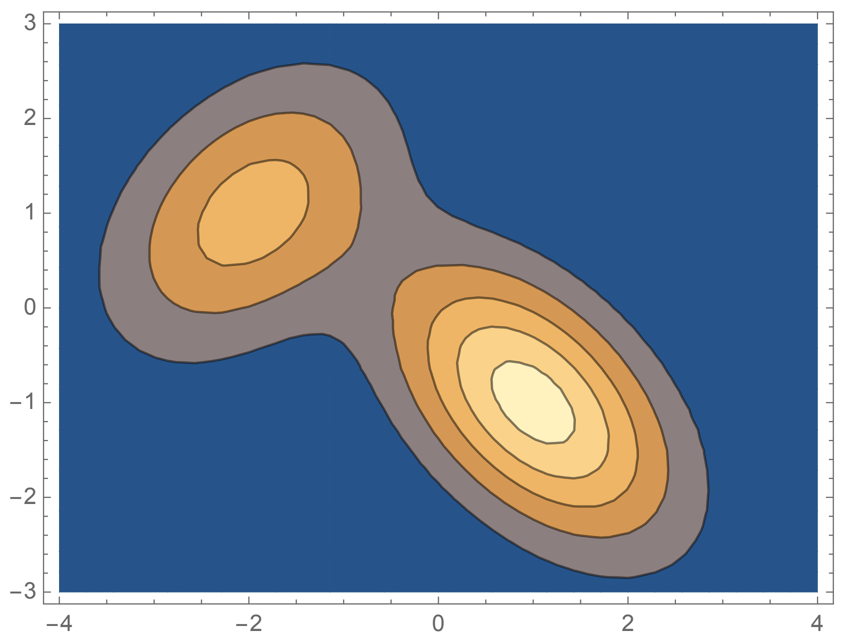
- **Example**: mixture of Gaussians

$$X \sim p_1 \mathcal{N}(\mu_1, \Sigma_1) + p_2 \mathcal{N}(\mu_2, \Sigma_2)$$



- From this:





Training set **Validation set**

Measuring errors

- Distribution **P*** is fixed, but unknown
- Generates both the training and validation sets

How to measure the errors of a given classifier *h(x)*?

- On the training set, **empirical error** (**# of mistakes)**

$$\mathcal{E}_n(h) = \frac{1}{n}\sum_{t=1}^{n} [[y^{(t)} h(x^{(t)}) \leq 0]]$$

Depends on <u>examples</u>!

- On the population, **test** or **generalization error**

$$\mathcal{E}(h) = \mathbf{E}_{(x,y)\sim\mathbf{P}^*}[[yh(x) \leq 0]]$$

Does not, population only

(notice, this is also the prob. of *h* misclassifying a random point)

Empirical risk minimization

- Ideally, would like to directly minimize test error $\mathcal{E}$(h)

$$h^* = \arg\min_{h \in \mathcal{H}} \mathcal{E}(h)$$

but we don't have direct access to this… :(

- Instead, minimize training (empirical) error

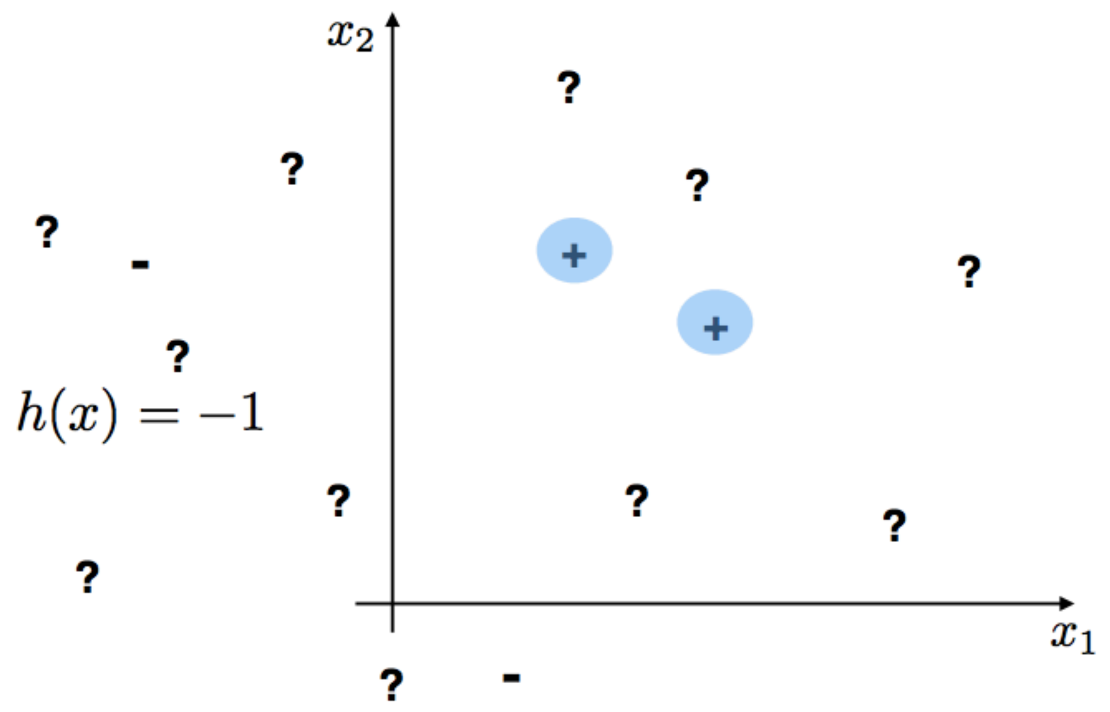
$$\hat{h} = \arg\min_{h \in \mathcal{H}} \mathcal{E}_n(h)$$

Q: When does small $\mathcal{E}_n$(h) imply small $\mathcal{E}$(h)?

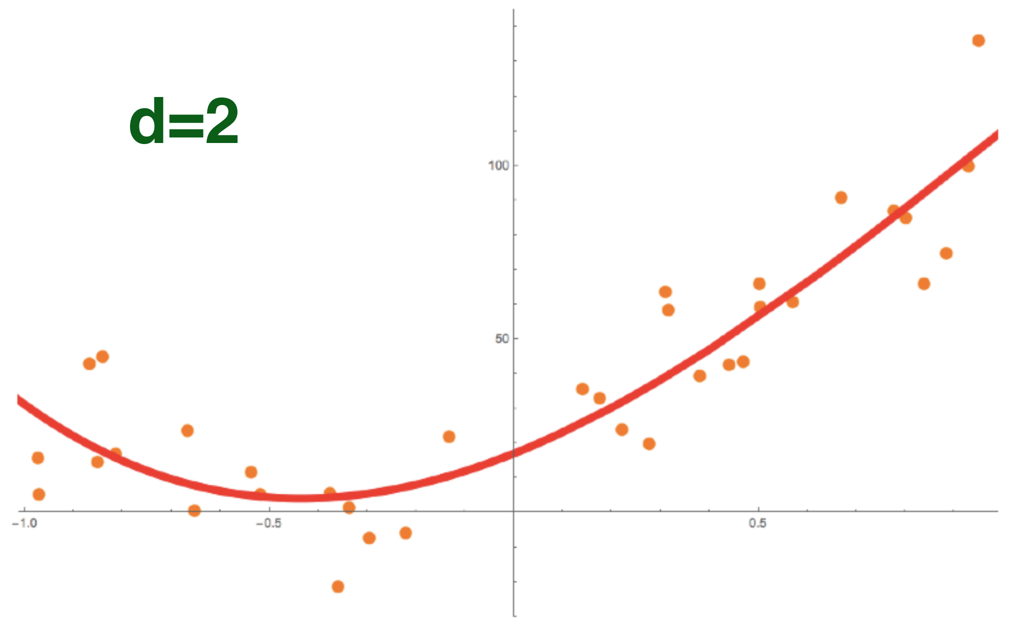
Q: What is the role of the model class $\mathcal{H}$?

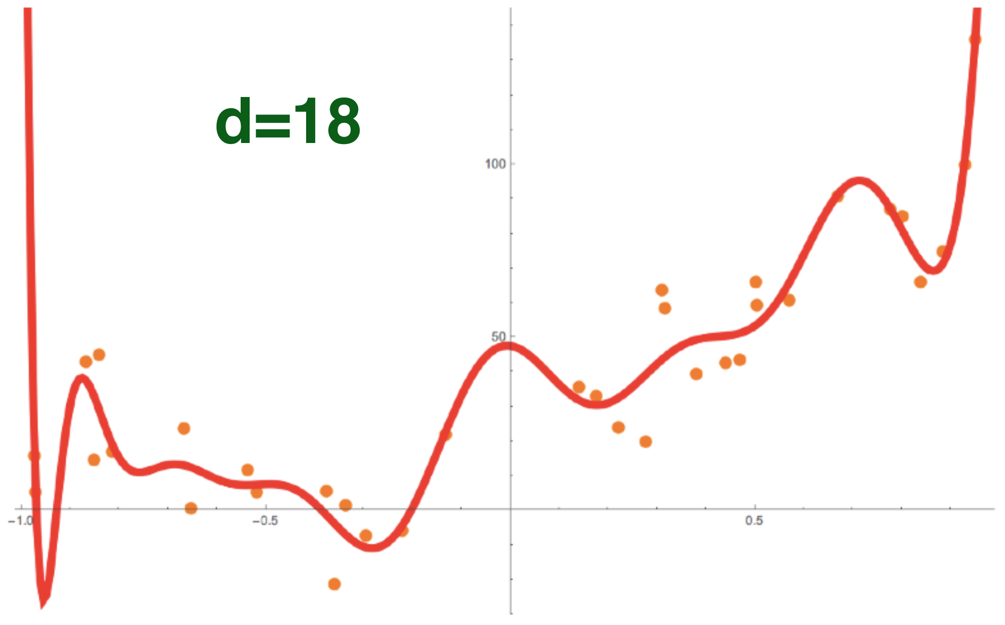
Model class and generalization

- **Intuition**: "small" model classes avoid fitting noise
- From classification lecture:



- From regression lecture:





Generalization guarantees

- What kind of guarantees can we expect?
  - Want small test/generalization error $\mathcal{E}$(h)
  - But, recall that training set is random (could be very bad — though hopefully with small probability)
- <u>Probably Approximately Correct</u> (PAC) Framework

*With probability at least 1-δ over the training set (sampled from **P***), the generalization error satisfies:*

$$\mathcal{E}(\hat{h}) \leq \epsilon$$

Generalization

- What must the relationship between all these parameters be, if we want to ensure learning?

$$\delta, \quad \epsilon, \quad n, \quad |\mathcal{H}|$$

confidence, error bound, number of samples, size of model class

- Assumptions (for simplicity only, can be removed):
  - Finite set of classifiers ($|\mathcal{H}|$ finite)
  - There exists at least one perfect classifier h* (*realizable* case, i.e., $\mathcal{E}$(h*)=0).

Simplified analysis

- Recall PAC: *With probability at least 1-δ over the training set (sampled from **P***), the generalization error is below ε.*

- Fix ε, $\mathcal{H}$, and *n*.
  How small can failure probability δ be?
  (to avoid ERM going badly on training set)

- Pick a "bad" classifier (i.e., one with $\mathcal{E}$(h) > ε)
  - Could ERM conceivably choose this as a solution (i.e., could *h* behave well on training set)?
  - Certainly, if *h* correctly classifies *every* sample: $(1-\varepsilon)^n$

- Since there may be many such classifiers:

$$\delta \leq |\mathcal{H}| \, (1 - \epsilon)^n$$

Generalization bound

$$\delta \leq |\mathcal{H}|\,(1-\epsilon)^n$$

- From this, simple manipulations yield:

$$\log\delta \leq \log|\mathcal{H}| + n\log(1-\epsilon) \leq \log|\mathcal{H}| - n\epsilon$$

- and rearranging

$$\boxed{\epsilon \leq \frac{\log|\mathcal{H}| + \log(1/\delta)}{n}}$$

- Insights:
  - For good generalization, $|\mathcal{H}|$ must be "small"
  - The more confident we want to be (smaller δ), the more training samples we'll need.
  - Increasing number of samples *n* decreases error

More generally...

- This analysis is somewhat restricted (“realizable case”).

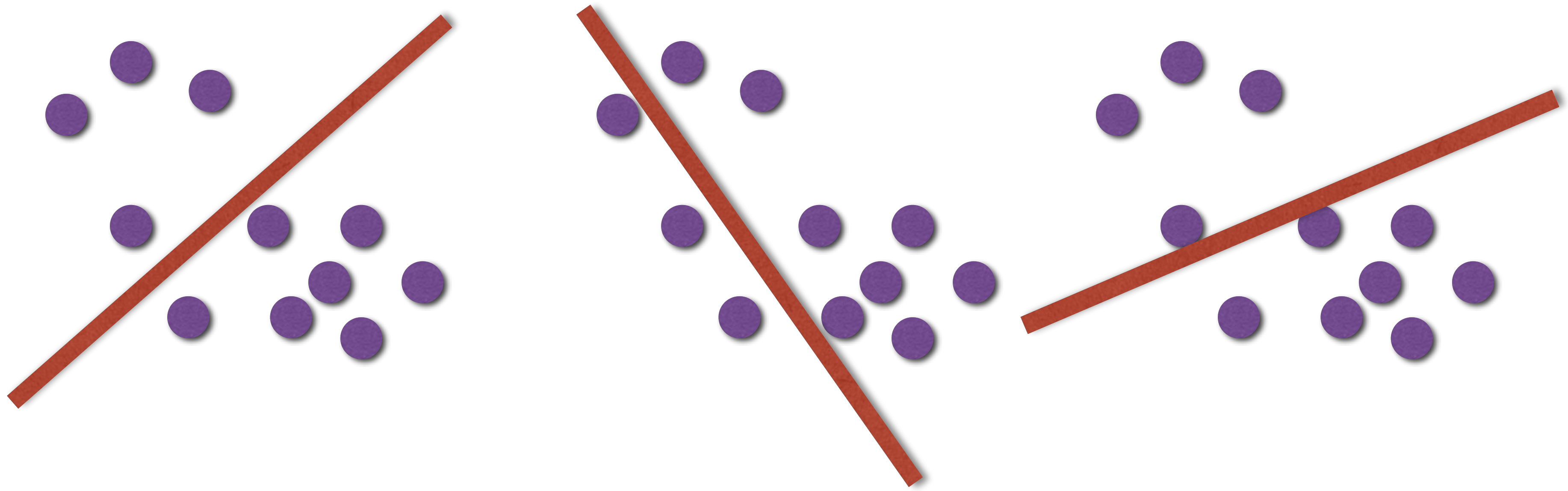
- Can be extended to the general situation:

$$\mathcal{E}(\hat{h}) \leq \mathcal{E}_n(\hat{h}) + \sqrt{\frac{\log|\mathcal{H}| + \log(2/\delta)}{2n}}$$

- Similar qualitative dependence on parameters

Infinite model classes

- In some situations, $|\mathcal{H}|$ is infinite!
  - E.g., set of all linear classifiers



- How to quantify the *size* of the model class? (naive "count" gives infinity!)

- Insight: only thing that matters are the *labelings* induced by a given classifier

Growth function

- Consider the labels induced by all classifiers in $\mathcal{H}$

$$\begin{array}{rcccc} & x^{(1)} & x^{(2)} & \cdots & x^{(n)} \\ h \in \mathcal{H}: & +1 & -1 & \cdots & -1 \\ h' \in \mathcal{H}: & +1 & -1 & \cdots & -1 \\ h'' \in \mathcal{H}: & +1 & +1 & \cdots & -1 \\ \cdots & \cdots & \cdots & \cdots & \cdots \end{array}$$

and let $N_{\mathcal{H}}(x^{(1)}, \ldots, x^{(n)})$ be the number of distinct rows

- This depends on the examples, so maximize

$$N_{\mathcal{H}}(n) = \max_{x^{(1)}, \ldots, x^{(n)}} N_{\mathcal{H}}(x^{(1)}, \ldots, x^{(n)})$$

- This is the *growth function*, and quantifies how powerful our family $\mathcal{H}$ of classifiers is

Generalization guarantees

- Similarly as before, we can give guarantees, but now in terms of the growth function:

$$\mathcal{E}(\hat{h}) \leq \mathcal{E}_n(\hat{h}) + \sqrt{\frac{\log N_{\mathcal{H}}(2n) + \log(4/\delta)}{n}}, \quad \text{for all } \hat{h} \in \mathcal{H}$$

- When does this go to zero as *n* increases?

- Need $\log N_{\mathcal{H}}(2n)$ to grow slower than *n*

VC-dimension

- A measure of complexity of a set of classifiers, the Vapnik-Chervonenkis (VC) dimension d_H
  - Largest number of points that can be labeled in all possible ways using classifiers from $\mathcal{H}$
  - Equivalently, largest *n* for which $N_{\mathcal{H}}(n) = 2^n$

- If $n > d_H$, then the number of labelings grows only logarithmically.

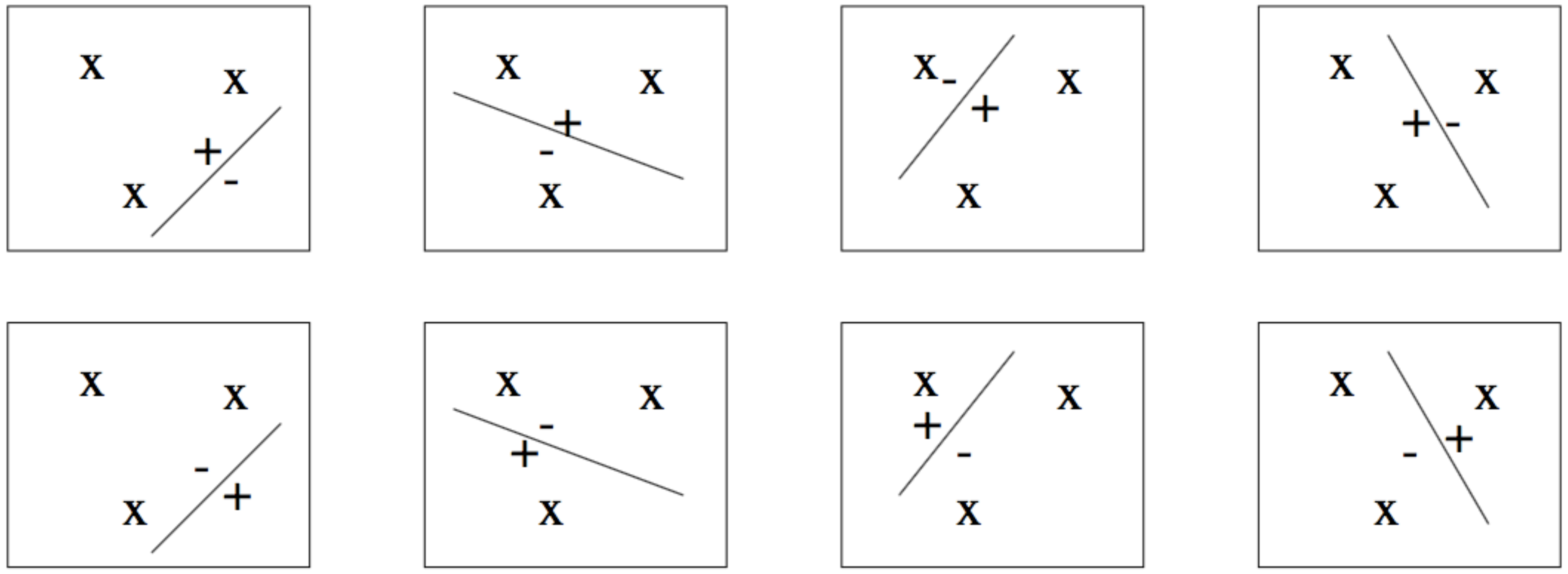
$$\log N_{\mathcal{H}}(2n) \leq d_{\mathcal{H}}(\log(2n/d_{\mathcal{H}}) + 1)$$

- Even if set of classifiers is infinite, finite VC-dimension can guarantee generalization

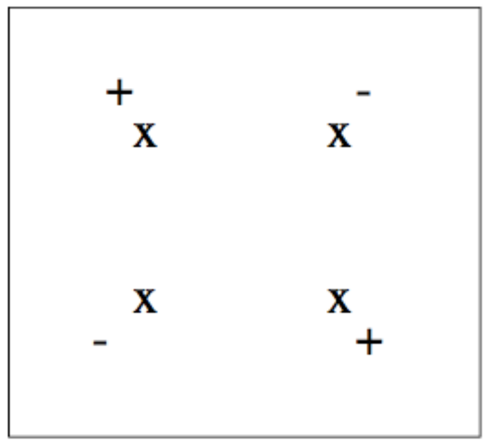
(Other existing approaches: regularization, Rademacher complexity, etc…)

VC-dimension (example)

- **Example**: in R^2, can arbitrarily label 3 points



but not 4 points:



- For *linear classifiers* in R^d, the VC-dimension is *exactly* d+1.

Summary - Generalization

- For learning, need to control generalization error, not just training error.
- PAC (probably approximately correct) framework:
  with high probability, a classifier that has small training error will have small generalization error.
- For generalization, it is sufficient to restrict the size of the model class $\mathcal{H}$
- For uncountable model classes, can quantify size using VC-dimension