Learning Theory: Why ML Works

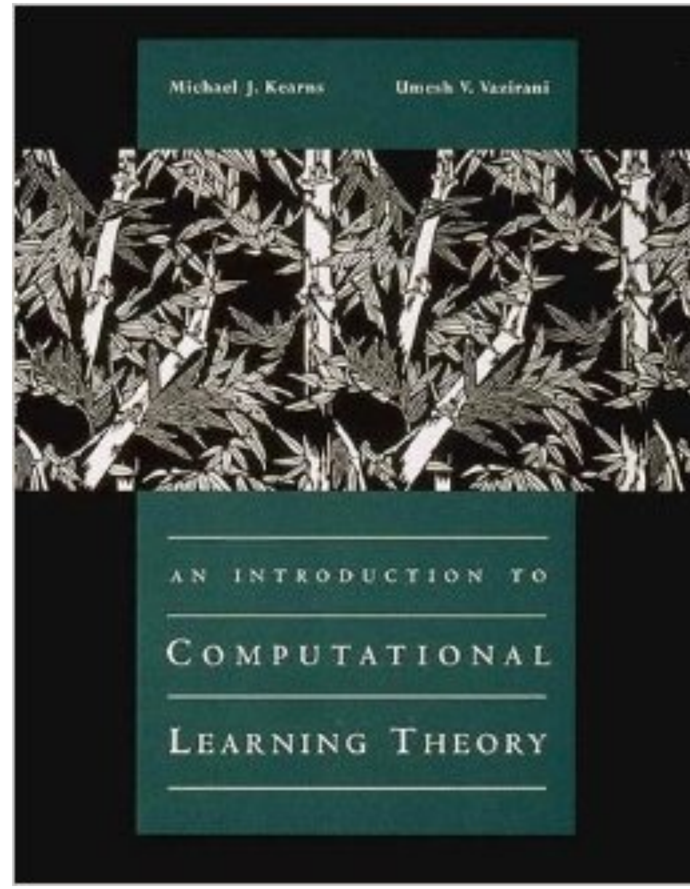
These slides were assembled by Eric Eaton, with grateful acknowledgement of the many others who made their course materials freely available online. Feel free to reuse or adapt these slides for your own academic purposes, provided that you include proper attribution. Please send comments and corrections to Eric.

Computational Learning Theory

Entire subfield devoted to the mathematical analysis of machine learning algorithms

Has led to several practical methods:

- PAC (probably approximately correct) learning → boosting
- VC (Vapnik–Chervonenkis) theory → support vector machines



Annual conference: Conference on Learning Theory (COLT)

Computational Learning Theory

Fundamental Question: What general laws constrain inductive learning?

Seeks theory to relate:

- Probability of successful learning
- Number of training examples
- Complexity of hypothesis space
- Accuracy to which target function is approximated
- Manner in which training examples should be presented

Sample Complexity

Assume that $f : \mathcal{X} \mapsto \{0, 1\}$ is the target concept

How many training examples are sufficient to learn the target concept f ?

1. If learner proposed instances as queries to teacher
   - Learner proposes instance x, teacher provides $\mathrm{f}(\mathrm{x})$
2. If teacher (who knows f) provides training examples
   - Teacher provides labeled examples in form $<\mathrm{x}, \mathrm{f}(\mathrm{x})>$
3. If some random process (e.g., nature) proposes instances
   - Instance x generated randomly, teacher provides $\mathrm{f}(\mathrm{x})$

Based on slide by Tom Mitchell

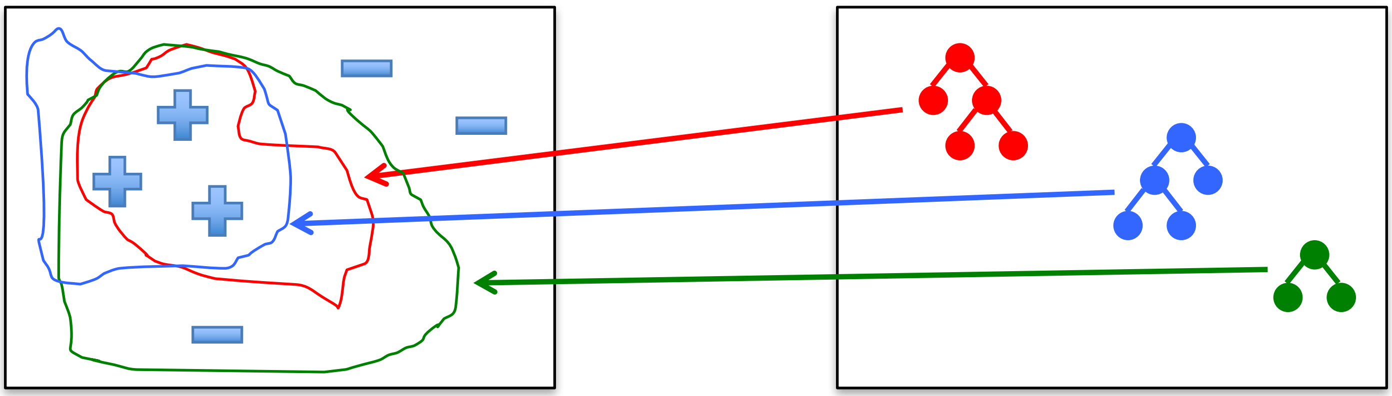
Function Approximation: The Big Picture

Instance Space $\mathcal{X} = \{0,1\}^d$

$\boldsymbol{x} = \langle x_1, x_2, \ldots, x_d \rangle \in \mathcal{X}$

Hypothesis Space

$H = \{h \mid h : \mathcal{X} \mapsto \{0,1\}\}$



if $d = 20$, $|\mathcal{X}| = 2^{20}$

$|h| = 2^{|\mathcal{X}|} = 2^{2^{20}}$

- How many labeled instances are needed to determine which of the $2^{2^{20}}$ hypotheses are correct?
  - <u>All</u> 2^{20} instances in $\mathcal{X}$ must be labeled!
- Generalizing beyond the training data (inductive inference) is impossible unless we add more assumptions (e.g., priors over H)

Based on example by Tom Mitchell

Bias-Variance Decomposition of Squared Error

- Assume that $y = f(\boldsymbol{x}) + \epsilon$
  - Noise ϵ is sampled from a normal distribution with 0 mean and variance σ^2: $\epsilon \sim N(0, \sigma^2)$
  - Noise lower-bounds the performance (error) we can achieve

- Recall the following objective function:

$$J(\boldsymbol{\theta}) = \frac{1}{n} \sum_{i=1}^{n} \left(y^{(i)} - h_{\boldsymbol{\theta}} \left(\boldsymbol{x}^{(i)} \right) \right)^2$$

- We can view this as an approximation of the expected value of the squared error: $\mathrm{E} \left(y - h_{\boldsymbol{\theta}} \left(\boldsymbol{x} \right) \right)^2$

Bias-Variance Decomposition of Squared Error

$$
\begin{aligned}
\mathrm{E}[(y - h_{\boldsymbol{\theta}}(\boldsymbol{x}))^2] &= \mathrm{E}[(y - f(\boldsymbol{x}) + f(\boldsymbol{x}) - h_{\boldsymbol{\theta}}(\boldsymbol{x}))^2] \\
&= \mathrm{E}[(y - f(\boldsymbol{x}))^2] + \mathrm{E}[(f(\boldsymbol{x}) - h_{\boldsymbol{\theta}}(\boldsymbol{x}))^2] \\
&\qquad + 2\,\mathrm{E}[(f(\boldsymbol{x}) - h_{\boldsymbol{\theta}}(\boldsymbol{x}))(y - f(\boldsymbol{x}))] \\
&= \mathrm{E}[(y - f(\boldsymbol{x}))^2] + \mathrm{E}[(f(\boldsymbol{x}) - h_{\boldsymbol{\theta}}(\boldsymbol{x}))^2] \\
&\qquad + 2\left(\cancel{\mathrm{E}[f(\boldsymbol{x}) h_{\boldsymbol{\theta}}(\boldsymbol{x})]} + \cancel{\mathrm{E}[y f(\boldsymbol{x})]} - \cancel{\mathrm{E}[y h_{\boldsymbol{\theta}}(\boldsymbol{x})]} - \cancel{\mathrm{E}[f(\boldsymbol{x})^2]}\right)
\end{aligned}
$$

cancels cancels

Therefore,

$$
\begin{aligned}
\mathrm{E}[(y - h_{\boldsymbol{\theta}}(\boldsymbol{x}))^2] &= \mathrm{E}[(y - f(\boldsymbol{x}))^2] + \mathrm{E}[(f(\boldsymbol{x}) - h_{\boldsymbol{\theta}}(\boldsymbol{x}))^2] \\
&= \mathrm{E}[\epsilon^2] + \mathrm{E}[(f(\boldsymbol{x}) - h_{\boldsymbol{\theta}}(\boldsymbol{x}))^2]
\end{aligned}
$$

This is actually $\mathrm{var}(\epsilon)$, since mean is 0

Aside:
Definition of Variance
$\mathrm{var}(z) = \mathrm{E}[(z - \mathrm{E}[z])^2]$

Bias-Variance Decomposition of Squared Error

$$
\begin{aligned}
\mathrm{E}[(y - h_{\boldsymbol{\theta}}(\boldsymbol{x}))^2] &= \mathrm{var}(\epsilon) + \mathrm{E}[(f(\boldsymbol{x}) - h_{\boldsymbol{\theta}}(\boldsymbol{x}))^2] \\
&= \mathrm{var}(\epsilon) + \mathrm{E}[(f(\boldsymbol{x}) - \mathrm{E}[h_{\boldsymbol{\theta}}(\boldsymbol{x})] + \mathrm{E}[h_{\boldsymbol{\theta}}(\boldsymbol{x})] - h_{\boldsymbol{\theta}}(\boldsymbol{x}))^2] \\
&= \mathrm{var}(\epsilon) + \mathrm{E}[(f(\boldsymbol{x}) - \mathrm{E}[h_{\boldsymbol{\theta}}(\boldsymbol{x})])^2] + \mathrm{E}[(\mathrm{E}[h_{\boldsymbol{\theta}}(\boldsymbol{x})] - h_{\boldsymbol{\theta}}(\boldsymbol{x}))^2] \\
&\qquad + 2\mathrm{E}[(\mathrm{E}[h_{\boldsymbol{\theta}}(\boldsymbol{x})] - h_{\boldsymbol{\theta}}(\boldsymbol{x}))(f(\boldsymbol{x}) - \mathrm{E}[h_{\boldsymbol{\theta}}(\boldsymbol{x})])] \\
&= \mathrm{var}(\epsilon) + \mathrm{E}[(f(\boldsymbol{x}) - \mathrm{E}[h_{\boldsymbol{\theta}}(\boldsymbol{x})])^2] + \mathrm{E}[(\mathrm{E}[h_{\boldsymbol{\theta}}(\boldsymbol{x})] - h_{\boldsymbol{\theta}}(\boldsymbol{x}))^2] \\
&\qquad + 2\left(\cancel{\mathrm{E}[f(\boldsymbol{x})\mathrm{E}[h_{\boldsymbol{\theta}}(\boldsymbol{x})]]} - \cancel{\mathrm{E}[\mathrm{E}[h_{\boldsymbol{\theta}}(\boldsymbol{x})]^2]} - \cancel{\mathrm{E}[f(\boldsymbol{x})h_{\boldsymbol{\theta}}(\boldsymbol{x})]} + \cancel{\mathrm{E}[h_{\boldsymbol{\theta}}(\boldsymbol{x})\mathrm{E}[h_{\boldsymbol{\theta}}(\boldsymbol{x})]]}\right)
\end{aligned}
$$

cancels cancels

Therefore,

$$
\mathrm{E}[(y - h_{\boldsymbol{\theta}}(\boldsymbol{x}))^2] = \underbrace{\mathrm{var}(\epsilon)}_{\text{noise}} + \underbrace{\mathrm{E}[(f(\boldsymbol{x}) - \mathrm{E}[h_{\boldsymbol{\theta}}(\boldsymbol{x})])^2]}_{\text{bias}} + \underbrace{\mathrm{E}[(\mathrm{E}[h_{\boldsymbol{\theta}}(\boldsymbol{x})] - h_{\boldsymbol{\theta}}(\boldsymbol{x}))^2]}_{\text{variance}}
$$

$$
\mathrm{E}[(y - h_{\boldsymbol{\theta}}(\boldsymbol{x}))^2] = \mathrm{bias}(h_{\boldsymbol{\theta}}(\boldsymbol{x}))^2 + \mathrm{var}(h_{\boldsymbol{\theta}}(\boldsymbol{x})) + \sigma^2
$$

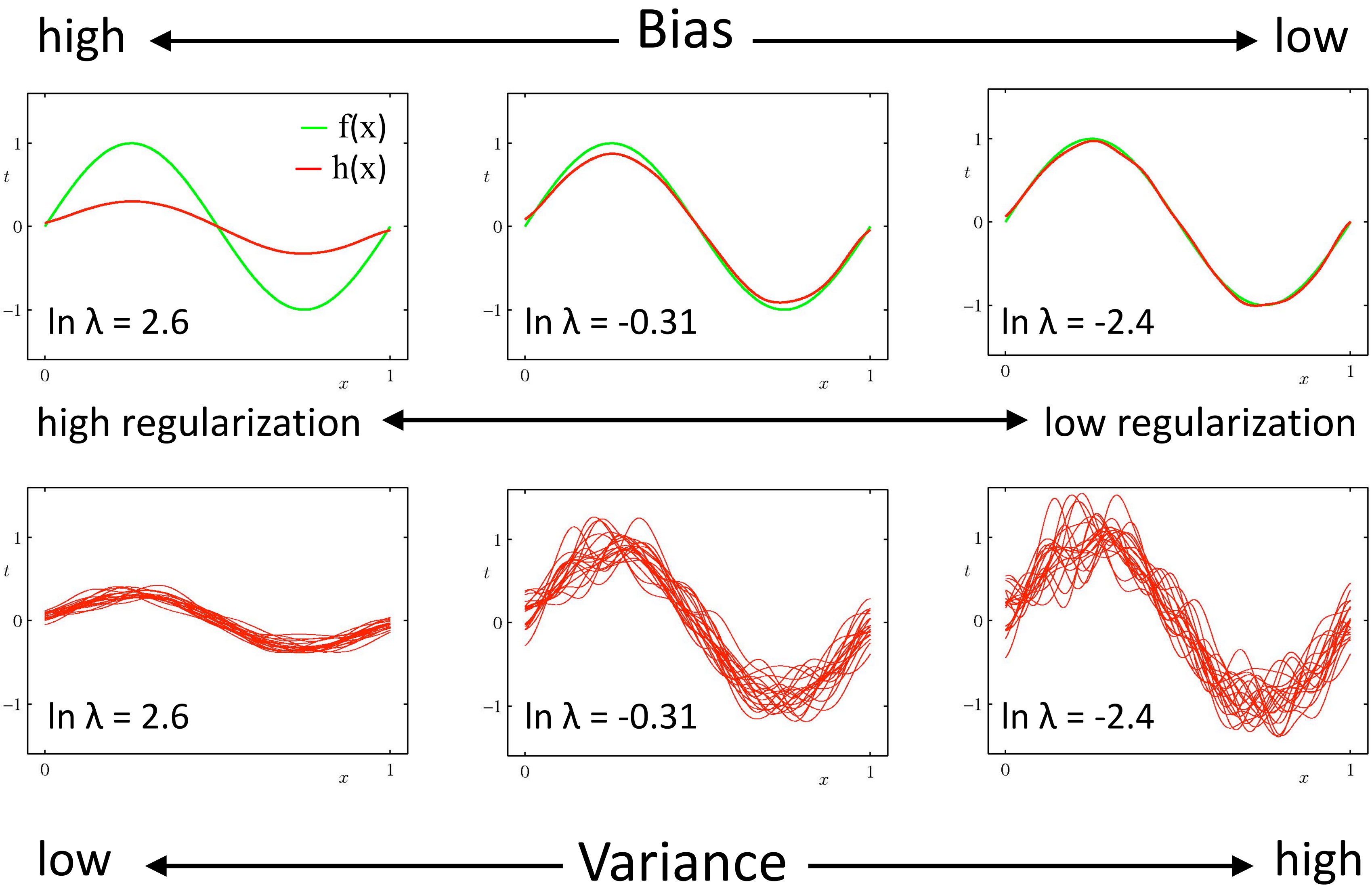
Regularization

- Linear regression objective function

$$J(\boldsymbol{\theta}) = \underbrace{\frac{1}{2n}\sum_{i=1}^{n}\left(h_{\boldsymbol{\theta}}\left(\boldsymbol{x}^{(i)}\right) - y^{(i)}\right)^2}_{\text{model fit to data}} + \underbrace{\frac{\lambda}{2}\sum_{j=1}^{d}\theta_j^2}_{\text{regularization}}$$

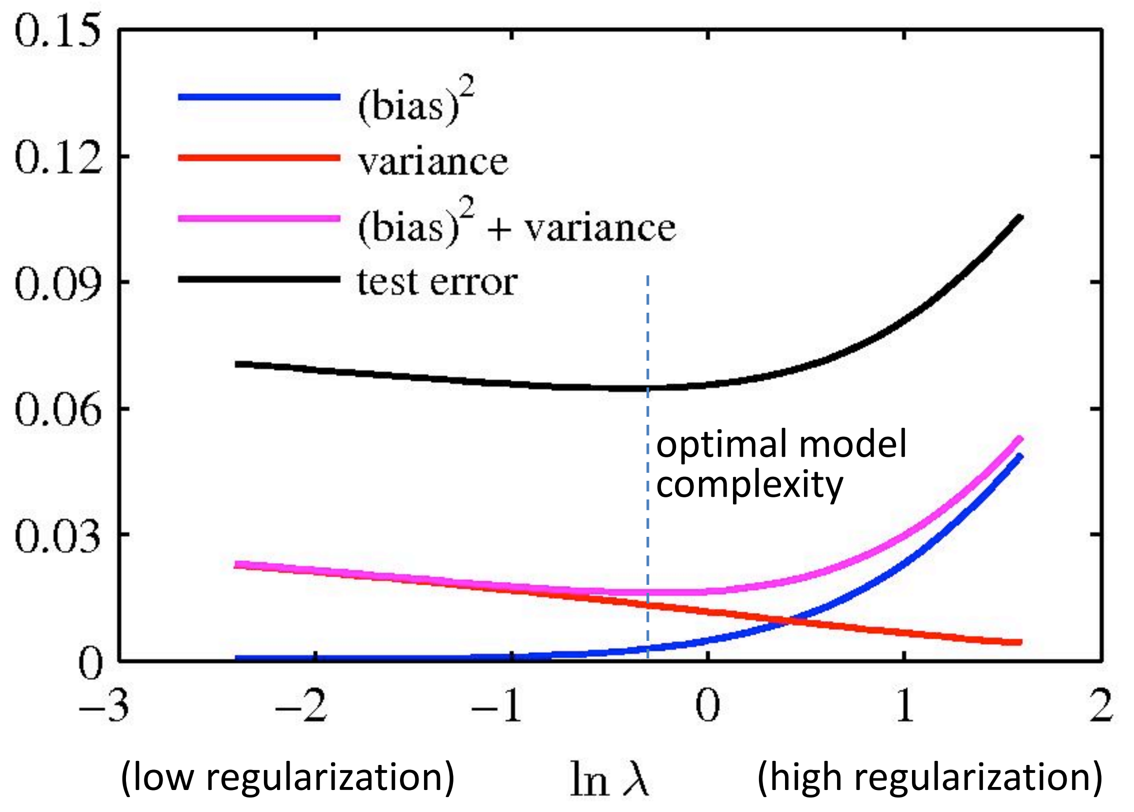
  – λ is the regularization parameter ($\lambda \geq 0$)

Illustration of Bias-Variance



Figures provided by by Max Welling

Illustration of Bias-Variance



- Training error drives down bias, but ignores variance

Figure provided by by Max Welling

A Way to Choose the Best Model

- It would be really helpful if we could get a guarantee of the following form:

$$\text{testingError} \leq \text{trainingError} + f(n, h, p)$$

  n = size of training set

  h = measure of the model complexity

  p = the probability that this bound fails

  We need p to allow for really unlucky test sets

- Then, we could choose the model complexity that minimizes the bound on the test error

A Weird Measure of Model Complexity

- Suppose that we pick n data points and assign labels of + or – to them at random
- If our model class (e.g., a decision tree, polynomial regression of a particular degree, etc.) can learn any association of labels with data, it is too powerful!

More power: can model more complex functions, but may overfit

Less power: won't overfit, but limited in what it can represent

- **Idea**: characterize the power of a model class by asking how many data points it can learn perfectly for all possible assignments of labels
  – This number of data points is called the Vapnik-Chervonenkis (VC) dimension

VC Dimension

- A measure of the power of a particular class of models
  - It does not depend on the choice of training set

- The VC dimension of a model class is the maximum number of points that can be arranged so that the class of models can shatter those points

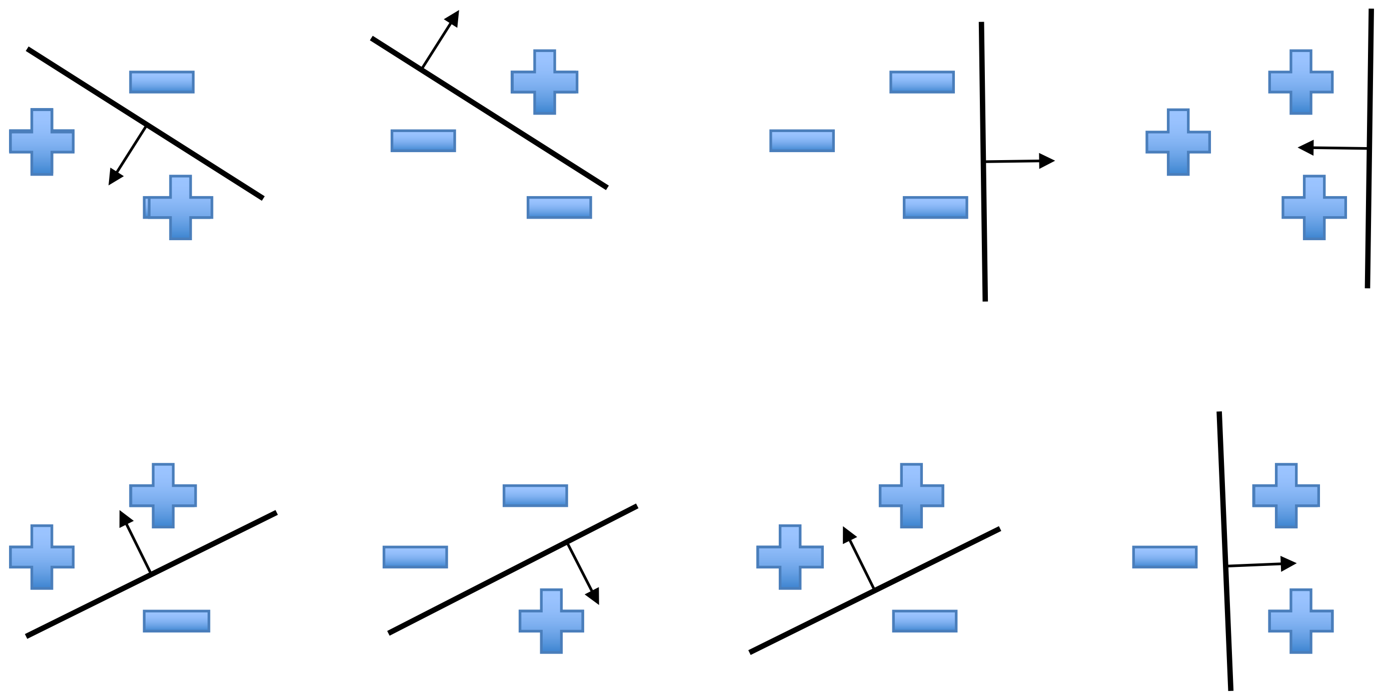
Definition: a model class can shatter a set of points

$$\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \ldots, \boldsymbol{x}^{(r)}$$

if for <u>every</u> possible labeling over those points, there exists a model in that class that obtains zero training error

An Example of VC Dimension

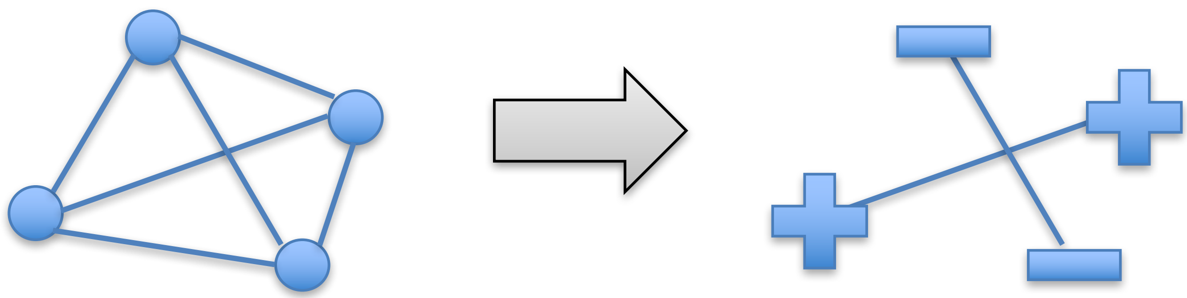
- Suppose our model class is a hyperplane
- Consider all labelings over three points in $\mathbb{R}^2$



- In $\mathbb{R}^2$, we can find a plane (i.e., a line) to capture any labeling of 3 points. A 2D hyperplane shatters 3 points

An Example of VC Dimension

- But, a 2D hyperplane cannot deal with some labelings of four points:



Connect all pairs of points; two lines will always cross

Can’t separate points if the pairs that cross are the same class

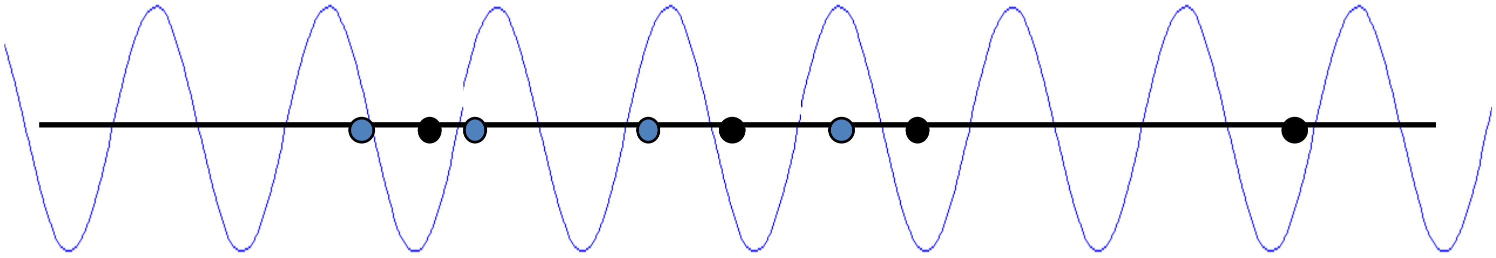
- Therefore, a 2D hyperplane cannot shatter 4 points

Some Examples of VC Dimension

- The VC dimension of a hyperplane in 2D is 3.
  - In d dimensions it is $d+1$
    - It's just a coincidence that the VC dimension of a hyperplane is almost identical to the # parameters needed to define a hyperplane

- A sine wave has infinite VC dimension and only 2 parameters!
  - By choosing the phase & period carefully we can shatter any random set of 1D data points (except for nasty special cases)

$$h(x) = a\sin(bx)$$



Assumptions

- Given some model class (which defines the hypothesis space H)
- Assume all training points were drawn i.i.d from distribution $\mathcal{D}$
- Assume all future test points will be drawn from $\mathcal{D}$

Definitions:

$$R(\boldsymbol{\theta}) = \text{testError}(\boldsymbol{\theta}) = \underbrace{E\left[\frac{1}{2}|y - h_{\boldsymbol{\theta}}(\boldsymbol{x})|\right]}_{\text{probability of misclassification}}$$

"official" notation

notation we'll use

$$R^{\text{emp}}(\boldsymbol{\theta}) = \text{trainError}(\boldsymbol{\theta}) = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{2}\left|y^{(i)} - h_{\boldsymbol{\theta}}(\boldsymbol{x}^{(i)})\right|$$

A Probabilistic Guarantee of Generalization Performance

Vapnik showed that with probability (1 –η):

$$\mathrm{testError}(\boldsymbol{\theta}) \leq \mathrm{trainError}(\boldsymbol{\theta}) + \sqrt{\frac{h(\log(2n/h)+1) - \log(\eta/4)}{n}}$$

n = size of training set

h = VC dimension of model class

η= the probability that this bound fails

- So, we should pick the model with the complexity that minimizes this bound
  - Actually, this is only sensible if we think the bound is fairly tight, which it usually isn’t
  - The theory provides insight, but in practice we still need some witchcraft

Take Away Lesson

Suppose we find a model with a low training error...

- If hypothesis space H is very big (relative to the size of the training data n), then we most likely overfit

- If the following holds:
  - H is sufficiently constrained in size
  - and/or the size of the training data set n is large,

  then low training error is likely to be evidence of low generalization error