Elemental Learning Theory

Alexander G. Ororbia II

Introduction to Machine Learning

CSCI-635

9/20/2023

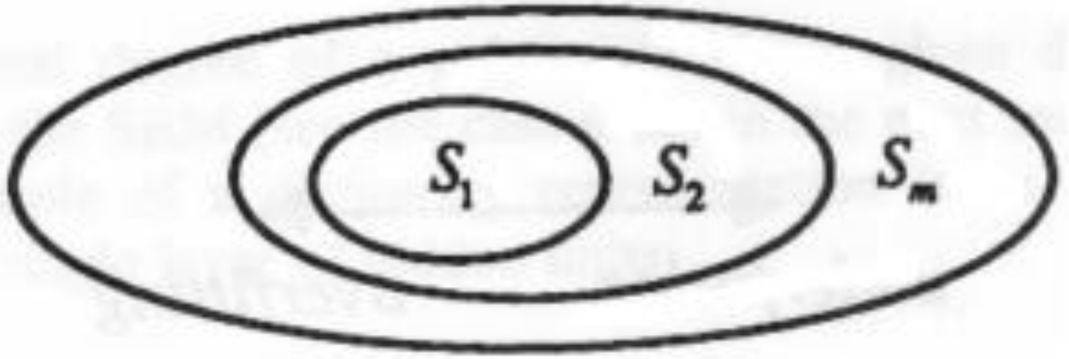
Representational and Effective Capacity

- Representational capacity:
  - Specifies family of functions learning algorithm can choose from
- Effective capacity:
  - Imperfections in optimization algorithm can limit representational capacity
- Occam's razor:
  - Among competing hypotheses that explain known observations equally well, choose the simplest one
    - Idea is formalized in VC dimension

Some Basic Statistical Learning Theory

- ***Capacity*** = ability to fit wide variety of functions
  - ***Low*** → model struggles to fit training data (underfit)
  - ***High*** → model can "memorize" data (overfit)
- ***Hypothesis space*** = set of functions learning algorithm is allowed to learn

  linear regression → linear functions
  polynomial regression → ***??***

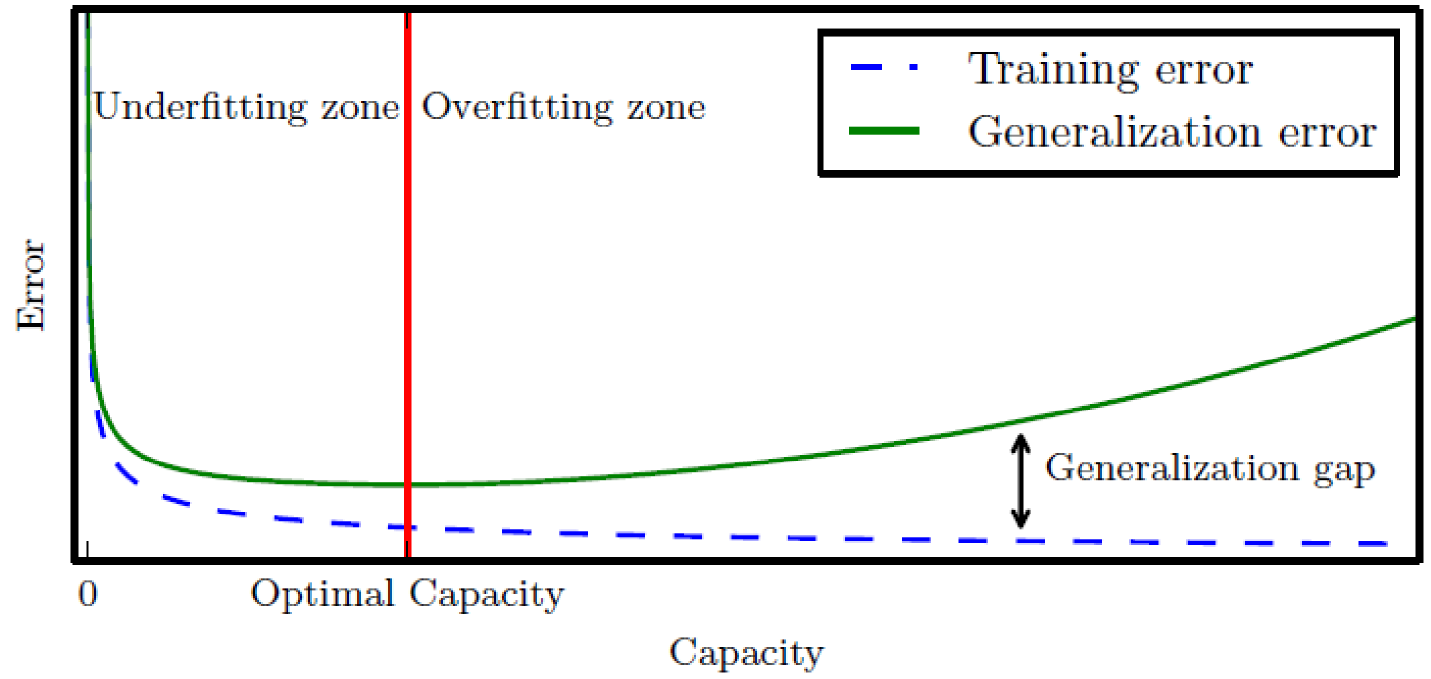


Generalization and Capacity

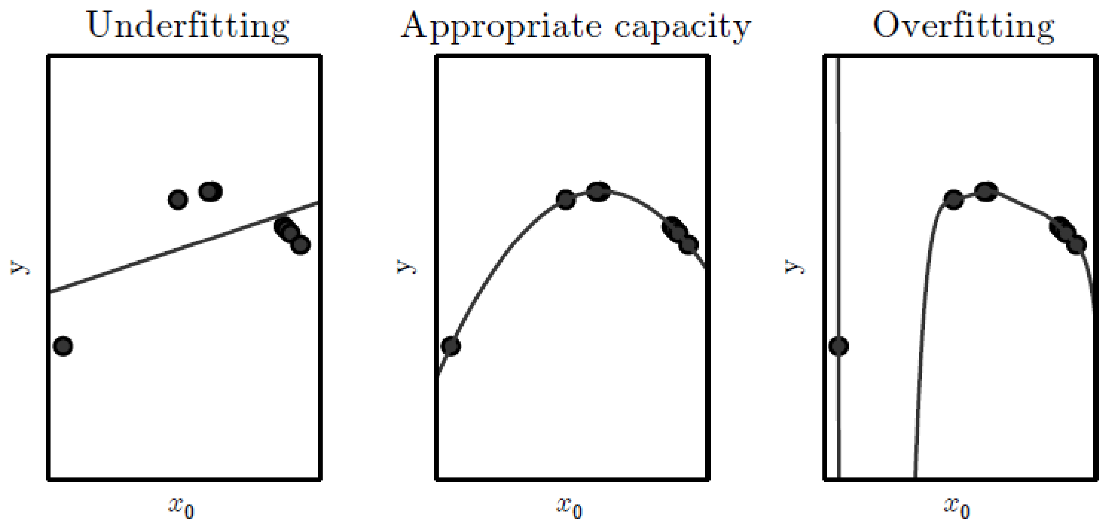
Model capacity:
Ability to fit a wide variety of functions, can be controlled by choosing a hypothesis space.

(High capacity → model tends to memorize noise)

Hypothesis space:
Set of functions that learner is allowed to "select" as a solution



Underfitting and Overfitting in Polynomial Estimation



Fundamental Principle of Statistical Learning

- When sample size is small → model should be simple
  - We must deliberately oversimplify our models!
  - Can use the VC/PAC to guide us
    - Occam's Razor = parsimony, choose "the simplest one"

$$\text{error rate} \propto \frac{\text{model complexity}}{\text{sample size}}$$

  - What does this mean for deeper architectures & non-parametric models??

Vapnik-Chervonenkis (VC) Dimension

- Measure of capacity (complexity, expressive power, richness, or flexibility) of a space of functions learnable by a statistical (binary) classification algorithm
- Largest possible value of *m* for which there exists training set of different points (**x**) that algorithm/classifier label arbitrarily
  - Discrepancy between training & generalization (test) error bounded above
    - Bound grows as capacity grows but shrinks as number of training samples increases
  - More data = better worst case generalization error!

Probably Approximately Correct (The PAC Framework)

- Learner receives samples & must select a generalization function (hypothesis) from certain class of possible functions
- Goal: with high probability ("probably"), the selected function will have low generalization error ("approximately correct")
  - Learner must be able to learn the concept given any arbitrary approximation ratio, probability of success, or distribution of the samples
  - Learner must be efficient (w.r.t. time-space complexity; bounded to polynomial of sample size)
  - Learner should find efficient function given sample size that is polynomially upper bounded, further accounting for approximation/likelihood adjustments/bounds

Finding the Best Model

- Overly complex family does not necessarily include target function, true data generating process, or even an approximation

- Best fitting model obtained not by finding the right number of parameters
- Instead, best fitting model is a large model that has been regularized appropriately

No Free Lunch Theorem

You can only get generalization through assumptions. No one algorithm will solve all problems (some will work better than others in some instances).

Slide credit: D. Hoiem

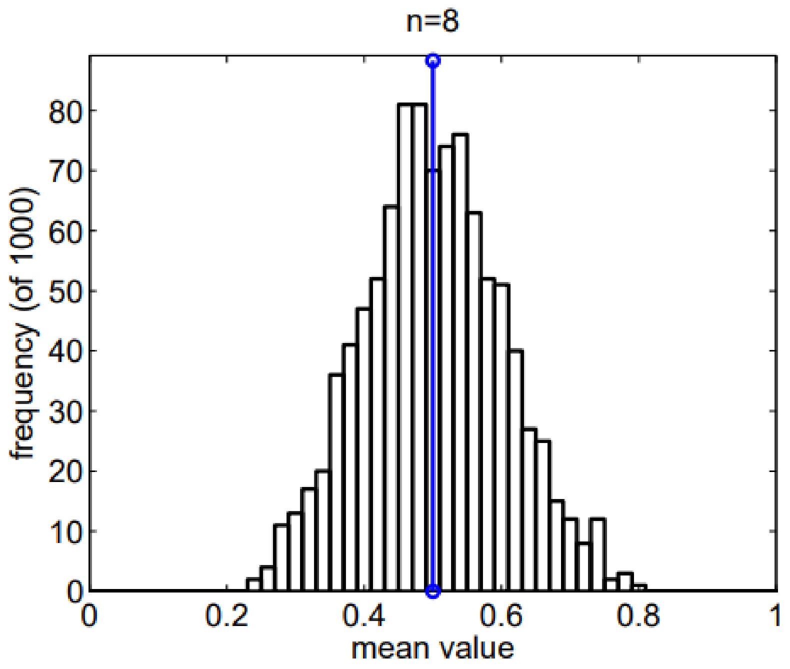
How to Reduce Variance?

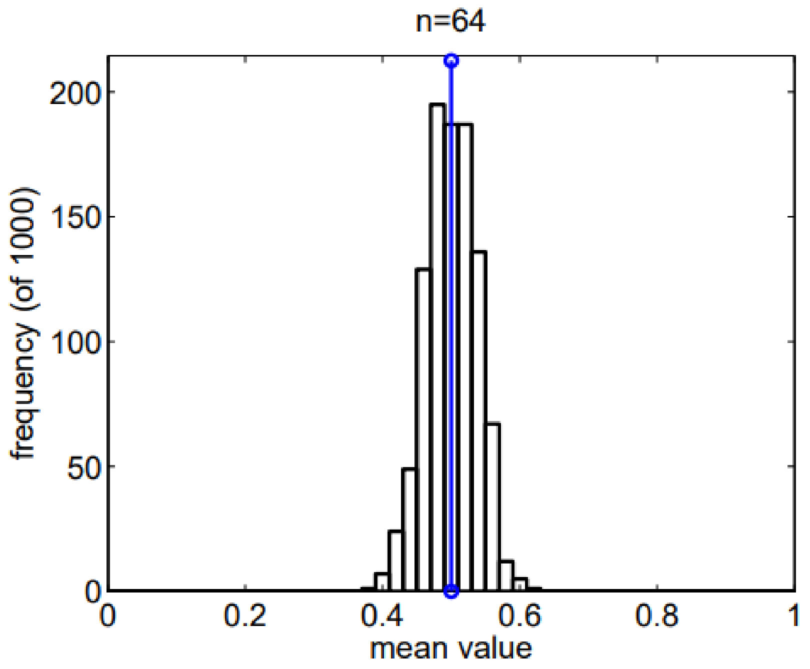
- Choose a simpler classifier
  - Occam's Razor: *Among competing hypotheses, the one with the fewest assumptions should be selected.*
- Regularize the parameters
  - Apply penalties / constraints
  - Go Bayesian – apply prior distributions over parameters
- Get more training data
  - ***BIG*** data (some theory justifies this!)

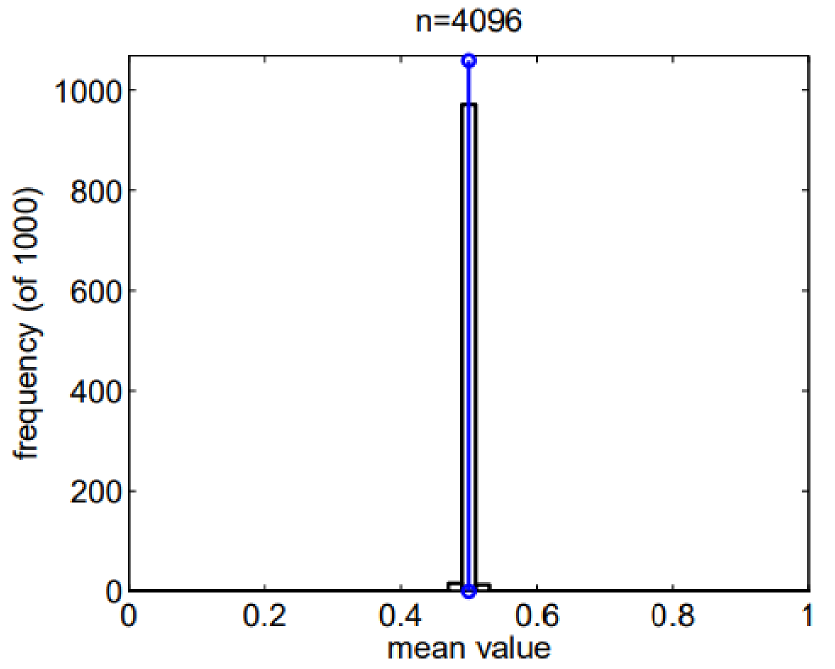
Slide credit: D. Hoiem

The Law of Large Numbers

- The (strong) law of large numbers
  - In limit of infinite data, or $N \to \infty$, sample mean converges to expected value $S_n \to \mu$ of data distribution
  - Intuitively, with more data, our estimate of a quantity/moment such as mean, estimate becomes more accurate
- A partial justification for big data!







Arbitrary Capacity and Nonparametric Models

- When do we reach most extreme case of arbitrarily high capacity?
- Parametric models such as linear regression:
  - learn a function described by a parameter whose size is finite and fixed before data is observed
- Nonparametric models have no such limitation
- Nearest-neighbor regression is an example
  - Their complexity is a function of training set size

K-NN = K nearest neighbors

Nearest Neighbor Regression

- Simply store the $\boldsymbol{X}$ and $\boldsymbol{t}$ from the training set
- When asked to classify a test point $\boldsymbol{x}$ the model looks up the nearest entry in the training set and returns the associated target, i.e.,

$$\hat{t} = t_i \quad \text{where} \quad i = \arg\min \|\boldsymbol{X}_{i,:} - \boldsymbol{x}\|_2^2$$

- Algorithm can be generalized to distance metrics other than L^2 norm such as learned distance metrics

Usually used w/ feature scaling:

$$a_i = \frac{v_i - \min v_i}{\max v_i - \min v_i}$$

Can also employ feature-weighted NN:

$$D(a,b) = \sqrt{\sum_k w_k (a_k - b_k)^2}$$

NN = Nearest Neighbor (1NN is K=1 Nearest Neighbor(s) Model)

The K-NN Model

Algorithm: ***K* = 1**

1. Find example $(\mathbf{x}^*, t^*)$ (from the stored training set) closest to the test instance $\mathbf{x}$. That is:

$$\mathbf{x}^* = \underset{\mathbf{x}^{(i)} \in \text{train. set}}{\operatorname{argmin}} \; \text{distance}(\mathbf{x}^{(i)}, \mathbf{x})$$

2. Output $y = t^*$

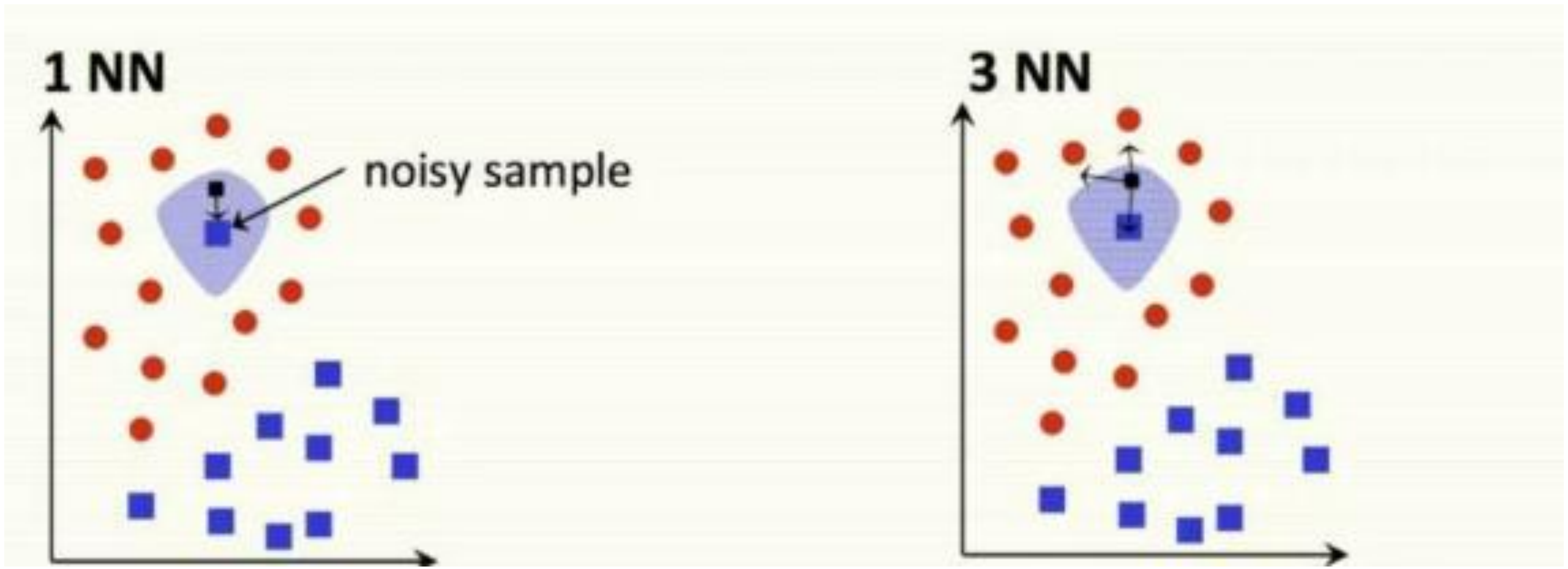
Algorithm (kNN):

1. Find k examples $\{\mathbf{x}^{(i)}, t^{(i)}\}$ closest to the test instance $\mathbf{x}$
2. Classification output is majority class

$$y = arg\,\max_{t^{(z)}} \sum_{r=1}^{k} \delta(t^{(z)}, t^{(r)})$$

Over z ∈ Z classes

(Lazy) Instance-Based Learning with K-NN

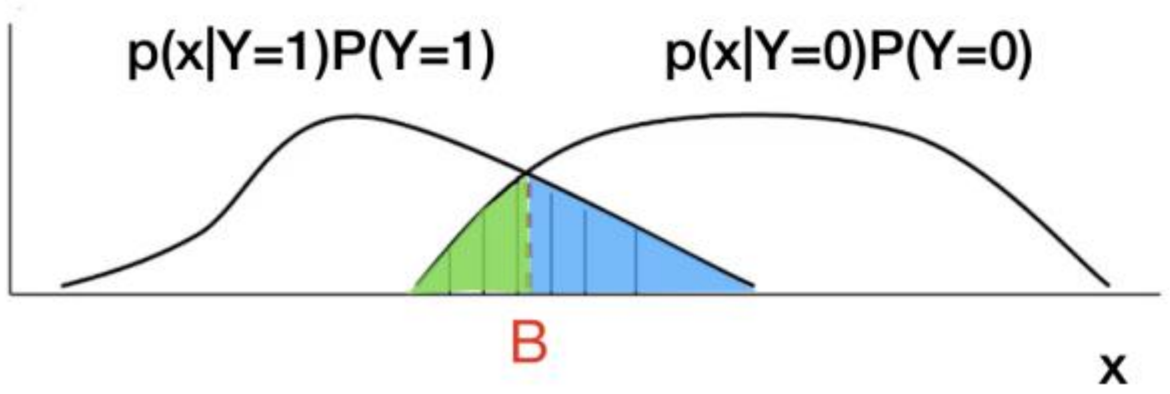


Every sample in blue shaded zone will be misclassified as blue class

Every sample in blue shaded zone will be misclassified as blue class

Bayes Error

- For classifiers, Bayes error is lower limit of error that one can get with any classifier
  - A classifier that achieves this error rate is an ***optimal*** classifier
  - Probability that we are *wrong*: this probability is the probability of the *less likely* label



$$P(Y=i|X=x) = \frac{p(x|Y=i)P(y=i)}{p(x)} = \frac{p(x|Y=i)P(Y=i)}{\sum_j p(x|Y=j)P(Y=j)}$$

Bayes Test: $p(x|Y=0)P(Y=0)$ and $p(x|Y=1)P(Y=1)$

Effect of Training Set Size

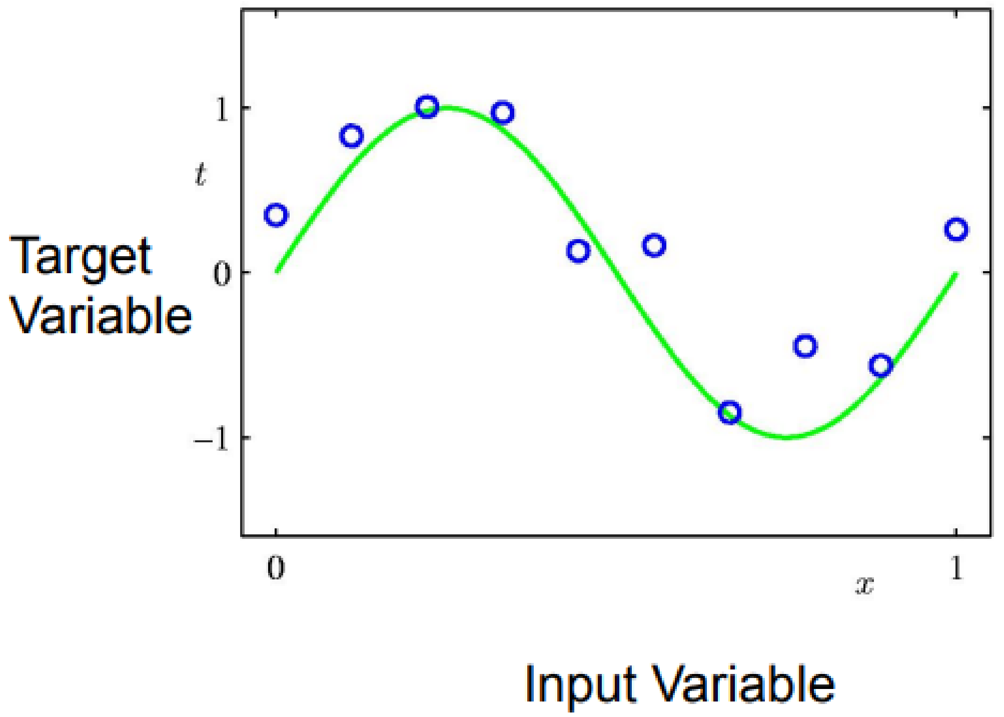
- Expected generalization error can never increase as the no of training examples increases
- For nonparametric models, more data yields better generalization until best possible error is achieved
- For any fixed parametric model with less than optimal capacity will asymptote to an error value that exceeds the Bayes error

For sufficiently large training set size n, the error rate of the 1NN classifier is less than twice the Bayes error rate, or $E^* > 2E$

THEORY THROUGH REGRESSION

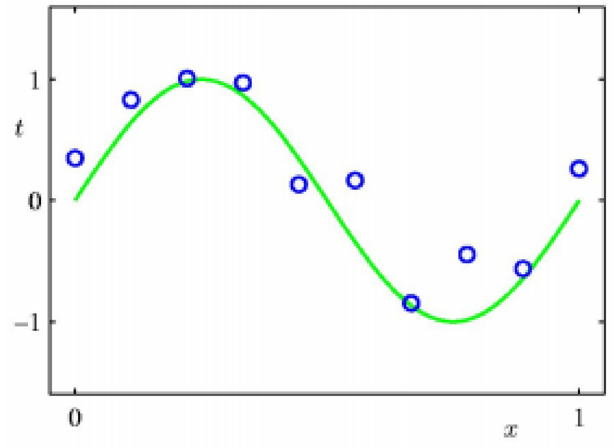
Synthetic Data for Regression

- Data generated from the function $\sin(2\pi\ x)$
  - Where x is the input
- Random noise in target values



Input values $\{x_n\}$ generated uniformly in range $(0,1)$. Corresponding target values $\{\mathrm{t_n}\}$ Obtained by first computing corresponding values of $\sin\{2\pi x\}$ then adding random noise with a Gaussian distribution with std dev 0.3

Training Set



- N observations of x

  $\mathrm{x} = (x_1,..,x_\mathrm{N})^\mathrm{T}$

  $\mathrm{t} = (t_1,..,t_\mathrm{N})^\mathrm{T}$
- Goal is to exploit training set to predict value $\hat{t}$ for some new value $\hat{x}$
- Inherently a difficult problem
- Probability theory provides framework for expressing uncertainty in a precise, quantitative manner
- Decision theory allows us to make a prediction that is optimal according to appropriate criteria

A Simple Approach to Curve Fitting

- Fit the data using a *polynomial function*

$$y(x, \mathrm{w}) = w_0 + w_1 x + w_2 x^2 + .. + w_M x^M = \sum_{j=0}^{M} w_j x^j$$

  - where M is the order of the polynomial
- Is higher value of M better? We' ll see shortly!
- Coefficients $w_0, \ldots w_M$ are collectively denoted by vector $\mathbf{w}$
- It is a nonlinear function of x, but a linear function of the unknown parameters
- Have important properties and are called Linear Models

Elemental Learning Theory (Wrap-up!)

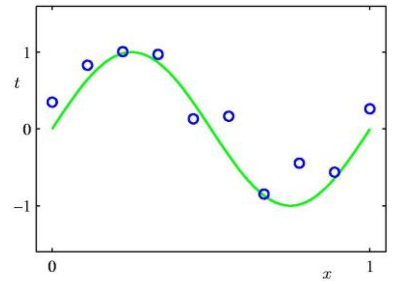
Alexander G. Ororbia II

Introduction to Machine Learning

CSCI-635

9/22/2023

Polynomial Curve Fitting with a Scalar



Training data set
N=10, Input x, target t

- With a <u>single</u> input variable x
- $y(x,\mathbf{w}) = w_0+w_1x+w_2x^2+\ldots+w_Mx^M = \sum_{j=0}^{M} w_j x^j$

  M = order of polynomial,

  x^j denotes x raised to power j,

  Coefficients $w_0,\ldots,w_M$ are collectively denoted by vector $\mathbf{w}$

- **Task**: Learn $\mathbf{w}$ from training data $D=\{(x_i,t_i)\}$, $i=1,..,N$
  - Can be done by minimizing an error function that minimizes misfit between $y(x,\mathbf{w})$ for any given $\mathbf{w}$ and training data
  - One simple choice of error function is sum of squares of error (*SSE*) between predictions $y(x_n,\mathrm{w})$ for each data point x_n and corresponding target values t_n so that we minimize: $E(\mathrm{w}) = \frac{1}{2}\sum_{n=1}^{N}\{y(x_n,\mathrm{w})-t_n\}^2$
  - It is zero when function $y(x,\mathrm{w})$ passes exactly through each training data point

On Basis Functions

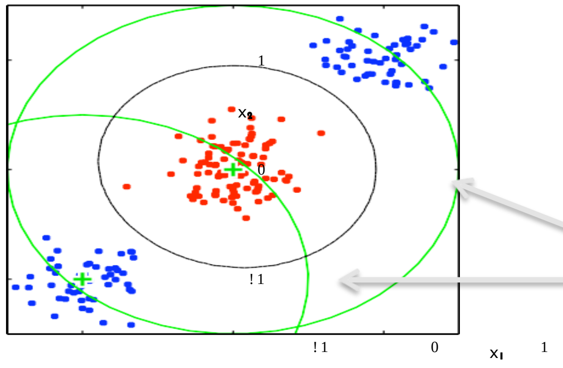
- In many applications, we apply some form of fixed-preprocessing, or feature extraction, to the original data variables
- If the original variables comprise the vector $\mathbf{x}$, then the features can be expressed in terms of basis functions $\{\phi_j(\mathbf{x})\}$
  - By using nonlinear basis functions we allow the function $y(\mathbf{x},\mathbf{w})$ to be a nonlinear function of the input vector $\mathbf{x}$
    - They are linear functions of parameters (gives them simple analytical properties), yet are nonlinear wrt input variables

Fixed Basis Functions

Although we use linear (classification) models

Linear-separability in ***feature*** space *does not* imply linear-separability in ***input*** space

Original Input Space (x_1, x_2)

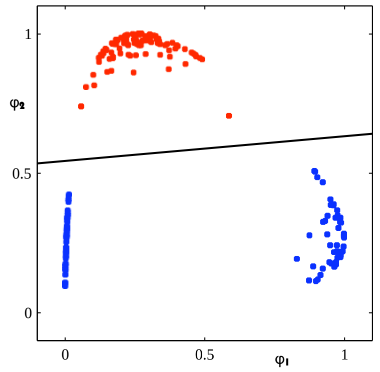


not linearly separable

Nonlinear transformation of inputs using vector of basis functions $\boldsymbol{\phi}(\boldsymbol{x})$

Basis functions are Gaussian with centers shown by crosses and green contours

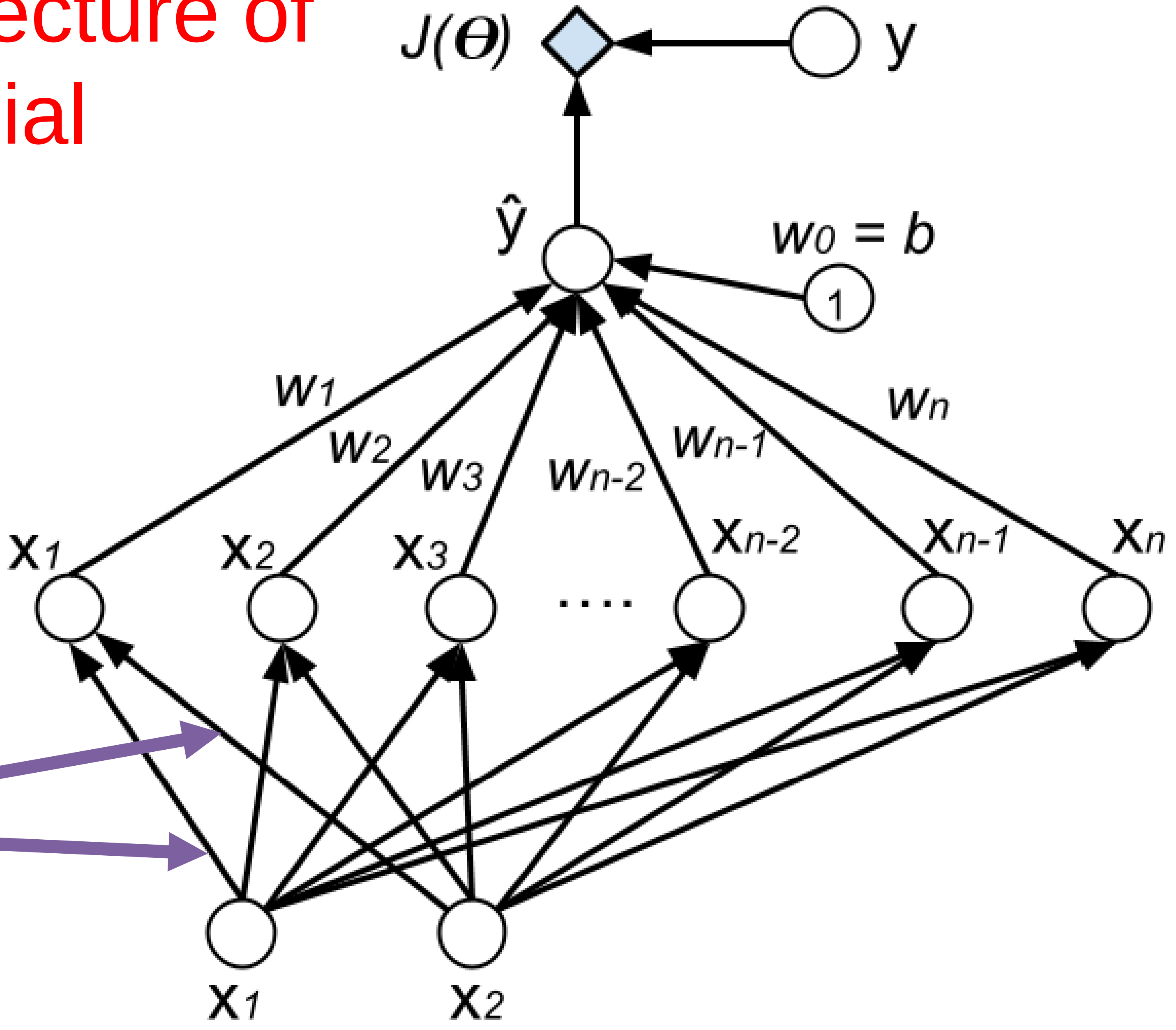
Feature Space (ϕ_1, ϕ_2)



linearly separable

Basis functions with increased dimensionality often used

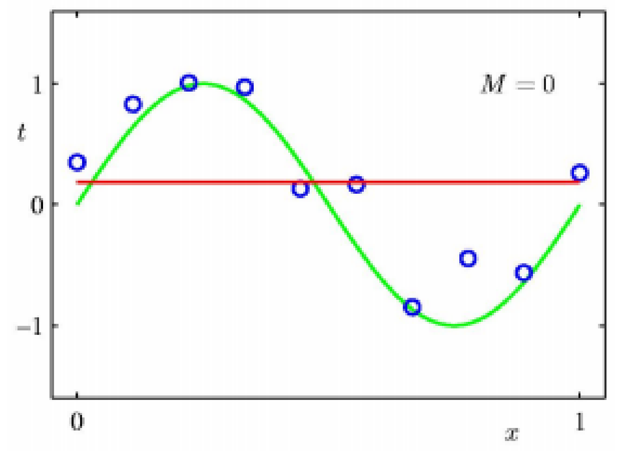
The Architecture of a Polynomial Regressor

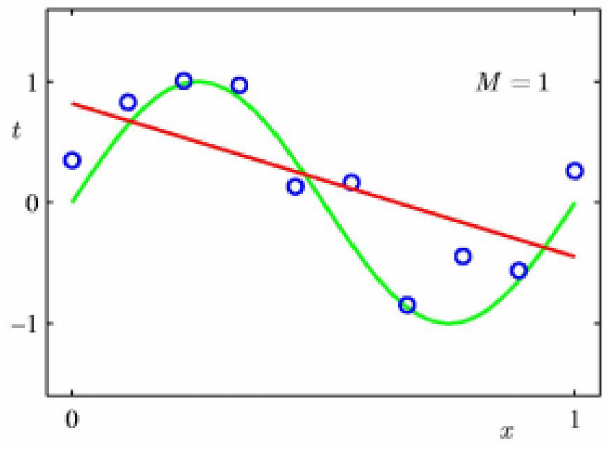


Basis function $\phi(x_j)$

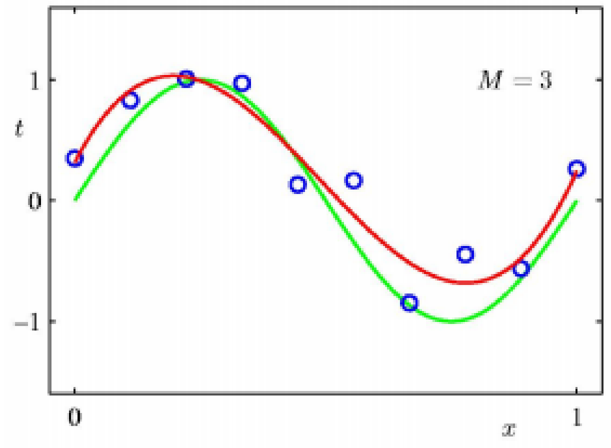
Choosing the Order of *M*

- Model Comparison or Model Selection
- Red lines are best fits with
  - $M = 0,1,3,9$ and N=10

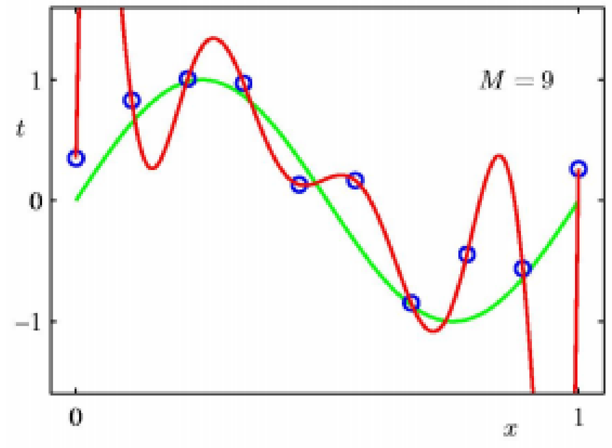




← Poor representations of $\sin(2\pi x)$



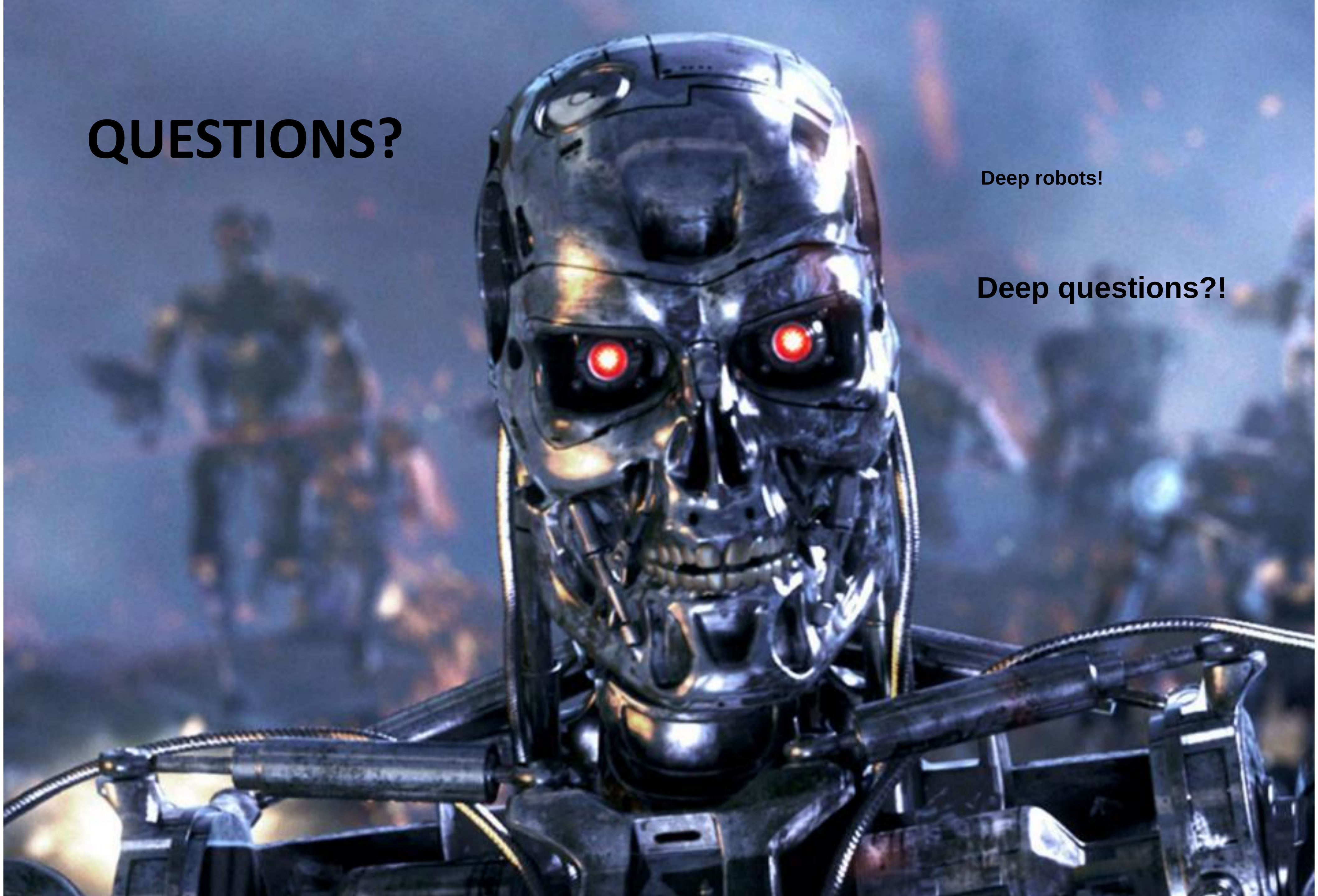
← Best Fit to $\sin(2\pi x)$



Over Fit Poor representation of $\sin(2\pi x)$

Computational Bottleneck

- A recurring problem in machine learning:
  - Large training sets are necessary for good generalization
  - *BUT* large training sets are also computationally expensive to use
- Stochastic gradient descent (SGD) is an extension of gradient descent (GD) that offers a solution
  - Moreover, it is a vehicle for generalization beyond training set
  - Expectation may be approximated using small set of samples (we will also later refer to these sets as “mini-batches” $\rightarrow$ mini-batch GD)



QUESTIONS?

Deep robots!

Deep questions?!