More Elemental Learning Theory

Alexander G. Ororbia II

Introduction to Machine Learning

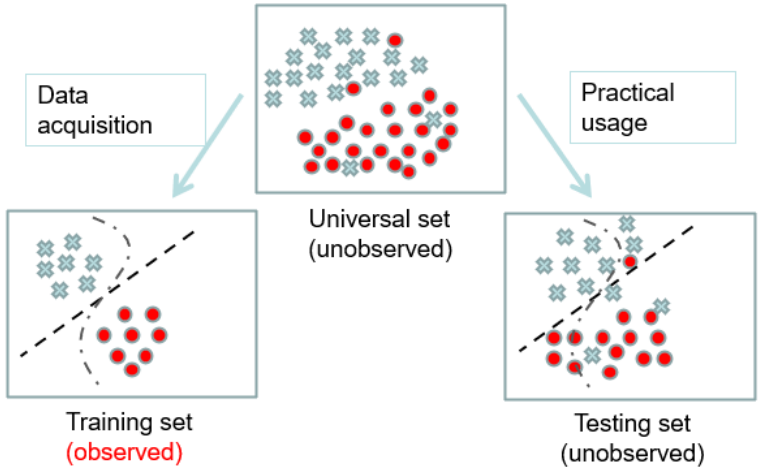
CSCI-335

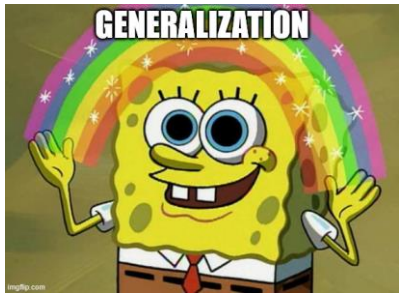
2/16/2026

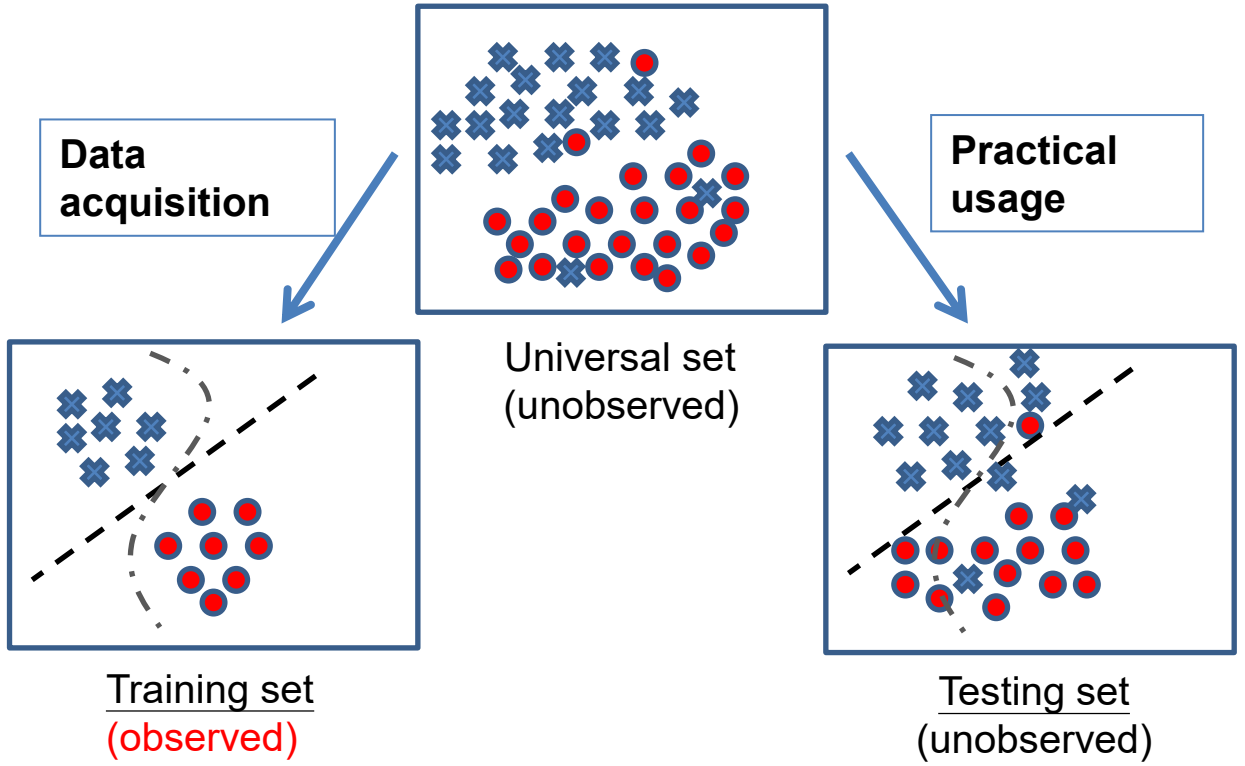
Generalization

Resting on the *IID* assumption!

- Challenge of ML is generalization
  - Perform well on previously unseen outputs
- ML training algorithm reduces training error, which is a task of optimization
- What differentiates ML from optimization is that we want test error (generalization error) low as well
- Generalization error definition
  - Expected error on a new input
  - Expectation wrt distribution encountered in practice







Training = process of making the system able to learn

Testing = process of seeing how well the system learned

- Simulates "real world" usage
- Training set and testing set should come from same distribution
- Need to make some **assumptions** or introduce a bias

Deployment = actually using the learned system in practice

The Machine Learning Task Description

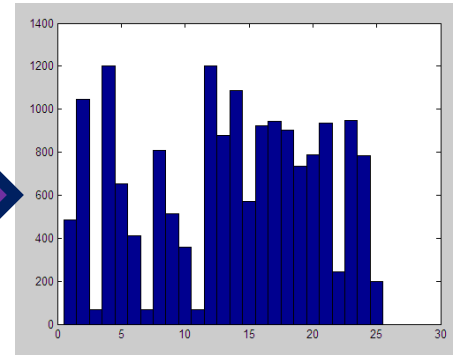
- Usually described in terms of how the machine learning system should process an example
- An example is a collection of features that have been quantitatively measured for some object/ event that we want the ML system to process
- Typically represent an example as a vector $\boldsymbol{x}$ where each entry x_n of the vector is another feature

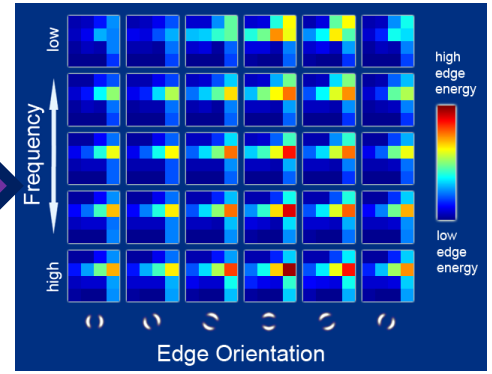
A dataset → $\{x_n \in R^d\}_{n=1}^N$

Features (What Could be "Inside" of x)

- Raw pixels
- Histograms
- GIST descriptors
- …

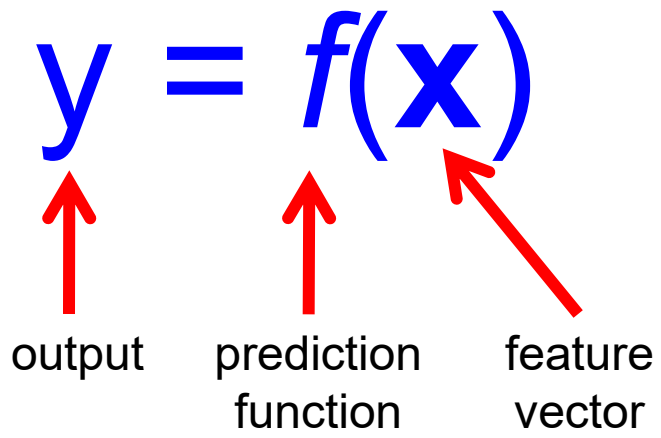
Sensor(y) representations (e.g., images, derived features)





Process of crafting input sensor features = ***feature engineering***

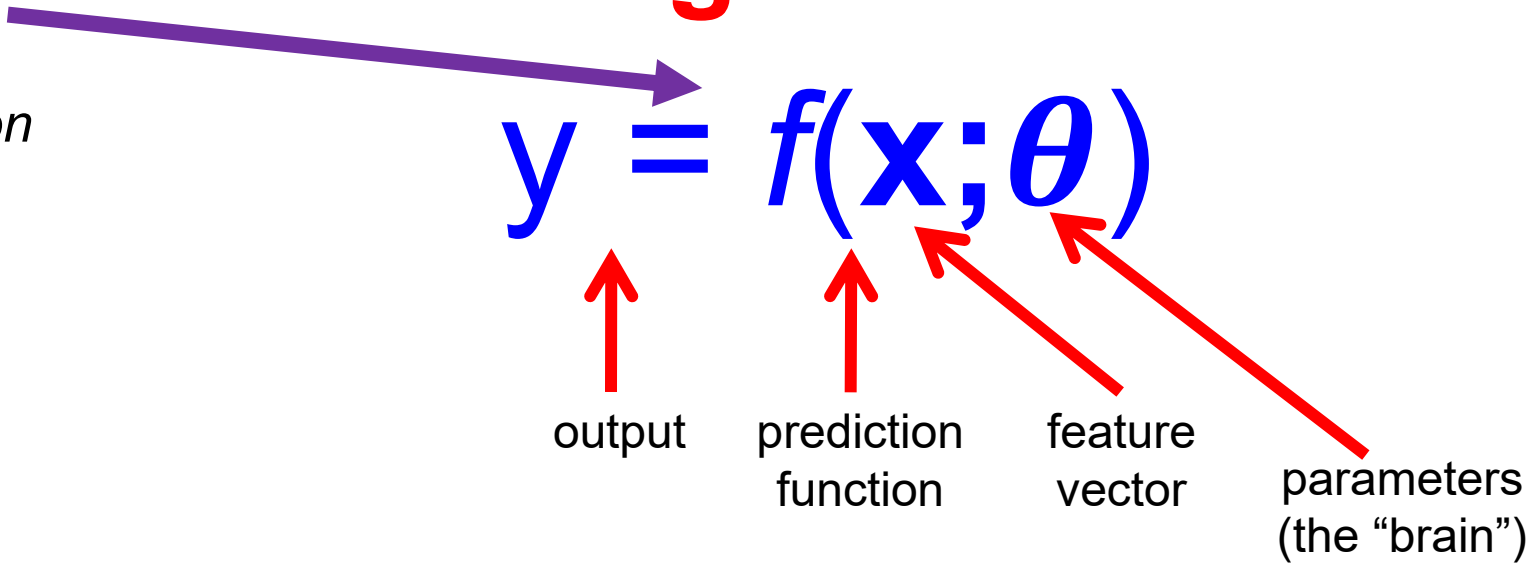
The Functional Machine Learning Framework



- **Training:** given a *training set* of labeled examples $\{(\mathbf{x}_1,y_1), \ldots, (\mathbf{x}_N,y_N)\}$, estimate the prediction function f by minimizing the prediction error on the training set
- **Testing:** apply *f* to a never-before-seen *test example* **x** and output the predicted value $y = f(\mathbf{x})$

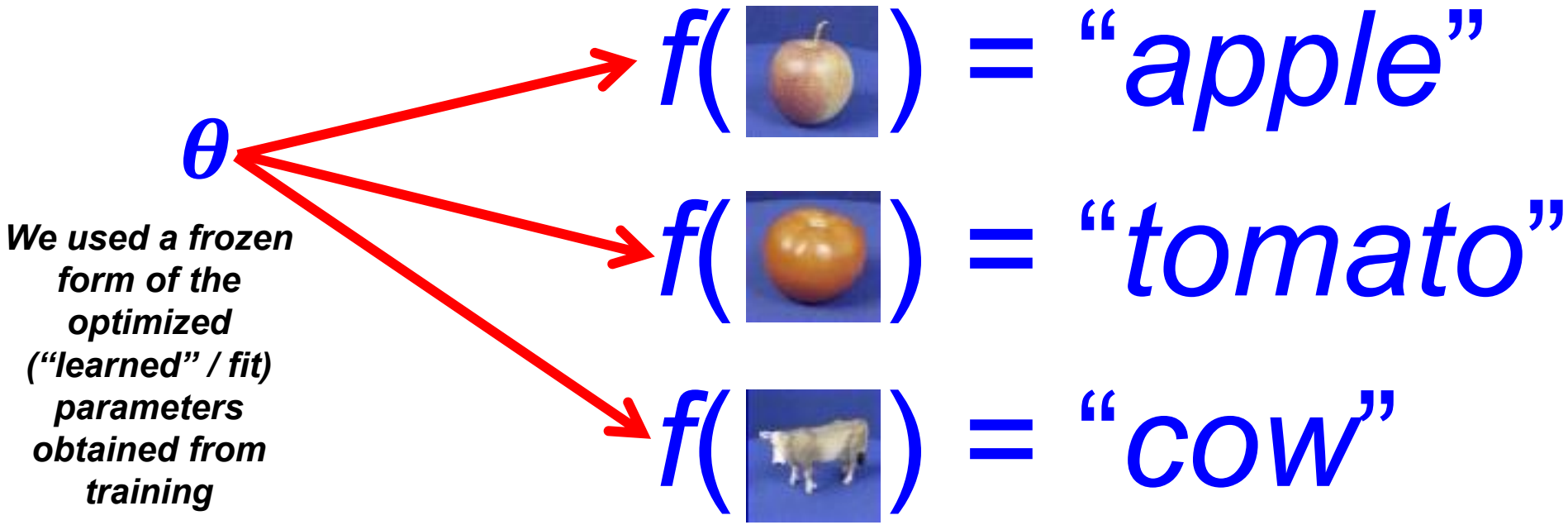
The Functional Machine Learning Framework

We are engaged in a form of *function approximation*



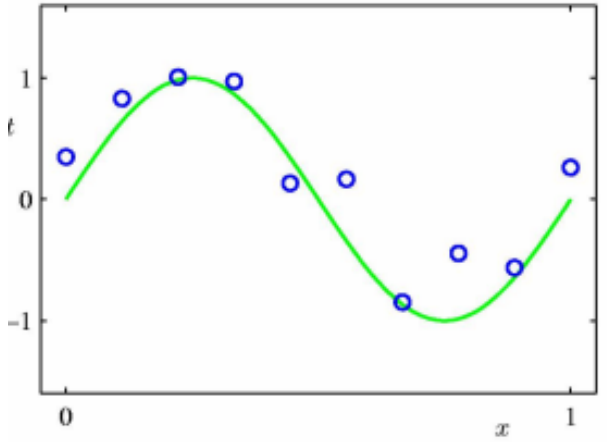
Note that this is a *parametric* form of learning (as opposed to *non-parametric* learning)

- ***Test-time inference***:
  apply a prediction function to a feature representation of the image to get the desired output:

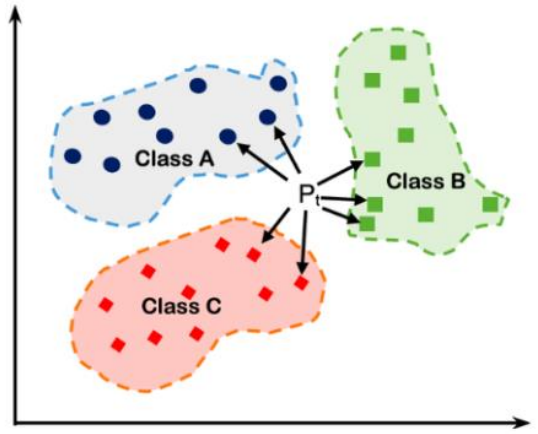


Parametric vs. Non-Parametric Functional Forms

- ***Parametric models***
  - Assume fixed, predefined function(al) form under a fixed / finite quantity of parameters Θ
  - Good for scarce, low-sample-size data or if structure/distribution is known (and simple) → make assumptions about data distribution
  - Regression, discriminant analysis, Bayes models
- ***Non-parametric models***
  - Make no assumptions about data distribution (more flexible), complexity grows with dataset size
  - Good for unknown, complex data with nonlinear structure (& possibly dealing with outliers)
  - K-nearest neighbors, decision trees (and ensembles), kernel-based support vector machines



Polynomial regression



K-nearest neighbors

Goal: Construct a Pipeline

Training

Training Images

Image Features

Training Labels

Training

θ

Optimized ("learned" / fit) parameters

Learned model

$y = f(\mathbf{x};\theta)$

Testing

Test Image

Image Features

Learned model

Prediction

Slide credit: D. Hoiem and L. Lazebnik

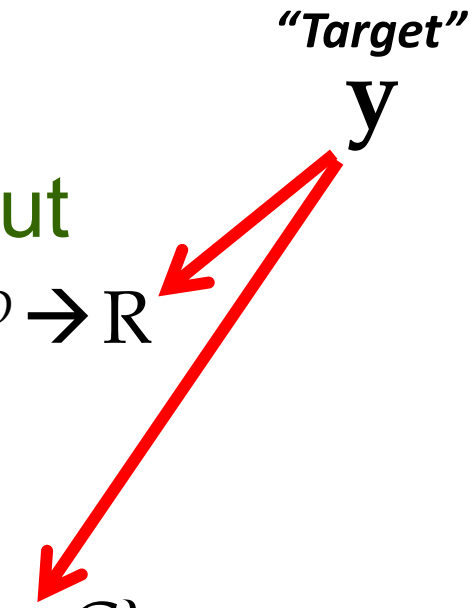
Learning Algorithm *A*

- Uses training values for the target function ***f*** to induce a hypothesized definition that fits these examples and hopefully generalizes to unseen examples

- In statistics, learning to approximate a:
  - Continuous function = ***regression***
  - Discrete function = ***classification*** (or categorization)

- Attempts to minimize some measure of error (***loss function***), such as ***mean squared error***

ML Terminology for Algorithm *A*



- *Regression*
  - Predict a numerical value $\mathbf{y}$ given some input
    - Learning algorithm has to output function $f: \mathrm{R}^D \rightarrow \mathrm{R}$
      - where D =number of input (decision) variables
- *Classification*
  - If $\mathbf{y}$ value is a label (categories): $f: \mathrm{R}^D \rightarrow \{1,..,C\}$
- *Ordinal Regression*
  - Discrete values, ordered categories
    - (We will not be covering this in this course!)

Evaluation of Learning Systems

- ***Experimental***
  - Conduct controlled cross-validation experiments to compare various methods on variety of benchmark datasets
  - Gather data on performance, e.g. test accuracy, training-time, testing-time
  - Analyze differences for statistical significance (frequentist measures)

- ***Theoretical***
  - Analyze algorithms mathematically and prove theorems about their:
    - Computational complexity
    - Ability to fit training data
    - Sample complexity (number of training examples needed to learn an accurate function)
  - ***Usefulness***:
    provides intellectual justification that ML works, but rarely used in practice b/c:
    - Bounds are loose
    - Hard to determine “capacity” of some ML algorithms

Factors of Generalization

- Components of *generalization error*
  - **Bias:** how much the average model over all training sets differ from the true model?
    - Error due to inaccurate assumptions/simplifications made by the model
  - **Variance:** how much models estimated from different training sets differ from each other
  - **(Irreducible) Noise**

Bias-Variance Tradeoff

- **Underfitting:** model is too "simple" to represent all the relevant class characteristics
  - High bias and low variance
  - High training error and high test error

You did bad!

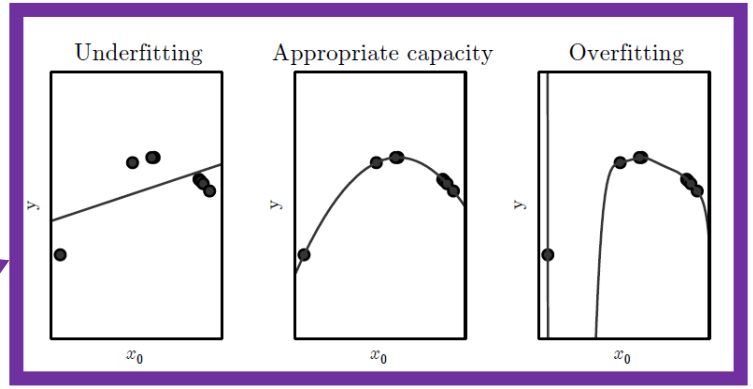
- **Overfitting:** model is too "complex" and fits irrelevant characteristics (noise) in the data
  - Low bias and high variance
  - Low training error and high test error

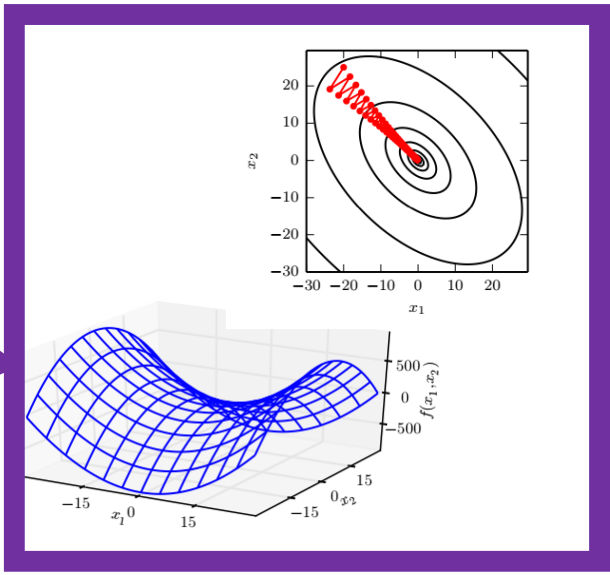
You tried too hard!

f() = "*apple*"
f() = "*tomato*"
f() = "*cow*"

On (Model) 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

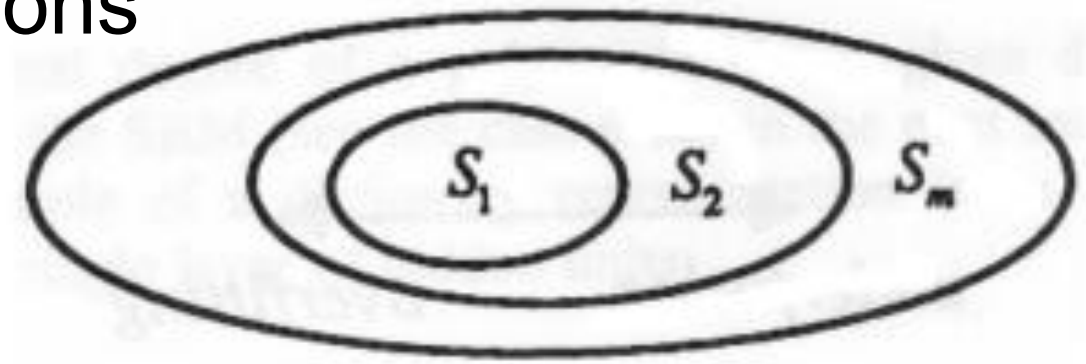




Some Basic Statistical Learning Theory

- ***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 → ***??***

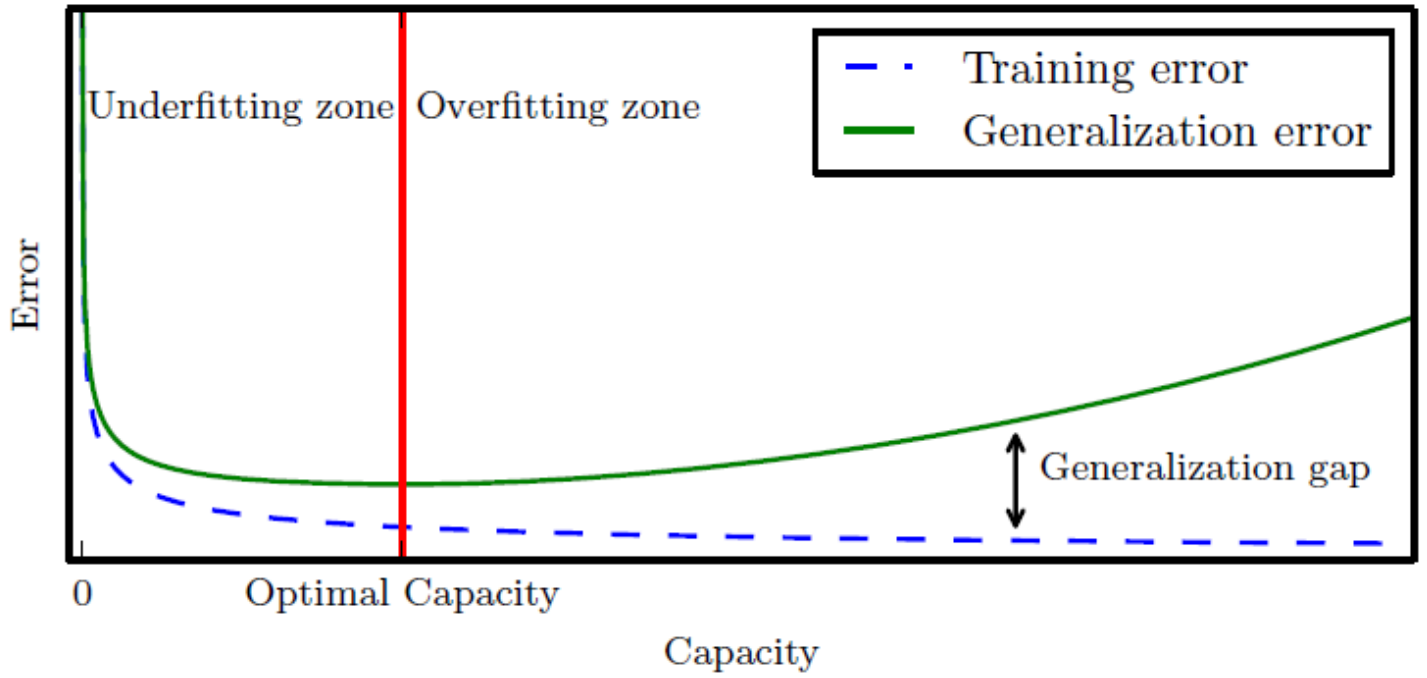


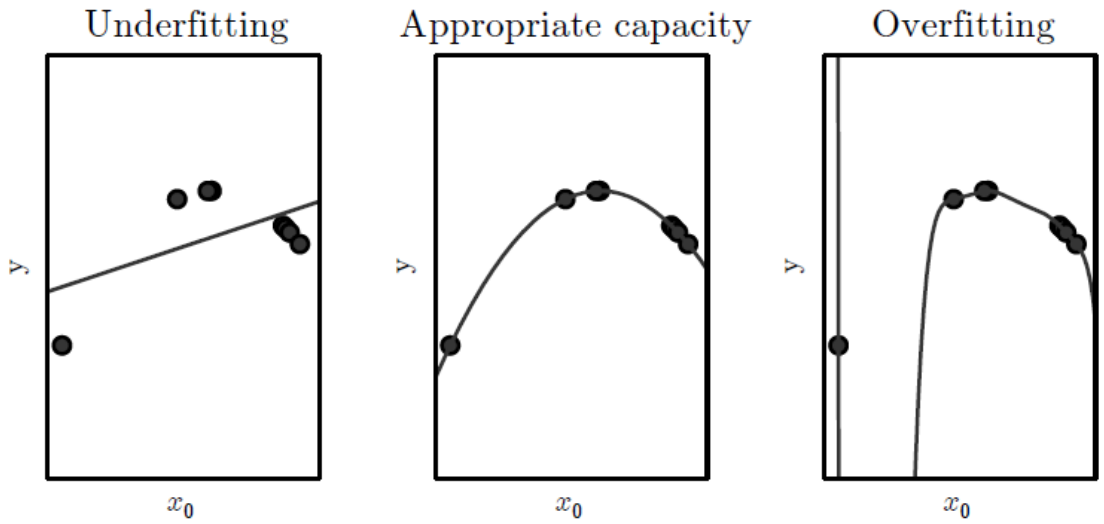
Generalization and Capacity

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





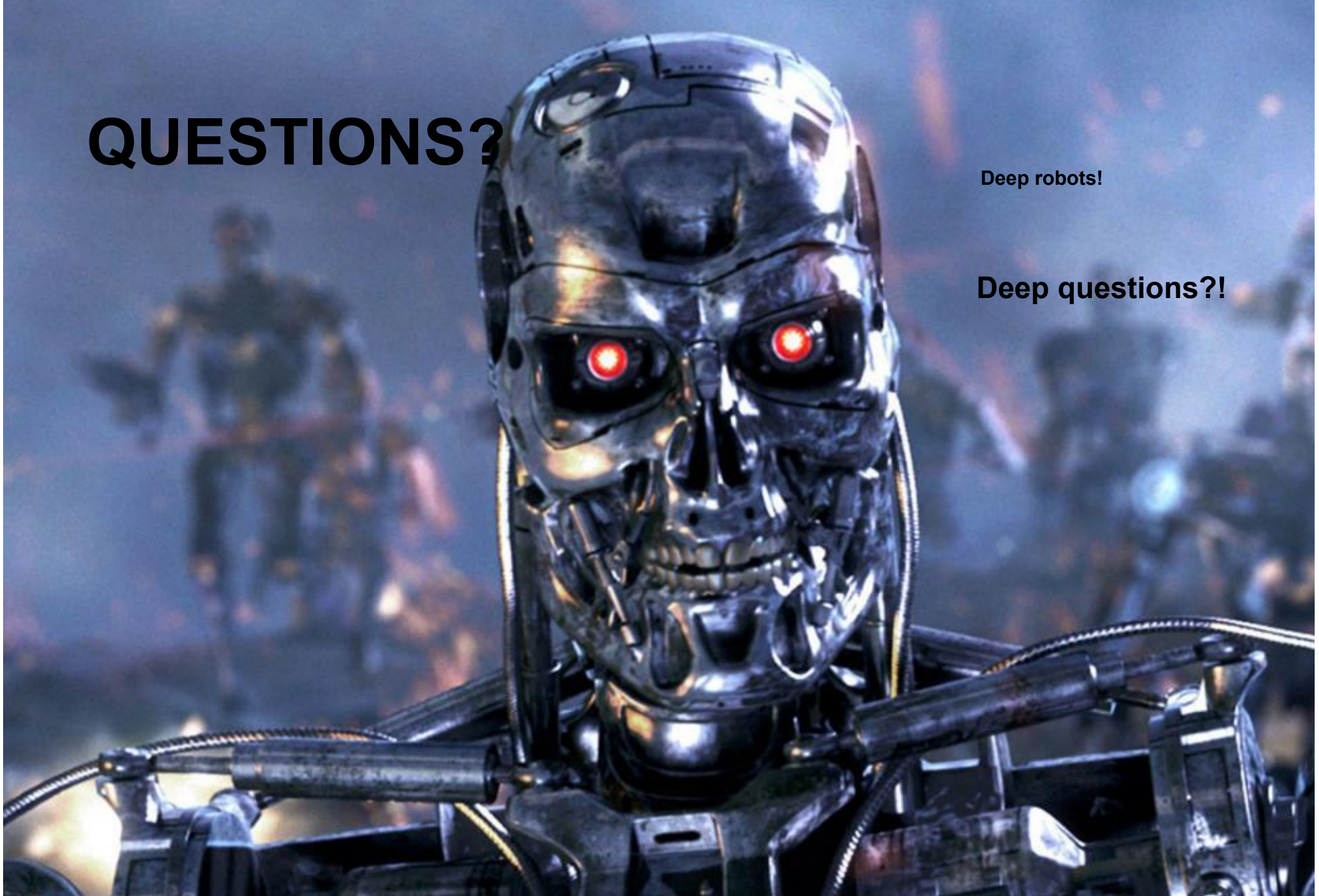
The Fundamental Principle of Statistical Learning

- 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??

- **More sophisticated theory:**
  - Vapnik-Chervonenkis (VC) Dimension
  - Probably Approximately Correct (PAC) Framework


QUESTIONS?
Deep robots!
Deep questions?!