Elemental Learning Theory

Alexander G. Ororbia II

Introduction to Machine Learning

CSCI-335

9/21/2026

ML in a Nutshell: Core Ideas

Representation (Modeling)

Data format, organization

Model structure, “architecture”

Evaluation

“Goodness” of model on data sample

Guides optimization / model fitting

Optimization

“Model fitting” → evolution/adjustment of parameters, error correction

Learning Algorithms

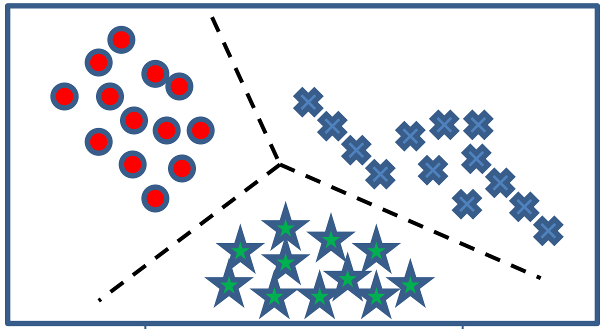
- Definition (well-posed learning problem):
  - A computer program is said to learn from *experience* E
  - with respect to some *class of tasks* T and *performance measure* P,
  - if its performance at task T, as measured by P, improves with experience E

Types of Learning

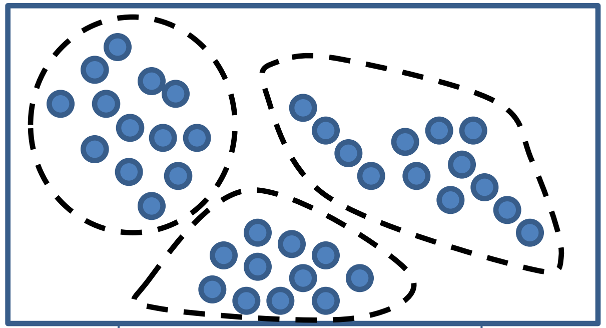
- **Supervised (inductive) learning** $\{\mathbf{x}_n \in \mathbb{R}^D, \mathbf{y}_n \in \mathbb{R}^C\}_{n=1}^N$
  - Training data includes desired outputs, $(\mathbf{x}_i, \mathbf{y}_i = f(\mathbf{x}_i))$
  - Prediction / Classification (discrete labels), Regression (real values)
- **Unsupervised learning** $\{\mathbf{x}_n \in \mathbb{R}^D\}_{n=1}^N$
  - Training data does not include desired outputs
  - Clustering / probability distribution estimation
  - Finding association (in features)
  - Dimension reduction
- **Semi-supervised learning** $\{\mathbf{x}_n \in \mathbb{R}^D, \mathbf{y}_n \in \mathbb{R}^C\}_{n=1}^N \cup \{\mathbf{x}_m \in \mathbb{R}^D\}_{m=1}^M, M \gg N$
  - Training data includes a few desired outputs
- **Reinforcement learning** $\{\mathbf{x}_t \in \mathbb{R}^D, r(\mathbf{x}_{t+1}, a_t) \in \mathbb{R}\}_{t=1}^{T=\infty}$
  - Rewards from sequence of actions
  - Decision making (robot, chess machine)

Reward function

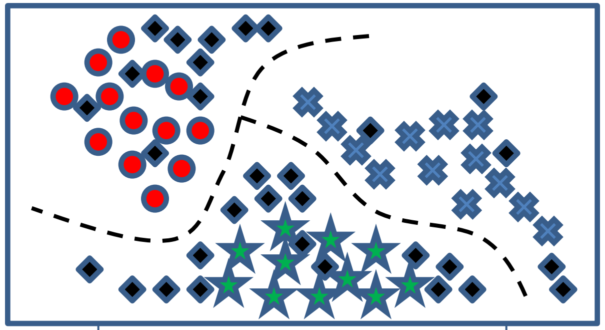
Visualizing the Types of Learning



Supervised learning



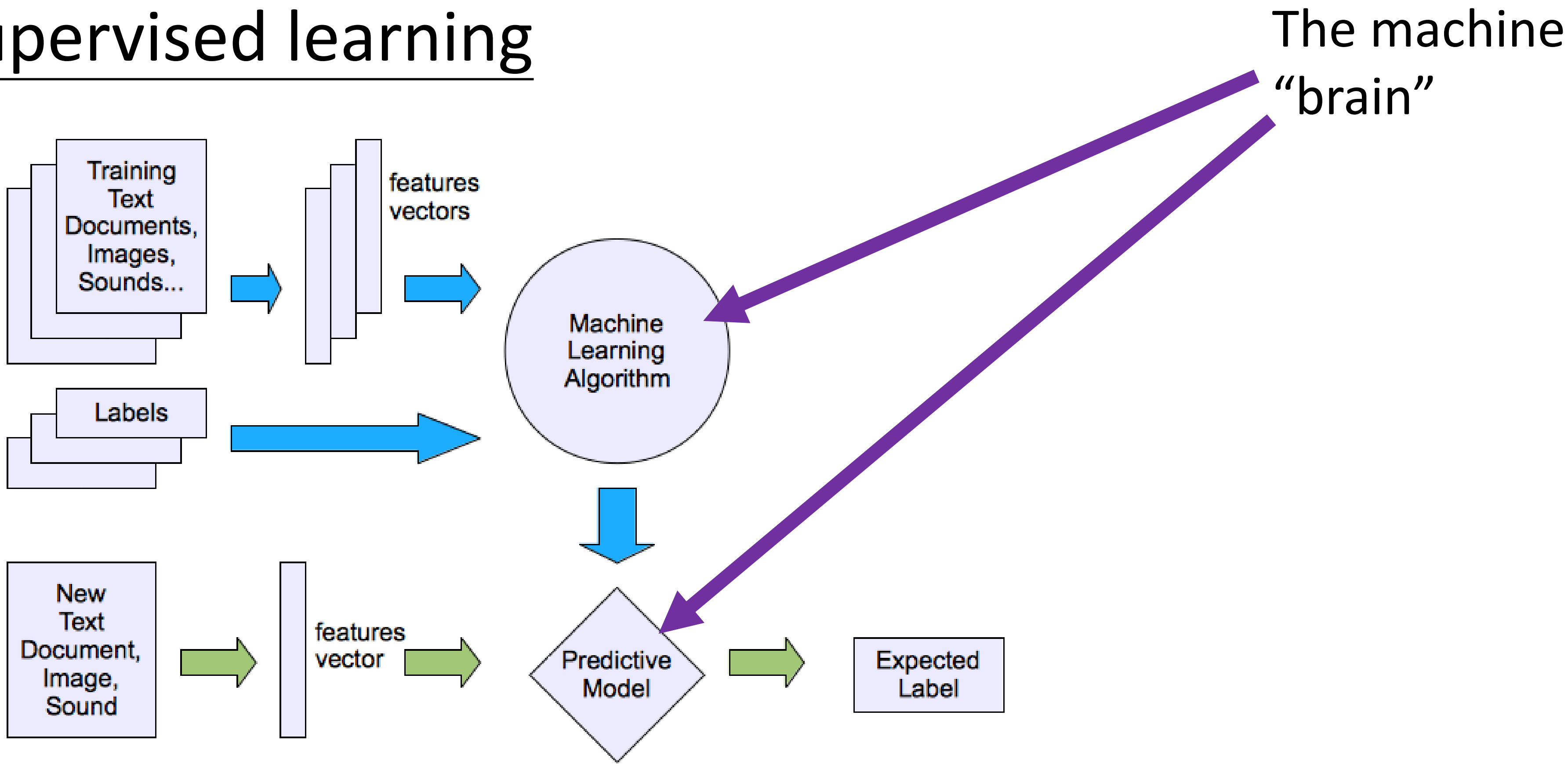
Unsupervised learning



Semi-supervised learning

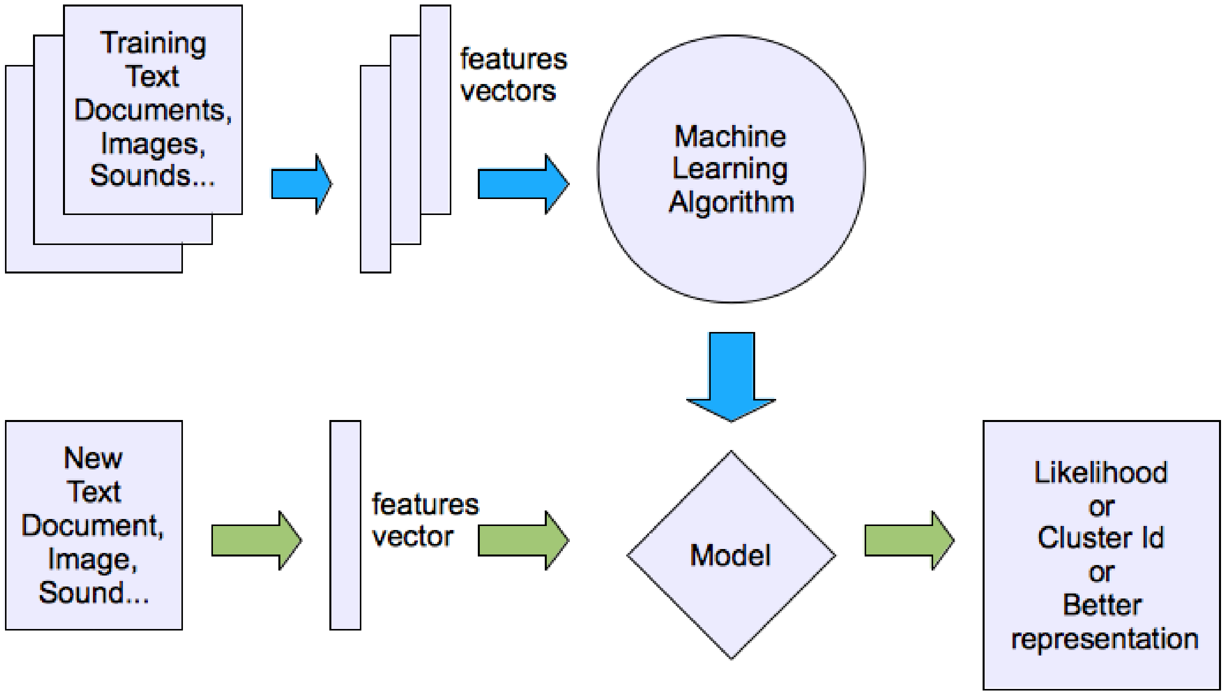
A Typical Supervised Learning Pipeline

- <u>Supervised learning</u>



A Typical Unsupervised Learning Pipeline

- Unsupervised learning



Statistical Learning in Practice

- Understanding domain, prior knowledge, and goals
- Data integration, selection, cleaning, pre-processing, etc.
- Learning model(s)
- Interpreting results
- Consolidating and deploying discovered knowledge
- ***Loop***

Statistical Learning in Practice

The science of the problem

- Understanding domain, prior knowledge, and goals
- Data integration, selection, cleaning, pre-processing, etc.
- Learning model(s)
- Interpreting results
- Consolidating and deploying discovered knowledge
- ***Loop***

CSCI 335
(***You are here!***)

What big data/infrastructure courses help with

Defining the *problem* helps with this!

Industry dictates this
(*machine learning engineering*)

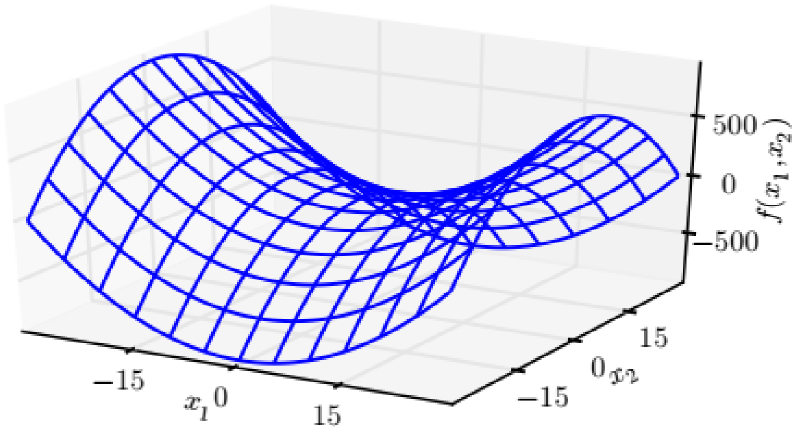
Training Experience

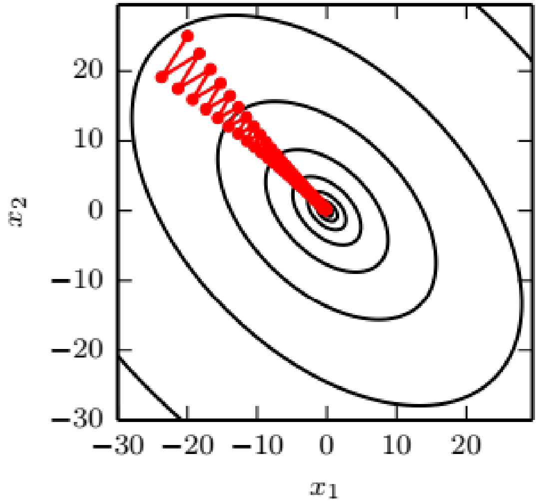
- Direct experience: Given sample input and output pairs for a useful target function
  - Checker boards labeled with the correct move, e.g. extracted from record of expert play
- Indirect experience: Given feedback which is ***not*** direct I/O pairs for a useful target function
  - Potentially arbitrary sequences of game moves and their final game results (i.e., a score or cumulative function output)
- Credit/Blame Assignment Problem: How to assign credit or blame to individual moves given only indirect feedback?
  - The **problem of credit assignment** = appears everywhere in statistical learning

So, Statistical Machine Learning is…

- ***Three Pillars:***
  - Representation
  - Evaluation
  - Optimization

It is not only <u>just</u> fitting/optimization…



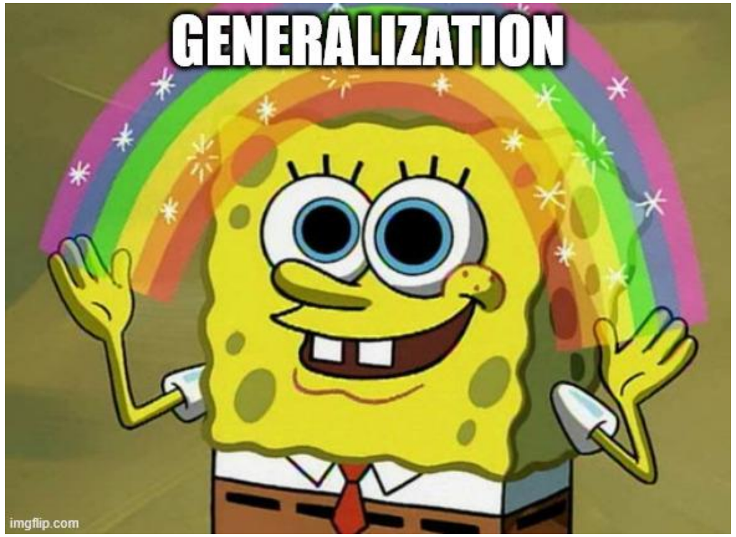


Key Point:

…it is also about generalization!

(the three pillars are meant to support this one goal)

So,…then what is <u>generalization</u>?

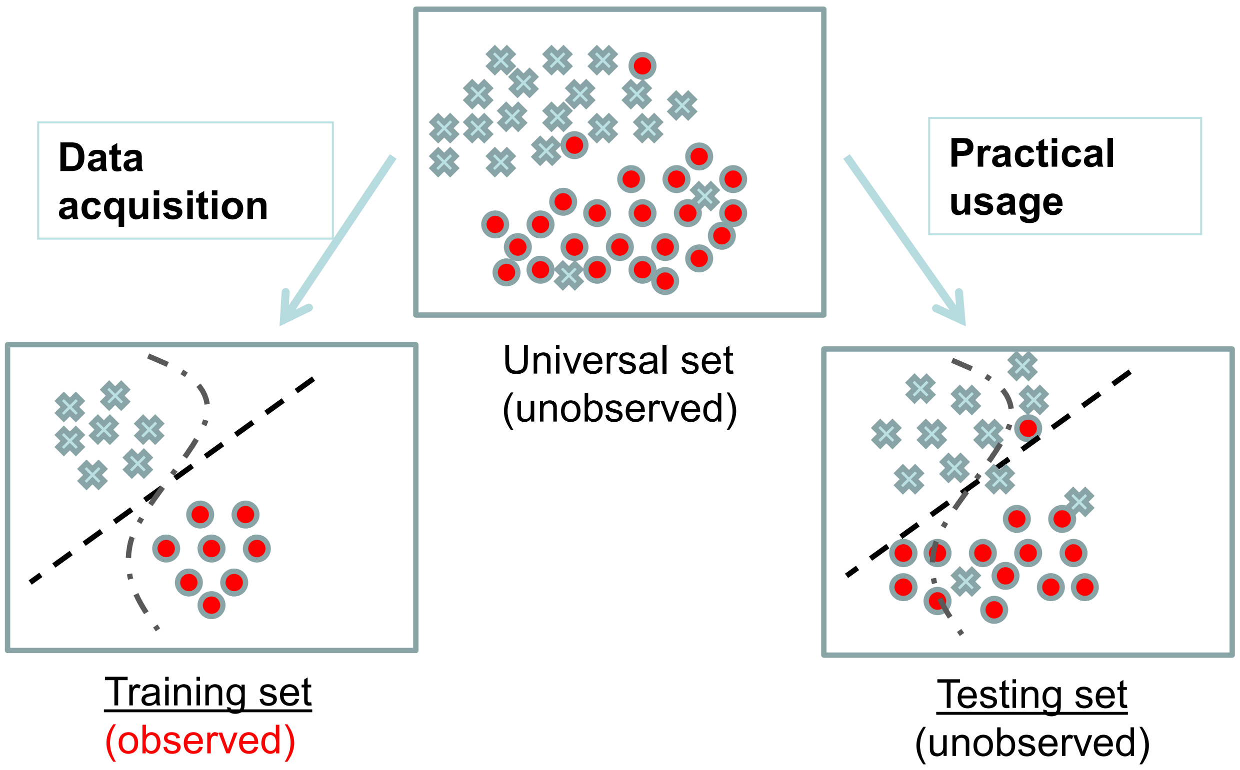


Training *vs.* Test Distribution

- *Generally,* assume that the training and test examples are independently drawn from the same overall distribution of data
  - **IID**: **Independently** and **identically distributed**
- If test distribution is different, requires *transfer learning*

<u>At the core of the assumption</u>:
A collection of random variates *must* fall under same probability distribution *and* are mutually independent

Identical = no overall trends/fluctuations in (same) distribution of collected objects
Independent = collected objects are all independent events; value of one item gives no knowledge about values of others (& vice versa)

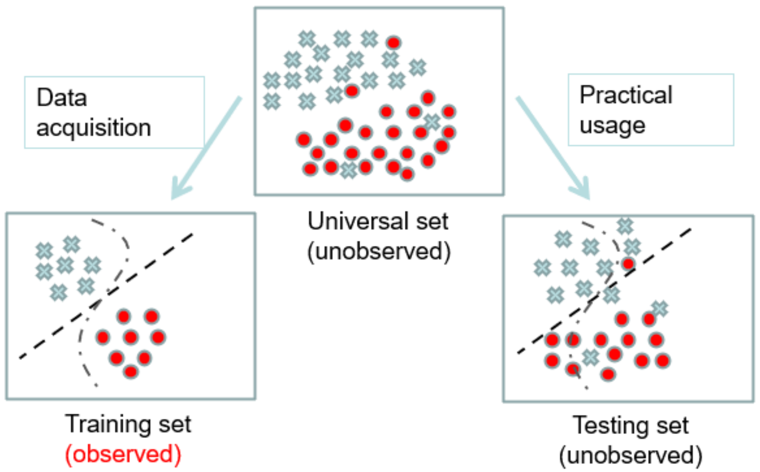


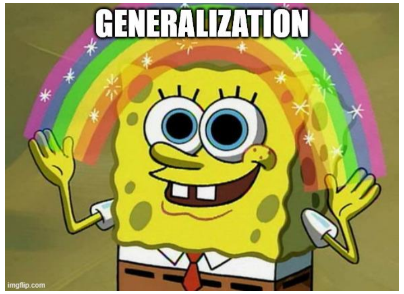
- ***Training*** = process of making the system able to learn
- ***Testing*** = process of seeing how well the system learned
  - Simulates “real world” usage
  - Training set and testing set should come from same distribution
  - Need to make some **assumptions** or introduce a bias
- ***Deployment*** = actually using the learned system in practice

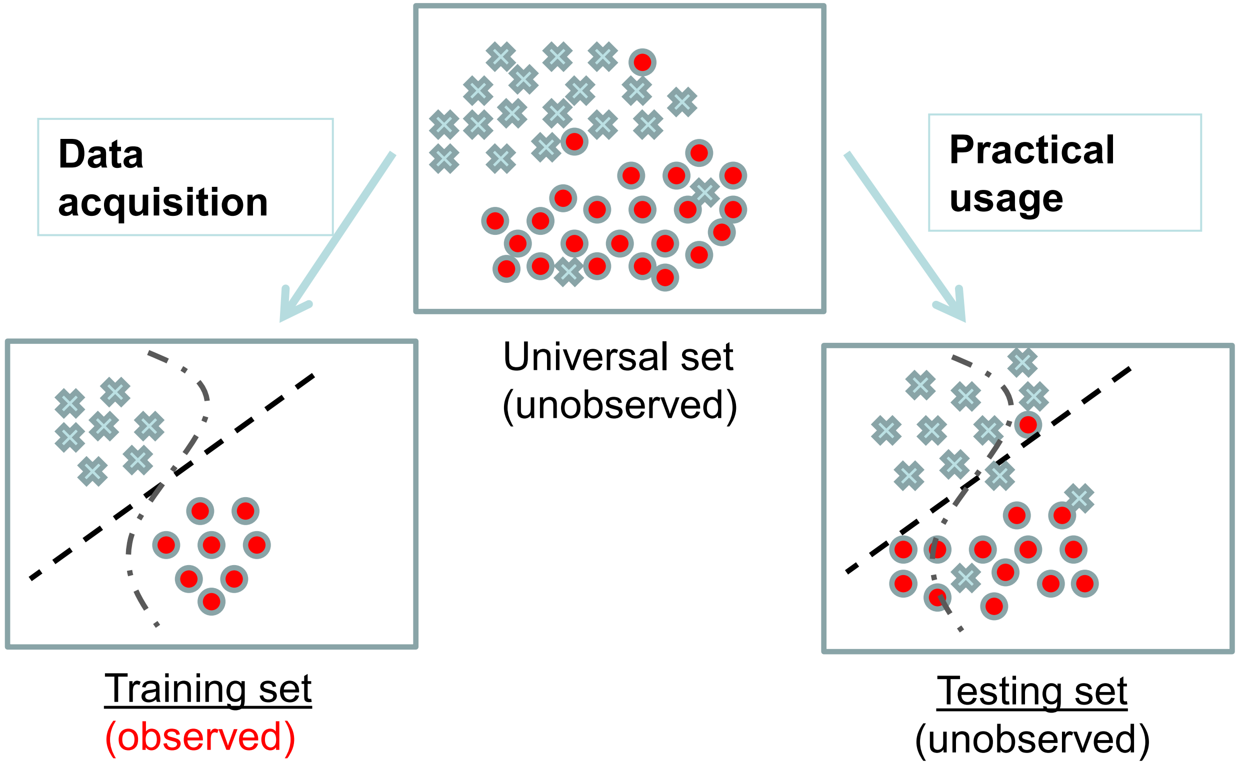
Generalization

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

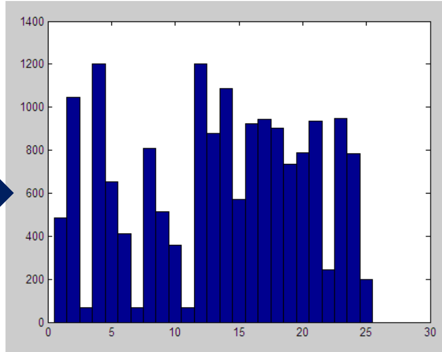
- Usually described in terms of how the machine learning system should process an example
- An example is a collection of features that have been quantitatively measured for some object/ event that we want the ML system to process
- Typically represent an example as a vector $\boldsymbol{x}$ where each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

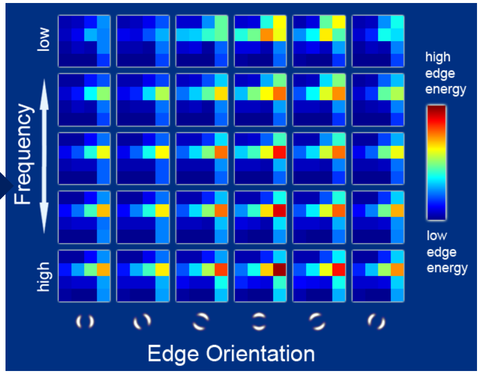
A dataset → $\{x_n \in R^d\}_{n=1}^N$

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

$$\mathrm{y} = f(\mathbf{x})$$

output

prediction function

feature vector

- **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* $\mathbf{x}$ and output the predicted value $y = f(\mathbf{x})$

The *Functional* Machine Learning Framework

We are engaged in a form of *function approximation*

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

output

prediction function

feature vector

parameters (the "brain")

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

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

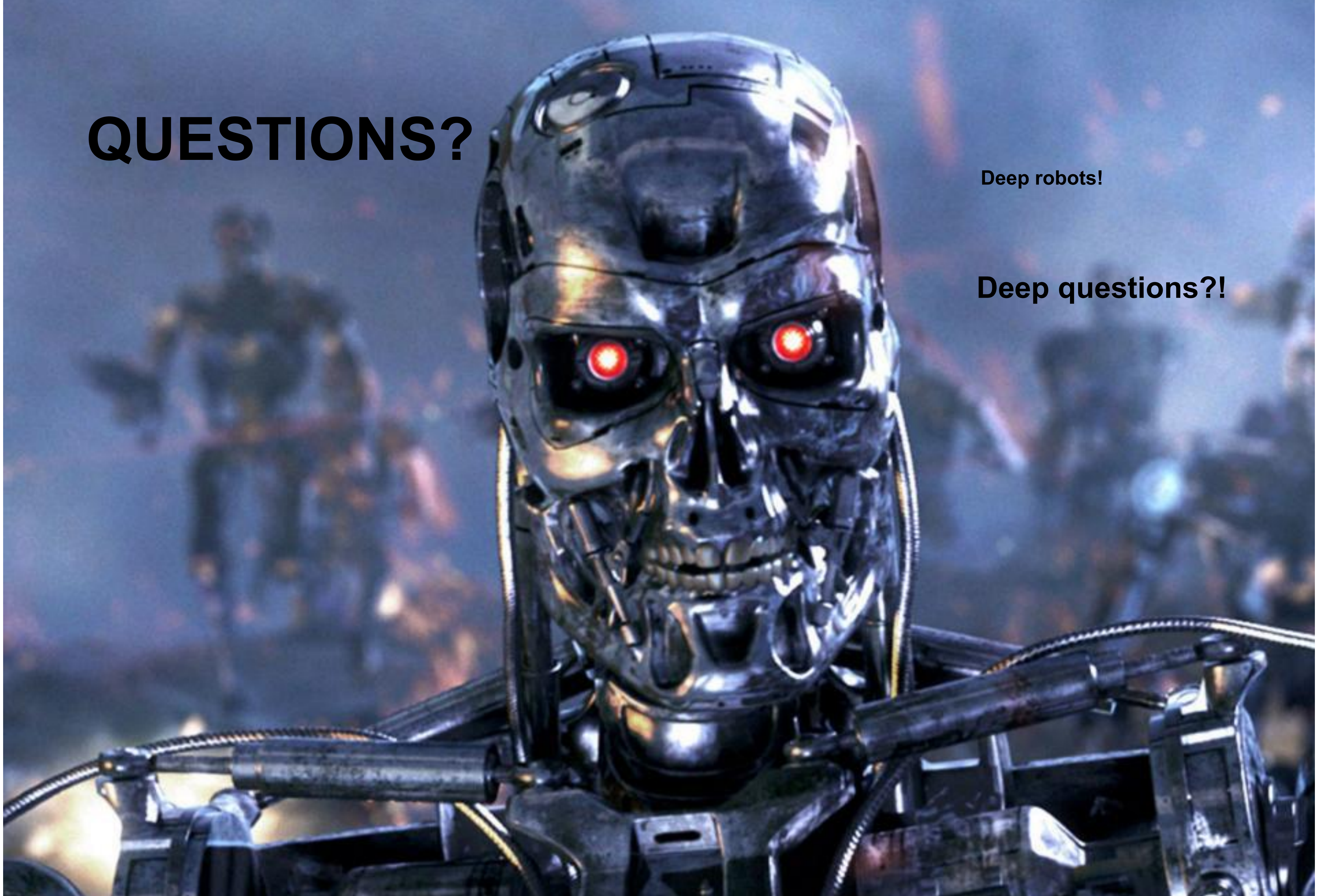
θ

We used a frozen form of the optimized ("learned" / fit) parameters obtained from training

f() = "*apple*"

f() = "*tomato*"

f() = "*cow*"


QUESTIONS?
Deep robots!
Deep questions?!