Elemental Learning Theory

Alexander G. Ororbia II

Introduction to Machine Learning

CSCI-635

9/13/2023

Example: Linear Least Squares

Suppose we want to find the value of $\boldsymbol{x}$ that minimizes

$$f(\boldsymbol{x}) = \frac{1}{2}||\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}||_2^2.$$

There are specialized linear algebra algorithms that can solve this problem efficiently. However, we can also explore how to solve it using gradient-based optimization as a simple example of how these techniques work.

First, we need to obtain the gradient:

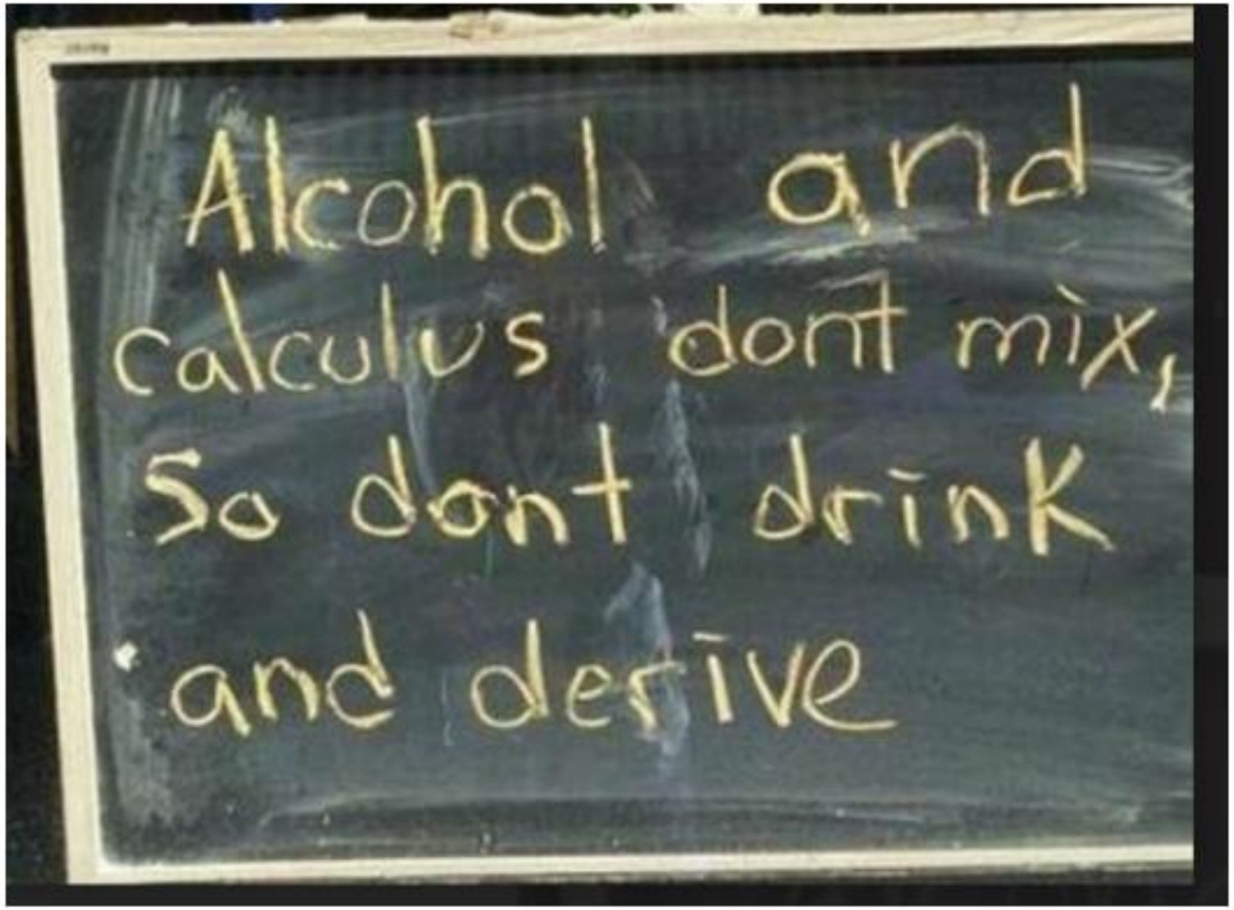
$$\nabla_{\boldsymbol{x}} f(\boldsymbol{x}) = \boldsymbol{A}^\top(\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}) = \boldsymbol{A}^\top\boldsymbol{A}\boldsymbol{x} - \boldsymbol{A}^\top\boldsymbol{b}.$$

Algorithm 4.1 An algorithm to minimize $f(\boldsymbol{x}) = \frac{1}{2}||\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}||_2^2$ with respect to $\boldsymbol{x}$ using gradient descent, starting from an arbitrary value of $\boldsymbol{x}$.

Set the step size (ϵ) and tolerance (δ) to small, positive numbers.
while $||\boldsymbol{A}^\top\boldsymbol{A}\boldsymbol{x} - \boldsymbol{A}^\top\boldsymbol{b}||_2 > \delta$ **do**
  $\boldsymbol{x} \leftarrow \boldsymbol{x} - \epsilon\left(\boldsymbol{A}^\top\boldsymbol{A}\boldsymbol{x} - \boldsymbol{A}^\top\boldsymbol{b}\right)$
end while

Gradient

- Essential role of calculus



ML in a Nutshell: Key Ideas

Representation / Modeling

Data format, organization

Model structure, “architecture”

Evaluation

“Goodness” of model on data sample

Guides optimization / model fitting

Optimization

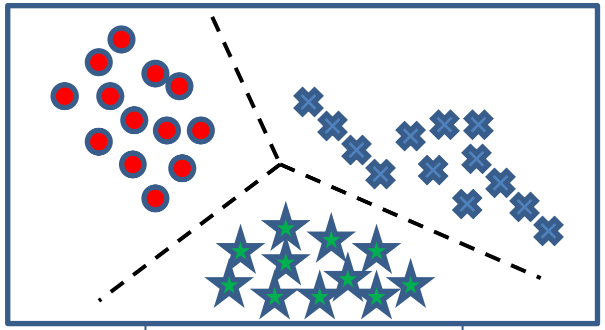
“Model fitting” -- evolution/adjustment of parameters, error correction

Types of Learning

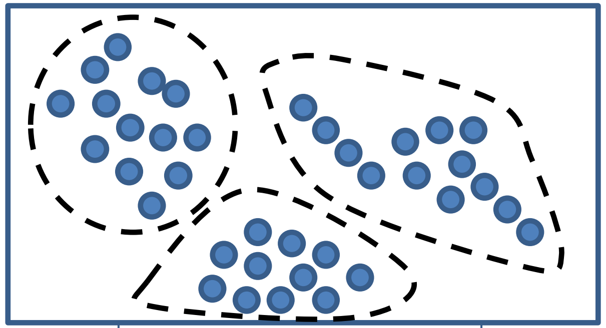
- **Supervised (inductive) learning** $\{\mathbf{x}_n \in \mathbb{R}^D, \mathbf{y}_n \in \mathbb{R}^C\}_{n=1}^N$
  - Training data includes desired outputs, $(\mathbf{x}_i, \mathbf{y}_i = f(\mathbf{x}_i))$
  - Prediction / Classification (discrete labels), Regression (real values)
- **Unsupervised learning** $\{\mathbf{x}_n \in \mathbb{R}^D\}_{n=1}^N$
  - 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

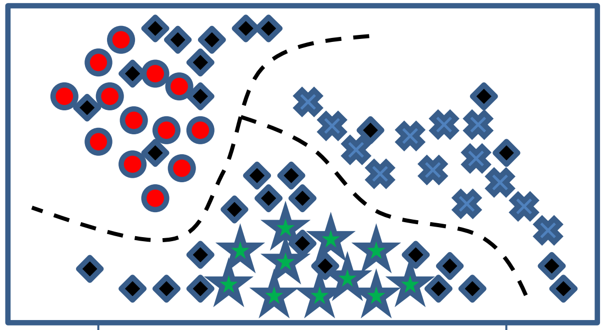
Visualizing the Types of Learning



Supervised learning



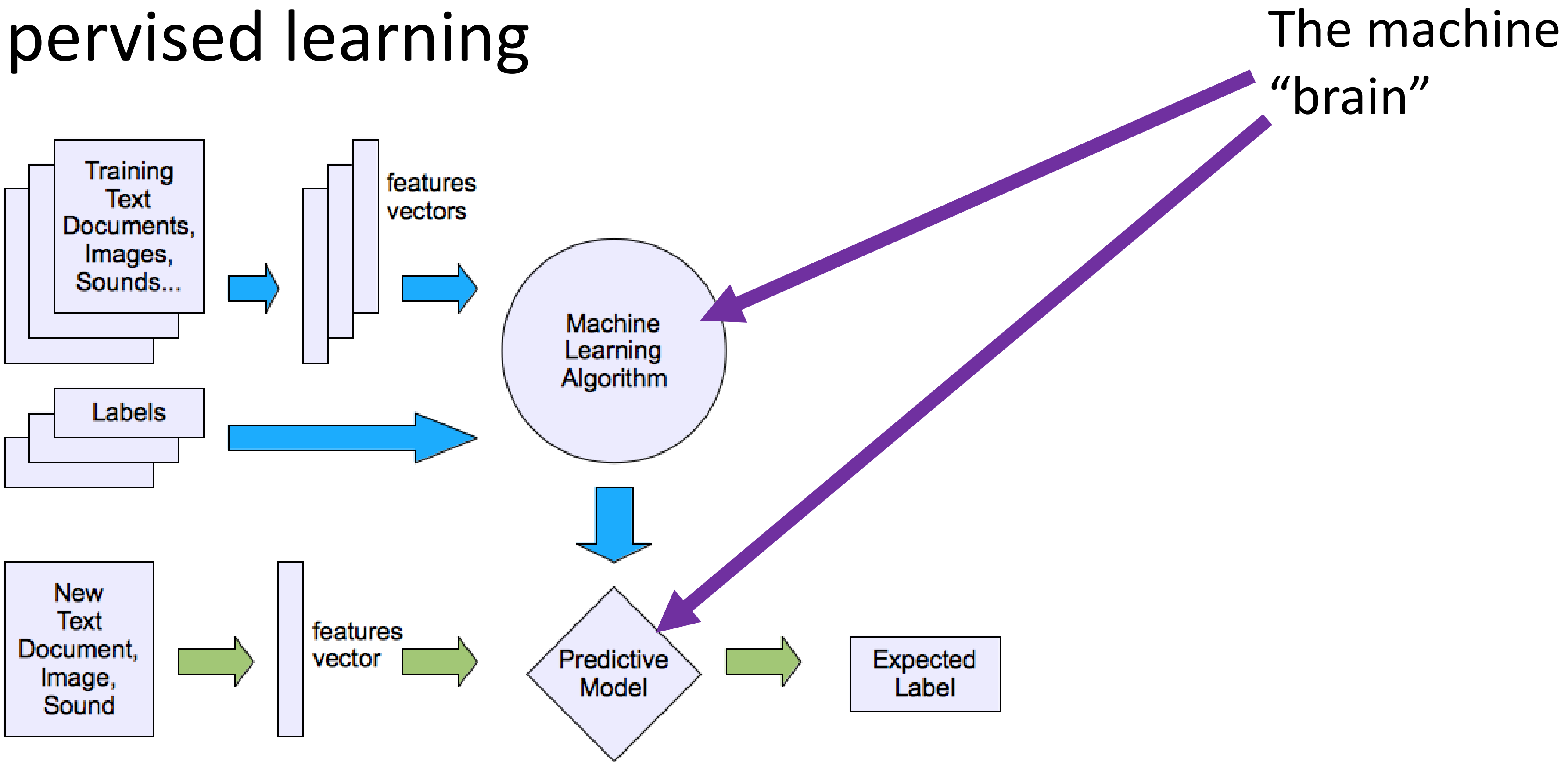
Unsupervised learning



Semi-supervised learning

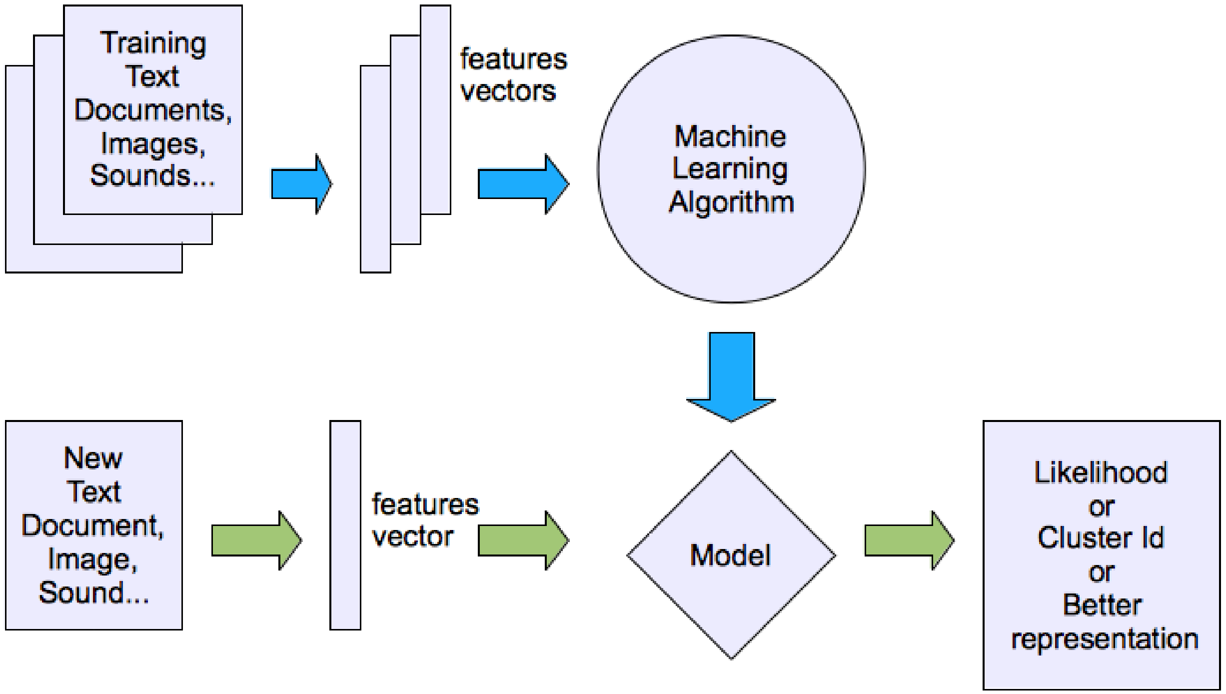
A Typical Supervised Learning Pipeline

- Supervised learning



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 635
(***You are here!***)

What big
data/infrastructure
courses help with

The problem
helps with this!

Industry dictates this
(machine learning
engineering)

Training Experience

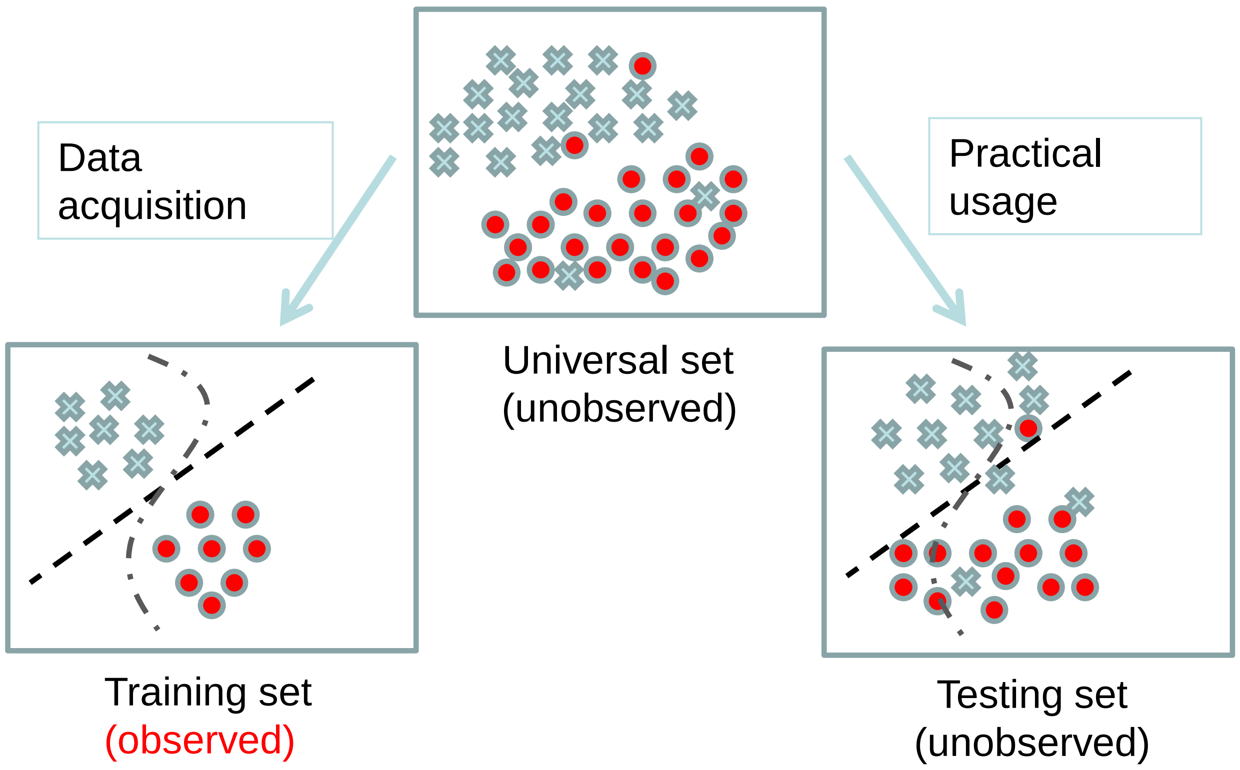
- Direct experience: Given sample input and output pairs for a useful target function
  - Checker boards labeled with the correct move, e.g. extracted from record of expert play
- Indirect experience: Given feedback which is ***not*** direct I/O pairs for a useful target function
  - Potentially arbitrary 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

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***

Key Idea: 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

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

Machine Learning Task Description

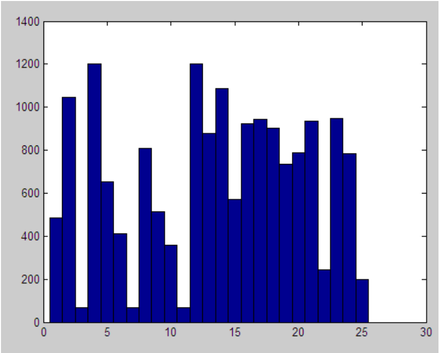
- Usually described in terms of how the machine learning system should process an example
- An example is a collection of features that have been quantitatively measured for some object/event that we want the ML system to process
- Typically represent an example as a vector $\boldsymbol{x}$ where each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

A dataset → $\{x_n \in R^d\}_{n=1}^N$

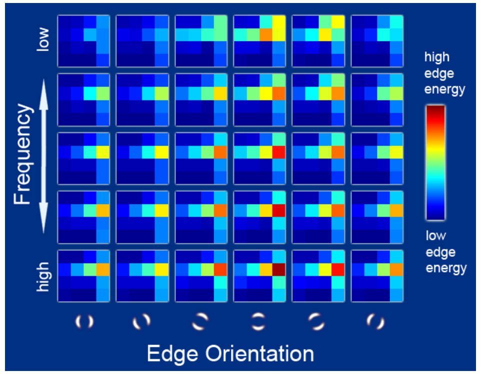
Features (What Could be "Inside" of **x**)

- Raw pixels

- Histograms



- GIST descriptors



- …

Slide credit: L. Lazebnik

The Functional Machine Learning Framework

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

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

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

output

prediction function

feature vector

parameters (the “brain”)

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

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

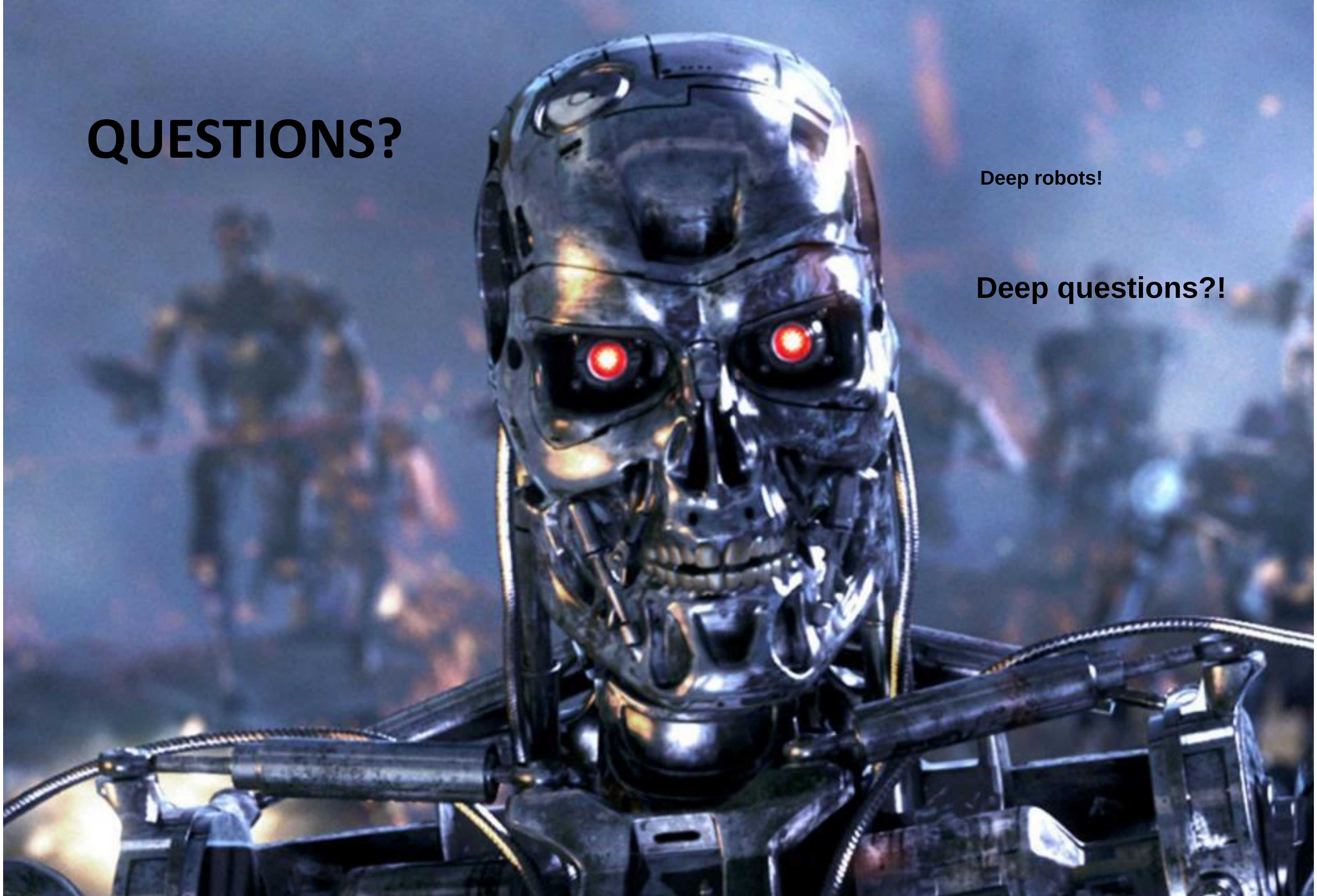
f() = “*apple*”

f() = “*tomato*”

f() = “*cow*”

Slide credit: L. Lazebnik

QUESTIONS?



Deep robots!

Deep questions?!

History of Machine Learning

- 1950s
  - Samuel’s checker player
  - Selfridge’s Pandemonium
- 1960s:
  - Neural networks: Perceptron
  - Pattern recognition
  - Learning in the limit theory
  - Minsky and Papert prove limitations of Perceptron
- 1970s:
  - Symbolic concept induction
  - Winston’s arch learner
  - Expert systems and the knowledge acquisition bottleneck
  - Quinlan’s ID3
  - Michalski’s AQ and soybean diagnosis
  - Scientific discovery with BACON
  - Mathematical discovery with AM

History of ML (cont.)

- 1980s:
  - Advanced decision tree and rule learning
  - Explanation-based Learning (EBL)
  - Learning and planning and problem solving
  - Utility problem
  - Analogy
  - Cognitive architectures
  - Resurgence of neural networks (connectionism, backpropagation)
  - Valiant’s PAC Learning Theory
  - Focus on experimental methodology
- 1990s
  - Data mining
  - Adaptive software agents and web applications
  - Text learning
  - Reinforcement learning (RL)
  - Inductive Logic Programming (ILP)
  - Ensembles: Bagging, Boosting, and Stacking
  - Bayes Net learning

History of ML (cont.)

- 2000s
  - Support vector machines
  - Kernel methods
  - Graphical models
  - Statistical relational learning
  - Transfer learning
  - Sequence labeling
  - Collective classification and structured outputs
  - Computer Systems Applications
    - Compilers
    - Debugging
    - Graphics
    - Security (intrusion, virus, and worm detection)
  - Email management
  - Personalized assistants that learn
  - Learning in robotics and vision

  - **Deep Learning**