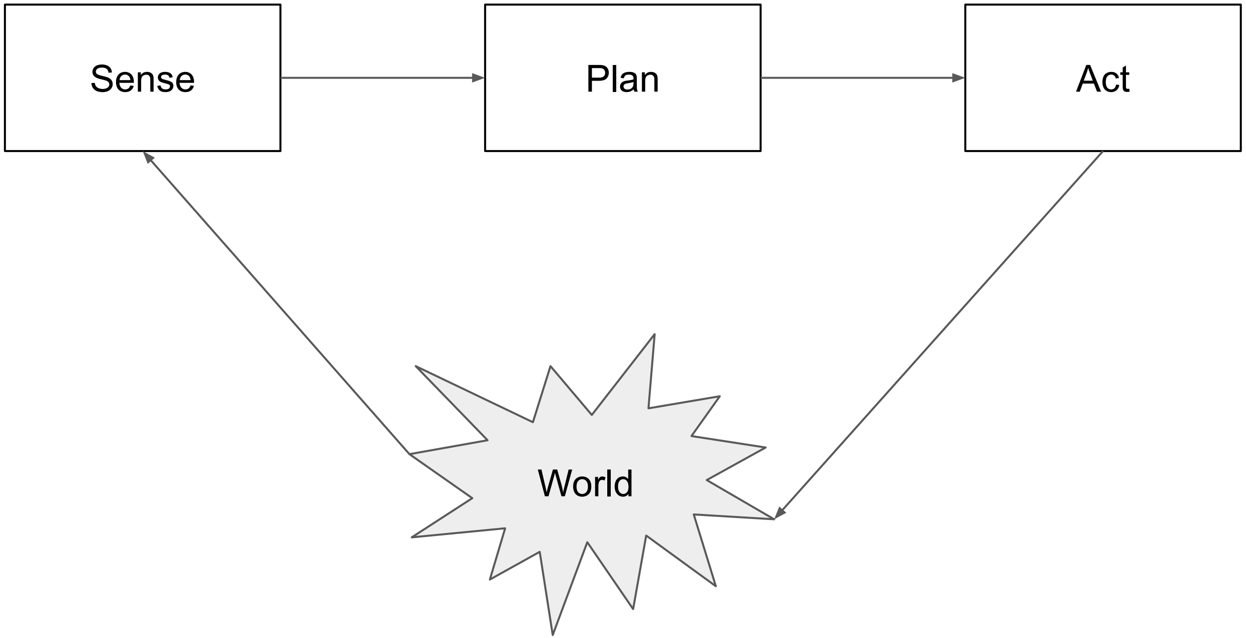
MECS 6616

Robot Learning

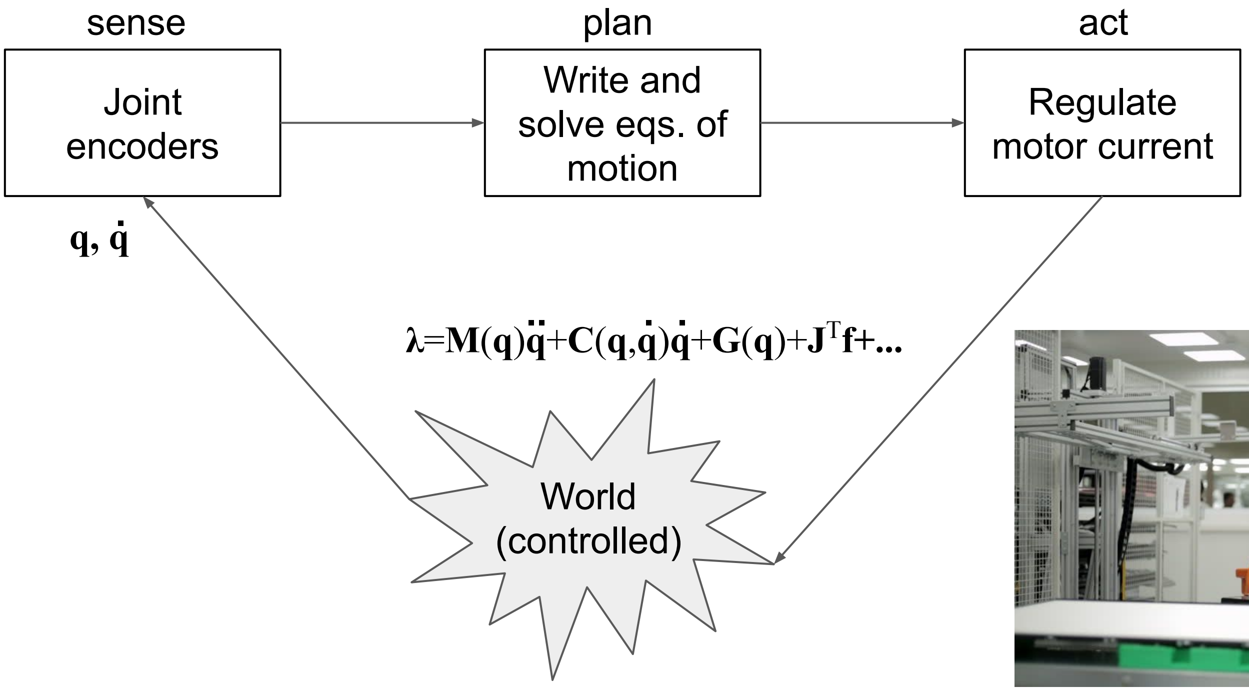
Spring 2020

Matei Ciocarlie

The Robotics Loop

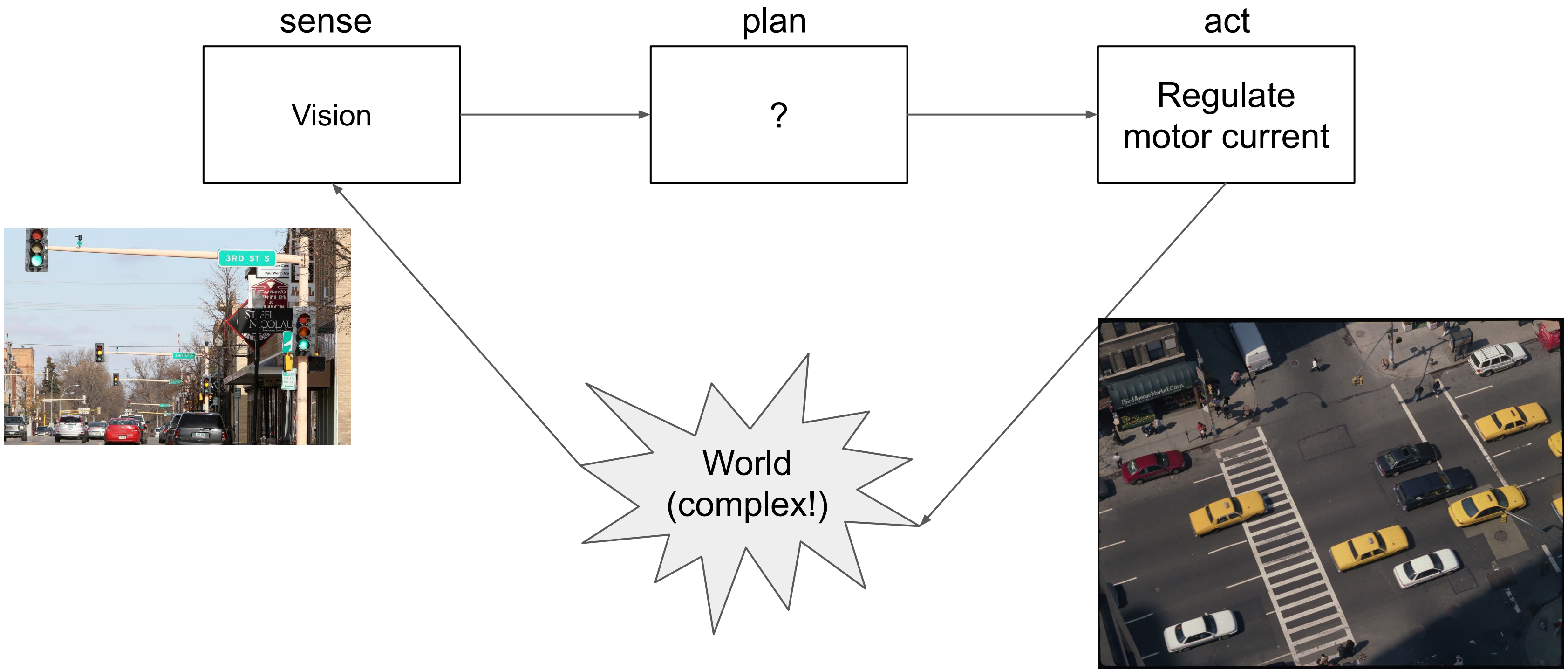


The Robotics Loop

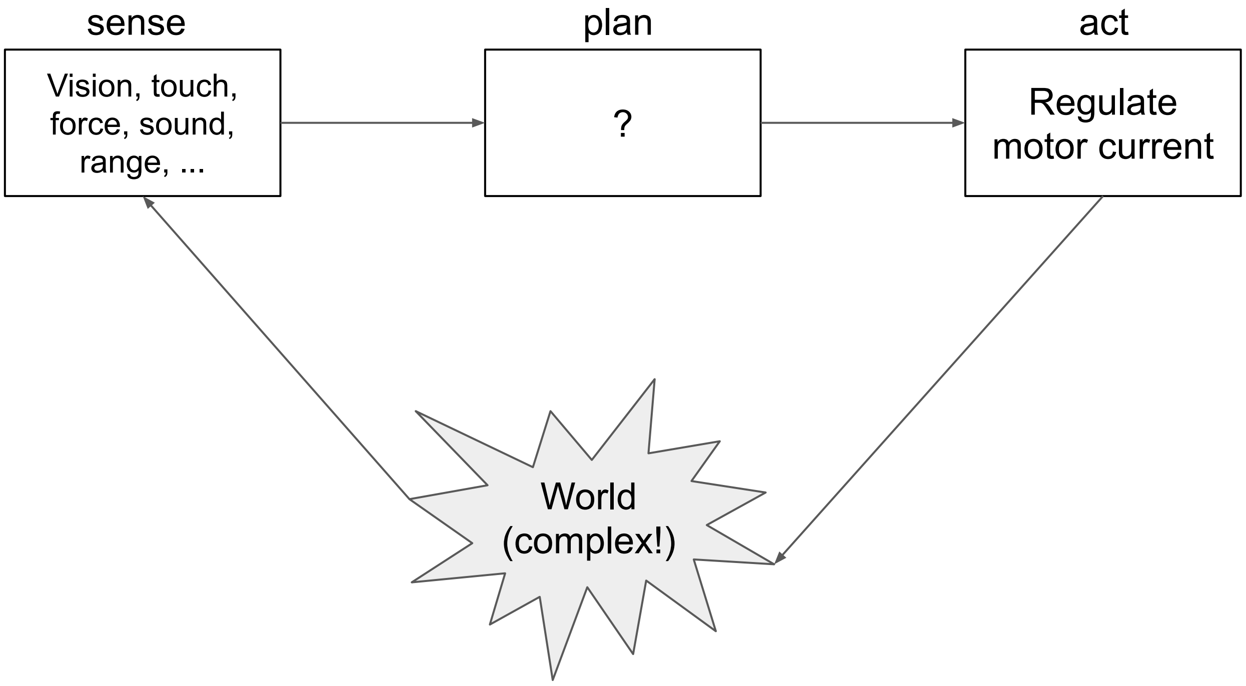


$$\boldsymbol{\lambda} = \mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{G}(\mathbf{q}) + \mathbf{J}^T\mathbf{f} + \ldots$$

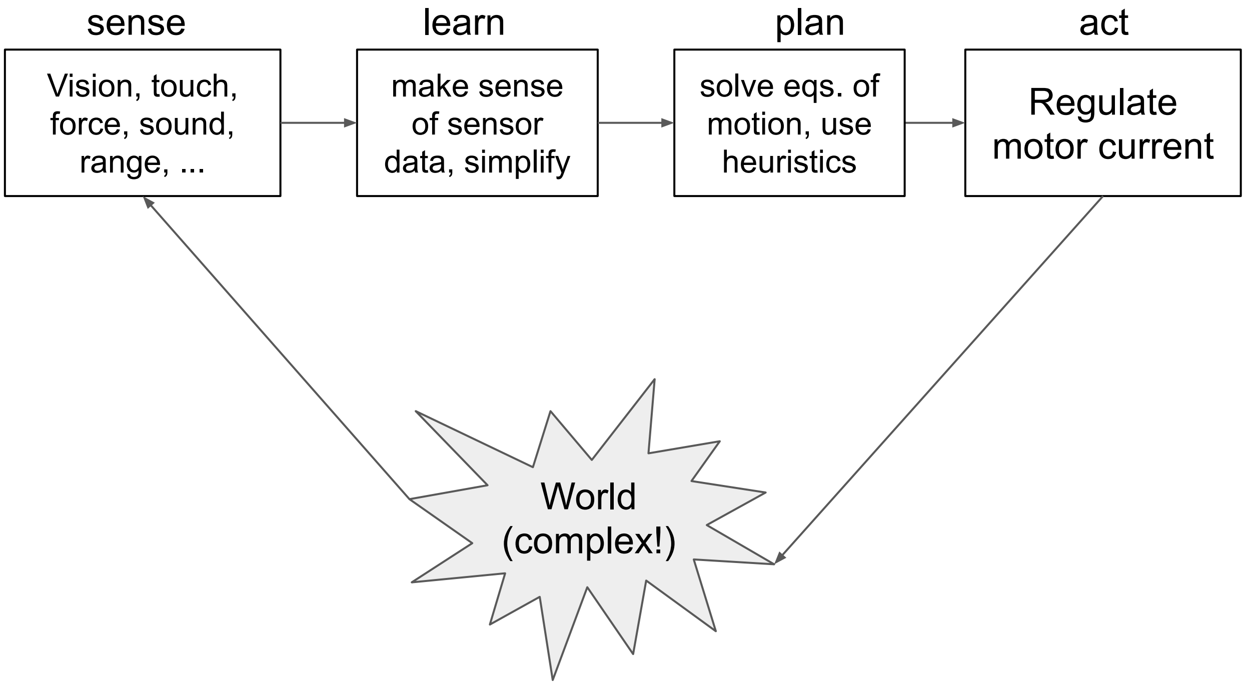
The Robotics Loop



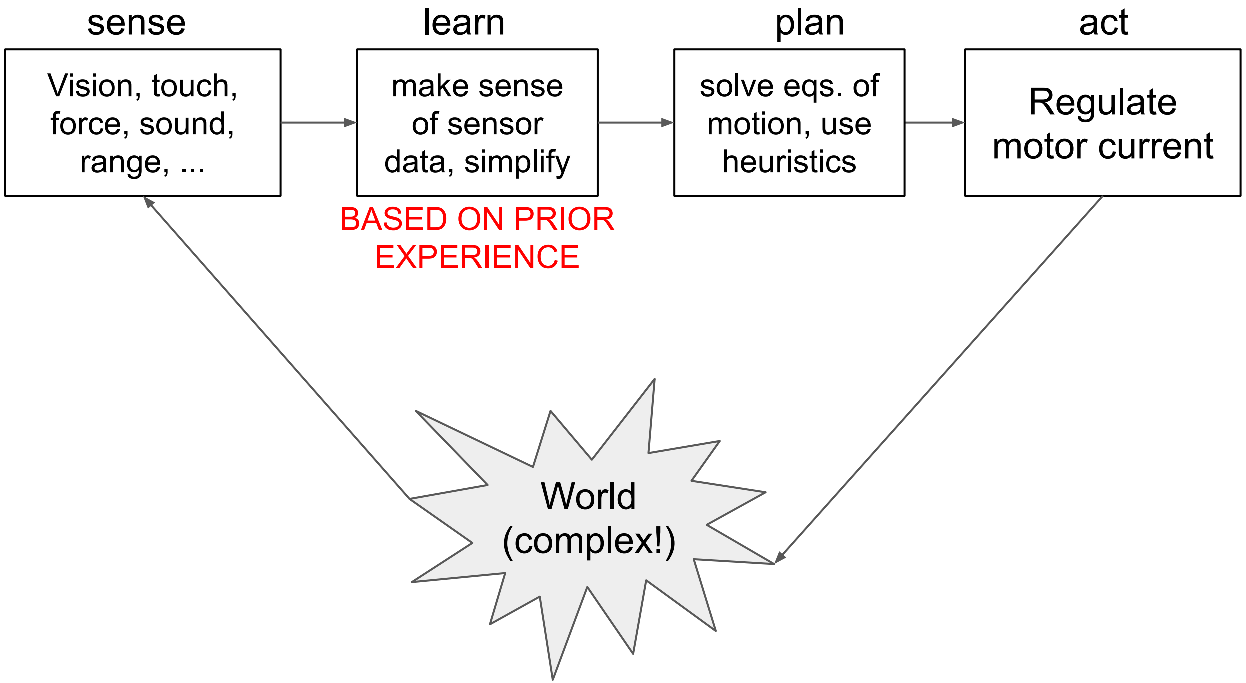
The Robotics Loop



The Robotics Loop

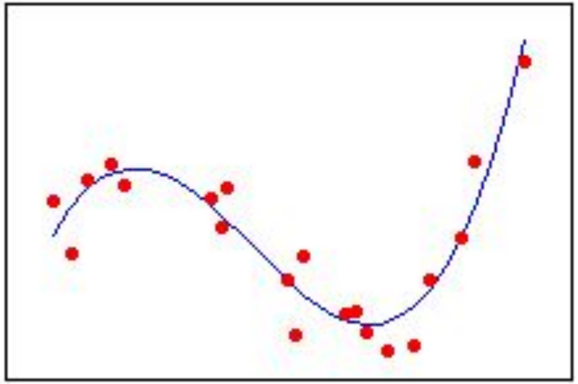


The Robotics Loop



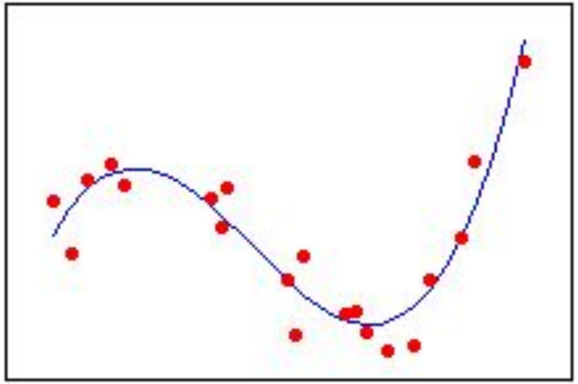
What is Machine Learning?

Option A: magic
You put in your problem and out comes the answer, and you don't need to write down a single equation about your robot.



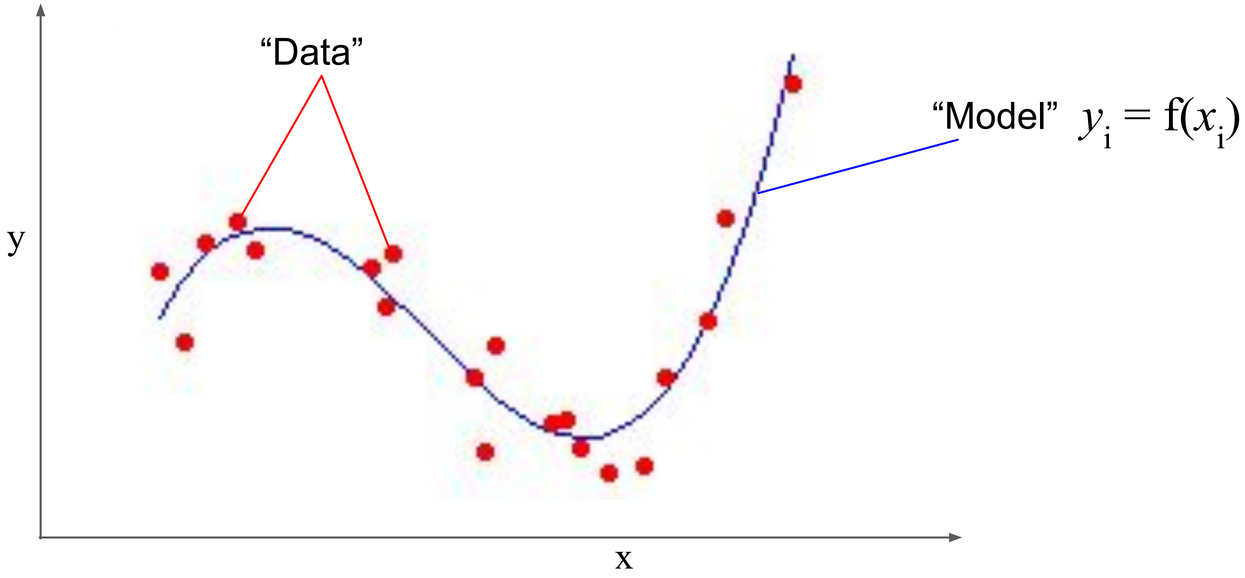
Option B: statistical estimation
You fit curves to points.

What is Machine Learning? Unfortunately:

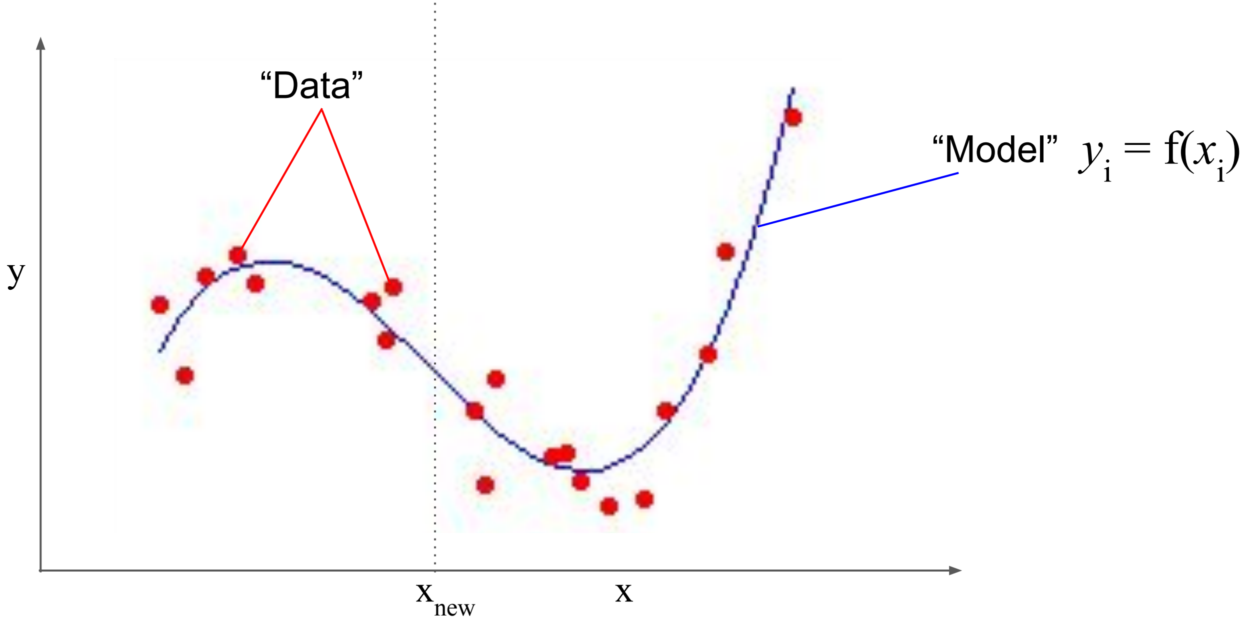


Option B: statistical estimation
You fit curves to points.

Statistical Estimation

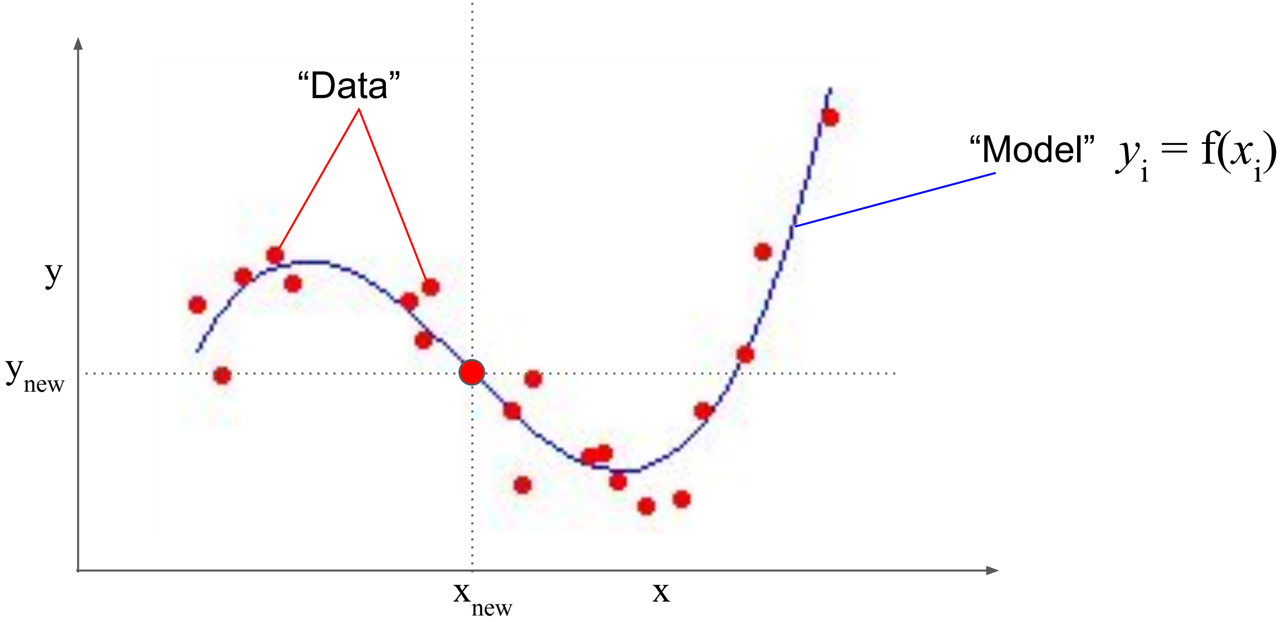


Statistical Estimation



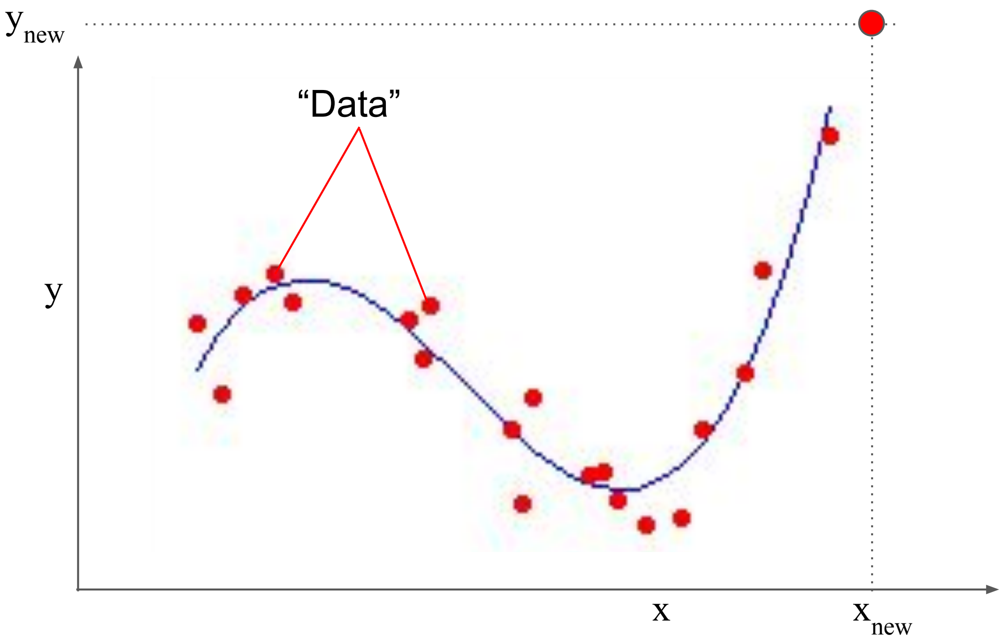
Interpolation

Statistical Estimation



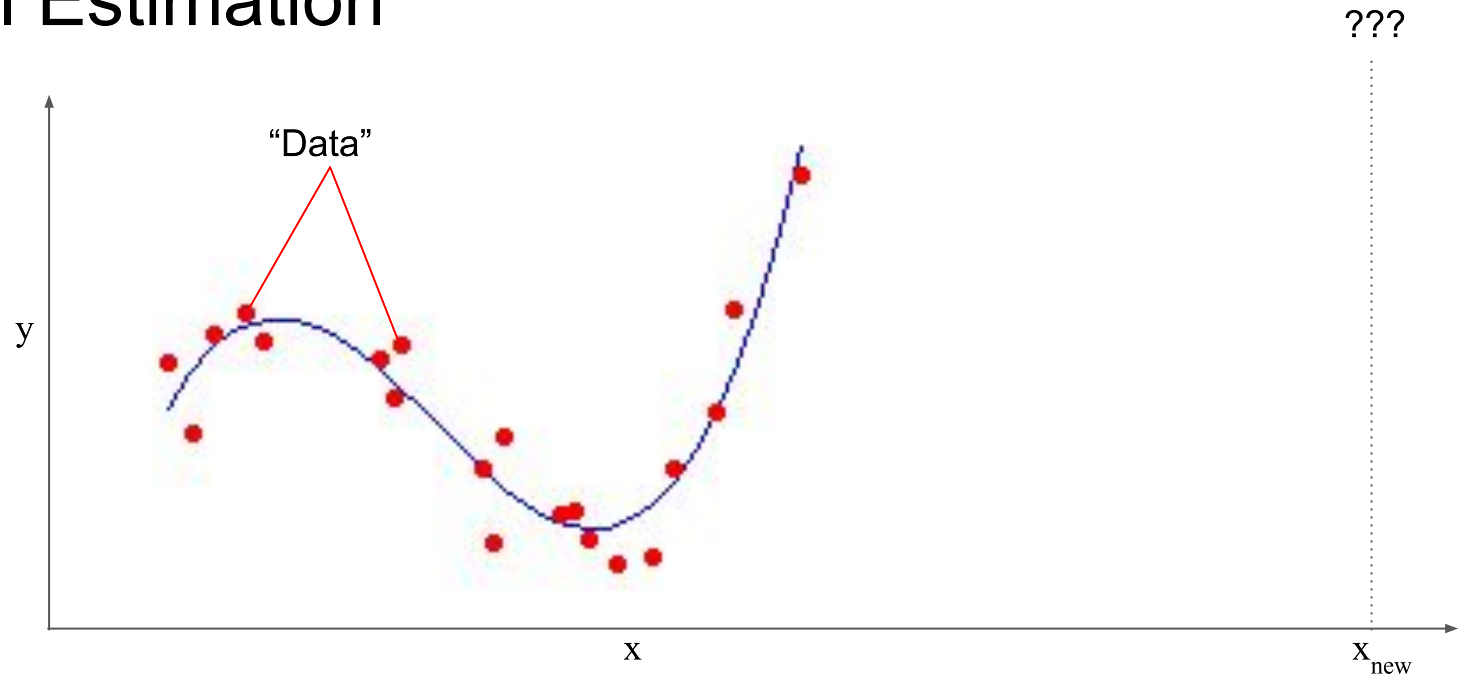
Interpolation

Statistical Estimation



Extrapolation

Statistical Estimation



Extrapolation

Broadest ML Problem Categorization

- Unsupervised
- Supervised

Fundamental Unsupervised ML Problems

$\mathbf{x} \subset \Re^{d1}$

Find some **structure** in the data you are given, **simplify**

$\mathbf{x}_1$

$\mathbf{x}_2$

...

$\mathbf{x}_n$

Clustering:

Find a number of discrete classes, or clusters:
$\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_k,\ k << n$

Assign each point to a cluster
$\mathbf{x}_k \subset \mathbf{c}_i$ (“belongs” to $\mathbf{c}_i$)

Minimize “distance” between points and their cluster

Fundamental Unsupervised ML Problems

$\mathbf{x} \subset \Re^{d1}$

Find some **structure** in the data you are given, **simplify**

$\mathbf{x}_1$

$\mathbf{x}_2$

...

$\mathbf{x}_n$

Clustering:

Find a number of discrete classes, or clusters:
$\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_k,\ k << n$

Assign each point to a cluster
$\mathbf{x}_k \subset \mathbf{c}_i$ ("belongs" to $\mathbf{c}_i$)

Minimize "distance" between points and their cluster

Dimensionality reduction:

Express each data point using fewer variables

$\mathbf{x}_i \sim f(\mathbf{y}_i),\ \mathbf{y}_i \subset \Re^{d2},\ d2 < d1$

Minimize information loss:
$\|\mathbf{x}_i - f(\mathbf{y}_i)\|$

Fundamental Supervised ML Problems

Fundamental Supervised ML Problems

$\mathbf{x} \subset \Re^{d1}, \mathbf{y} \subset \Re^{d2}$

$$\mathbf{x}_1 \rightarrow \mathbf{y}_1$$

$$\mathbf{x}_2 \rightarrow \mathbf{y}_2$$

$$\ldots$$

$$\mathbf{x}_n \rightarrow \mathbf{y}_n$$

Fundamental Supervised ML Problems

$\mathbf{x} \subset \Re^{d1}, \mathbf{y} \subset \Re^{d2}$

$\mathbf{x}_1 \rightarrow \mathbf{y}_1$

$\mathbf{x}_2 \rightarrow \mathbf{y}_2$

$\dots$

$\mathbf{x}_n \rightarrow \mathbf{y}_n$

$\mathbf{x}_{new} \rightarrow ?$

Fundamental Supervised ML Problems

$\mathbf{x} \subset \Re^{d1}, \mathbf{y} \subset \Re^{d2}$

$\mathbf{x}_1 \rightarrow \mathbf{y}_1$

$\mathbf{x}_2 \rightarrow \mathbf{y}_2$

...

$\mathbf{x}_n \rightarrow \mathbf{y}_n$

$\mathbf{x}_{new} \rightarrow ?$

Classification:

$d2 = 1$

$y \subset \{c_1, c_2, \ldots, c_k\}$

y is **discrete**

y_i tells you what *class* x_i belongs to

Fundamental Supervised ML Problems

$\mathbf{x} \subset \Re^{d1}, \mathbf{y} \subset \Re^{d2}$

$\mathbf{x}_1 \rightarrow \mathbf{y}_1$

$\mathbf{x}_2 \rightarrow \mathbf{y}_2$

...

$\mathbf{x}_n \rightarrow \mathbf{y}_n$

$\mathbf{x}_{new} \rightarrow ?$

Classification:

$d2 = 1$

$y \subset \{c_1, c_2, \ldots, c_k\}$

y is **discrete**

y_i tells you what *class* x_i belongs to

Regression:

$y \subset \Re^{d2}$

y is **continuous** (potentially multi-dimensional)

x_i is being **mapped** to point y_i in a different space

Fundamental Supervised ML Problems

$\mathbf{x} \subset \Re^{d1}, \mathbf{y} \subset \Re^{d2}$

$\mathbf{x}_1 \rightarrow \mathbf{y}_1$

$\mathbf{x}_2 \rightarrow \mathbf{y}_2$

...

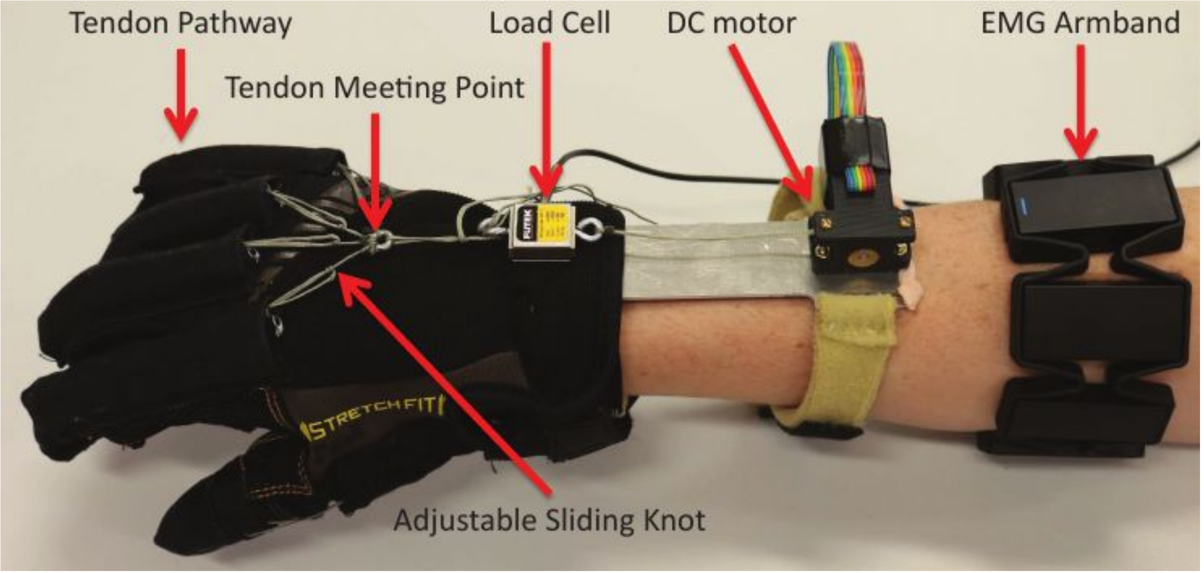
$\mathbf{x}_n \rightarrow \mathbf{y}_n$

$\mathbf{x}_{new} \rightarrow ?$

Classification:

$\mathbf{x} = [e_1, e_2, \ldots, e_8]^T$

$\mathbf{y}$ = {"trying to open", "relaxing"}



Fundamental Supervised ML Problems

$\mathbf{x} \subset \Re^{d1}$, $\mathbf{y} \subset \Re^{d2}$

$\mathbf{x}_1 \rightarrow \mathbf{y}_1$

$\mathbf{x}_2 \rightarrow \mathbf{y}_2$

...

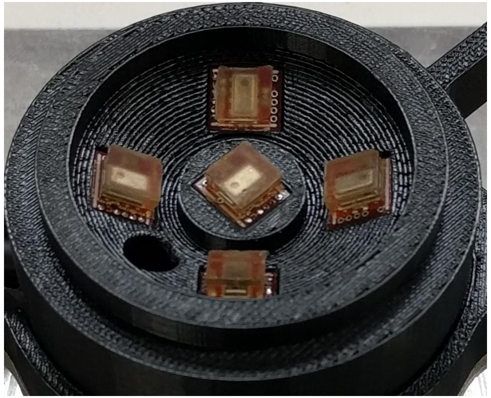
$\mathbf{x}_n \rightarrow \mathbf{y}_n$

$\mathbf{x}_{new} \rightarrow ?$

Regression:

$\mathbf{x} = [s_1, s_2, \dots, s_5]^T$

$\mathbf{y} = [l_a, l_b]^T$



Nomenclature

$\mathbf{x} \subset \Re^{d1}$, $\mathbf{y} \subset \Re^{d2}$

$\mathbf{x}_1 \rightarrow \mathbf{y}_1$

$\mathbf{x}_2 \rightarrow \mathbf{y}_2$

$\ldots$

$\mathbf{x}_n \rightarrow \mathbf{y}_n$

$\mathbf{x}_{new} \rightarrow ?$

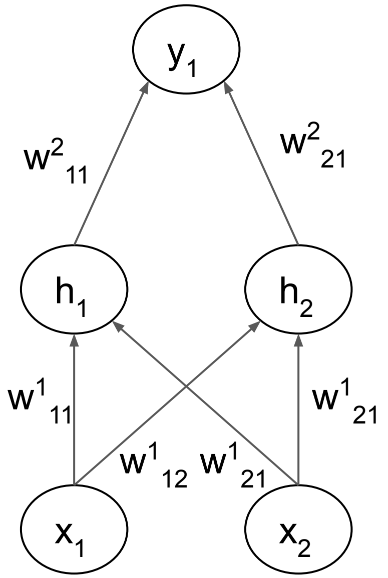
$\mathbf{x}$: “features” / “independent variables”
$\mathbf{y}$: “labels” / “dependent variables”

“examples” / “training” / “fitting”

“testing” / “predicting”

Neural Networks

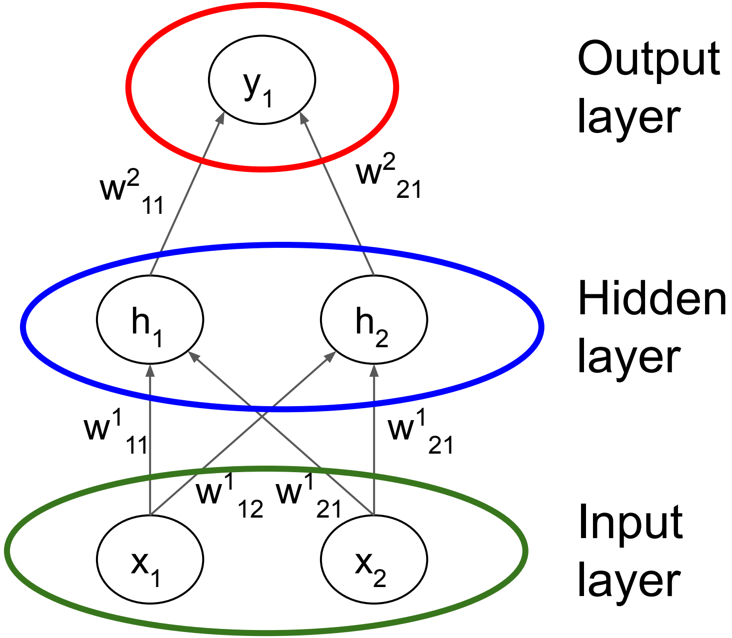
- $h_i = f(\sum x_j \, w^1_{ji})$
- $y_i = \sum h_j \, w^2_{ji}$
- Training the network: finding w^k_{ij}
- Done via backpropagation algorithm



“Fully connected”

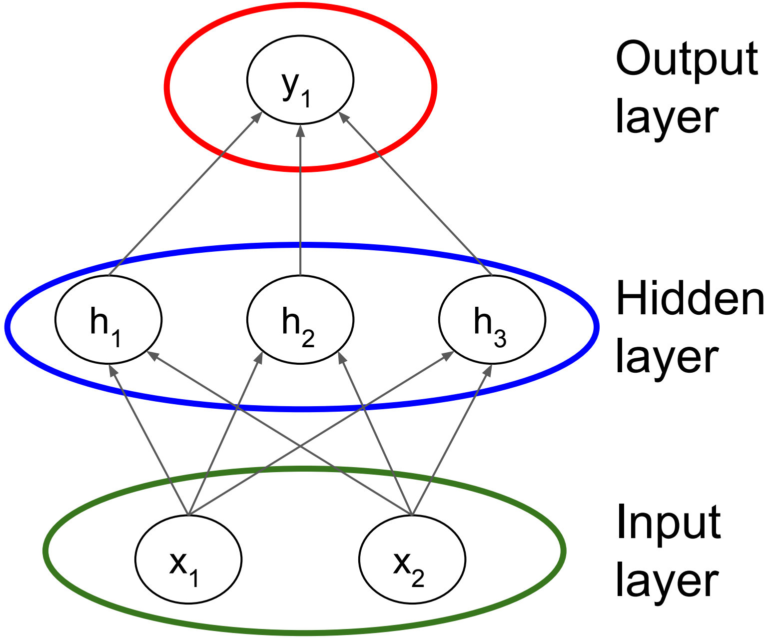
Neural Networks

- $h_i = f(\sum x_j w^1_{ji})$
- $y_i = \sum h_j w^2_{ji}$
- Training the network: finding w^k_{ij}
- Done via backpropagation algorithm



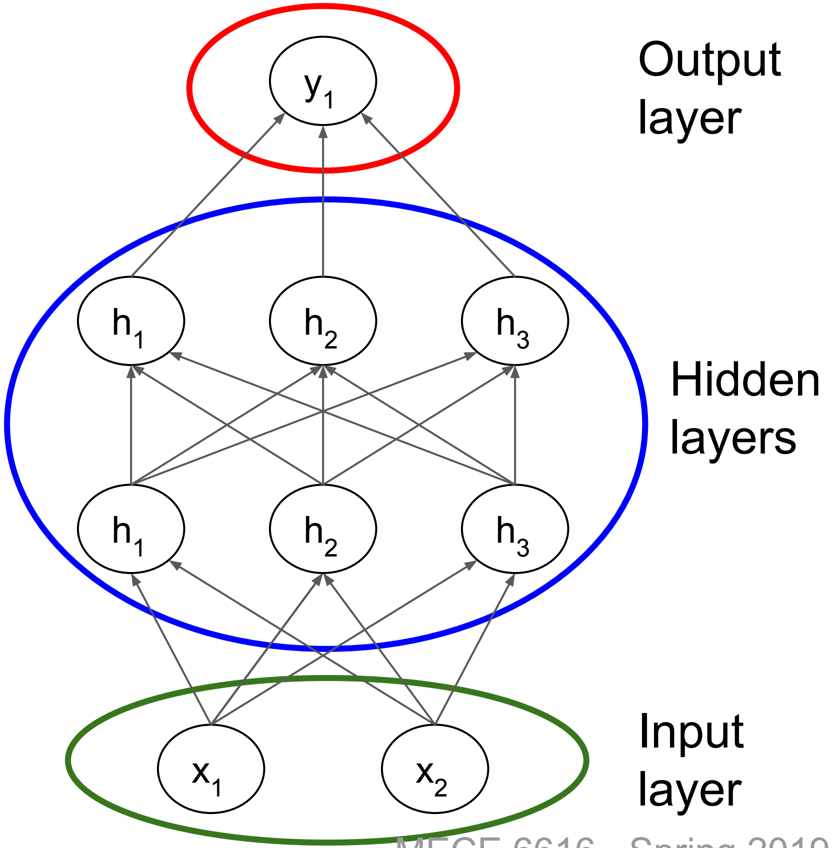
Neural Networks

- $h_i = f(\sum x_j \, w^1_{ji})$
- $y_i = \sum h_j \, w^2_{ji}$
- Training the network: finding w^k_{ij}
- Done via backpropagation algorithm



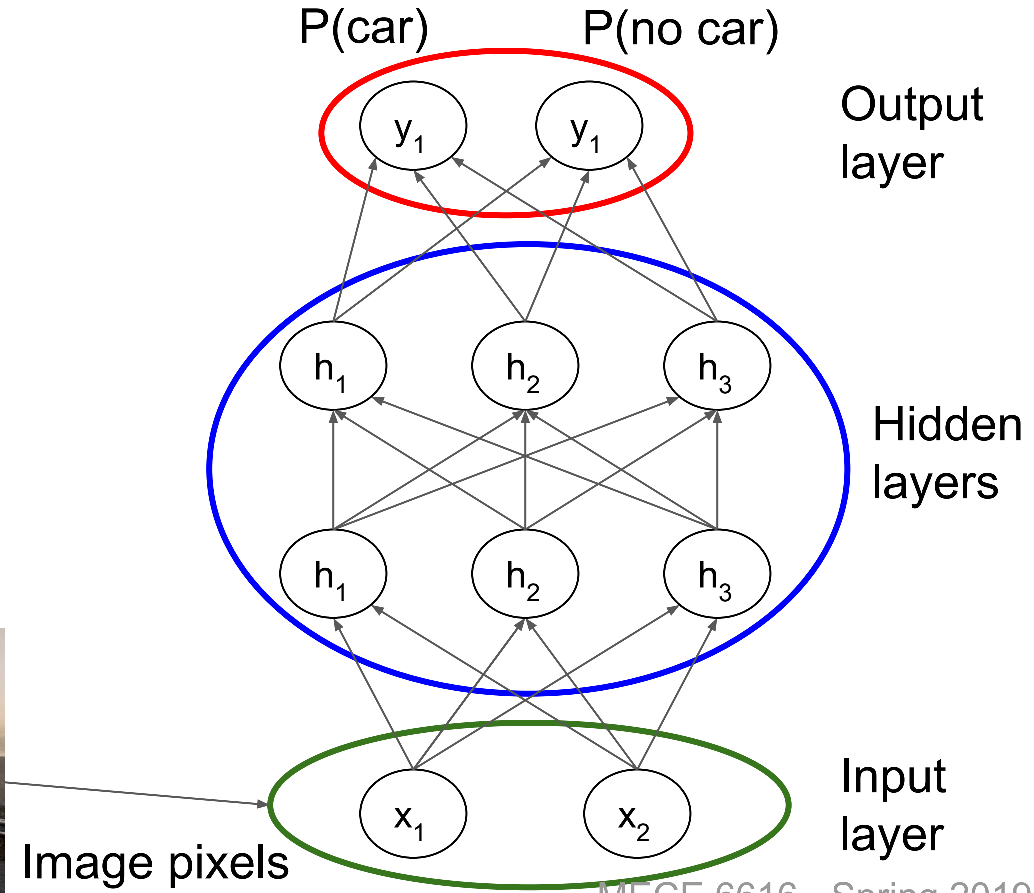
Deep Learning

- Deep Neural Network (DNN): more than one hidden layer
- Even two hidden layers make for incredibly powerful approximators even for highly non-linear functions
- Hidden layers can have different sizes
- Enabled by:
  - better training (backprop)
  - massively parallel computers (GPUs)



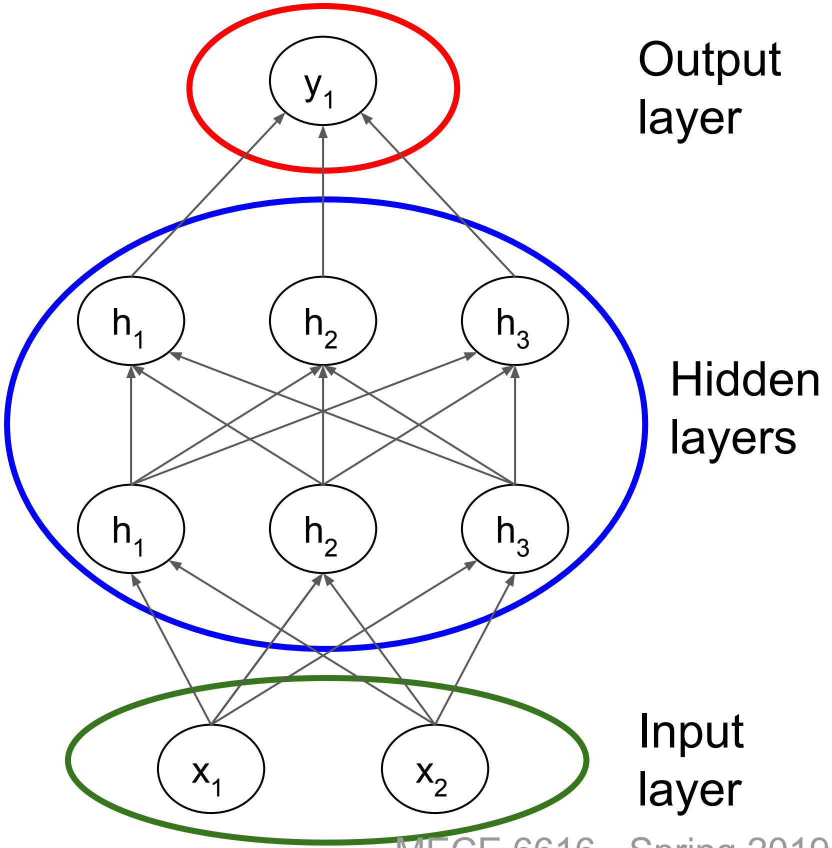
Deep Neural Networks

Image pixels
(all of them!)



Deep Neural Networks

- "Network engineering": setting the number and dimensions of the hidden layers
- Qualitative characteristics:
  - for some problems, they easily outperform anything else
  - very hard to model and analyze quantitatively, to get a sense for what makes them "tick"
  - need large amounts of data to train



What Could I Be Learning?

- Input: **sensor data**
  - Semantic meaning of my sensor data
- Input **sensor data plus action**
  - What will happen next?
  - How “good” is this action?
  - What action should I take?

What Could I Be Learning?

- Input: **sensor data**
  - Semantic meaning of my sensor data
- Input **sensor data plus action**
  - What will happen next?
  - How “good” is this action?
  - What action should I take?

Learning Dynamic Models

- time enters the picture - denoted by subscript t
- system state: $\mathbf{x}_t \subset \Re^{d1}$
- action / command: $\mathbf{a}_t \subset \Re^{d2}$
- **model of system dynamics:**
  - $\mathbf{x}_{t+1} = f(\mathbf{x}_t, \mathbf{a}_t)$
  - shorthand: dynamic(s) model, forward model

Learning Dynamic Models

- time enters the picture - denoted by subscript t
- system state: $\mathbf{x}_t \subset \Re^{d1}$
- action / command: $\mathbf{a}_t \subset \Re^{d2}$
- **model of system dynamics:**
  - $\mathbf{x}_{t+1} = f(\mathbf{x}_t, \mathbf{a}_t)$
  - shorthand: dynamic(s) model, forward model
- “model-based”: $f(\mathbf{x}_t, \mathbf{a}_t)$ is an analytical function written by the roboticist
- “learning-based”: $f(\mathbf{x}_t, \mathbf{a}_t)$ is learned from data, standard regression problem:

$$(\mathbf{x}_t, \mathbf{a}_t) \rightarrow \mathbf{x}_{t+1}$$

Learning Dynamic Models

- **model of system dynamics:**
  - $\mathbf{x}_{t+1} = f(\mathbf{x}_t, \mathbf{a}_t)$
- “learning-based”: $f(\mathbf{x}_t, \mathbf{a}_t)$ is learned from data, standard regression problem
- why learn dynamic models?
  - can use many non-learning techniques derived over the years that require the existence of one

What Could I Be Learning?

- Input: **sensor data**
  - Semantic meaning of my sensor data
- Input **sensor data plus action**
  - What will happen next?
  - How “good” is this action?
  - What action should I take?

Reinforcement Learning

Conceptually very different from supervised / unsupervised learning.

- system state: $\mathbf{x}_t \subset \Re^{d1}$ - known
- action / command: $\mathbf{a}_t \subset \Re^{d2}$ - controllable
- dynamics / forward model:
  - deterministic: $\mathbf{x}_{t+1} = f(\mathbf{x}_t, \mathbf{a}_t)$
  - stochastic: $P(\mathbf{x}_{t+1} \mid \mathbf{x}_t, \mathbf{a}_t) = f(\mathbf{x}_t, \mathbf{a}_t)$
  - could be known (model-based RL) or unknown (model-free RL)
- reward: $r(\mathbf{x}_t, \mathbf{a}_t)$ - known
- goal: learn a **policy** that maximizes **expected reward**
  - deterministic: $\mathbf{a}_t = \Pi(\mathbf{x}_t)$
  - stochastic: $P(\mathbf{a}_t \mid \mathbf{x}_t) = \Pi(\mathbf{a}_t, \mathbf{x}_t)$

Reinforcement Learning

Conceptually very different from supervised / unsupervised learning.

- reward: $\mathrm{r}(\mathbf{x}_t, \mathbf{a}_t)$ - known
- goal: learn a **policy** that maximizes **expected reward**
    - deterministic: $\mathbf{a}_t = \Pi(\mathbf{x}_t)$
    - stochastic: $\mathrm{P}(\mathbf{a}_t \mid \mathbf{x}_t) = \Pi(\mathbf{a}_t, \mathbf{x}_t)$
- robot is never told “how to do something”, only if what it did is good or bad
- many algorithms:
    - Q-learning
    - policy gradient
    - ...

How Can I Apply ML to My Robotics Problem?

Key questions:

- **What kind of a problem is it?** Clustering, dimensionality reduction, regression, classification, policy.

How Can I Apply ML to My Robotics Problem?

Key questions:

- **What kind of a problem is it?** Clustering, dimensionality reduction.
- **What are the features?** How do I get a feature vector $\mathbf{x} \subset \Re^{d1}$ from my data?

How Can I Apply ML to My Robotics Problem?

Key questions:

- **What kind of a problem is it?** Regression, classification.
- **What are the features?** How do I get a feature vector $\mathbf{x} \subset \Re^{d1}$ from my data?
- **What are the labels?** How do I express what I care about as $\mathbf{y} \subset \Re^{d2}$?
- **Where does my training data come from?** It needs to be labeled with "ground truth".

How Can I Apply ML to My Robotics Problem?

Key questions:

- **What kind of a problem is it?** Policy
- **What is my state space?** It needs to be known for training **and** testing.
- **What is my action space?** It needs to contain things I can control.
- **What is my reward function?** I need to have access to a reward signal, either by computing it, or from an oracle.

How Can I Apply ML to My Robotics Problem?

For all problems:

- **What algorithm/model is being used?**
  - “fit” my data well
  - run in reasonable time on my problem size
  - have low training loss
- **How are the results tested?**
  - what is the performance on the testing data?
  - what is the relationship between training and testing data?
  - <u>*is testing data predictive of deployment?*</u>

Course Structure

- Components (roughly 7-10 lectures each):
  - Part 1: Dimensionality reduction, classification and regression
  - Part 2: Deep Learning and learning in Computer Vision
  - Part 3: Reinforcement learning
- Lectures
  - theoretical concept presentation (in moderate depth)
  - discussion of papers in Robotics using these concepts

Course Structure

- 3 Programming Projects: one for each part
  - Each project will make up ~15% of the grade.
  - All projects: Python, Ubuntu 16.04, ROS, SVN. **Must be comfortable with all of these.**
    - There will be no skeleton code. You will write ROS nodes starting from blank files.
  - You will get a budget of 5 late days on projects, to use how you see fit.
- Written exams
  - Midterm and final: each will make up ~20% of the grade.
  - Pencil-and-paper questions: conceptual, algorithmic, math-y

Additional details for course structure, grading, collaboration policy, late submissions, etc. **can be found on the Courseworks2 course webpage.**