CS 4644 / 7643-A
DANFEI XU

Topics:

- Machine learning intro, applications (CV, NLP, etc.)
- Parametric models and their components

- **PS0: This should take less than 2hrs!**
- Please do it now, and give others a chance at waitlist if your background is not sufficient (beef it up and take it next time)
    - Do it even if you're on the waitlist!

- **Office hours** start next week
- **Start finding your project partners (piazza post is up)**

Administrivia

- **Grace period**
  - 2 days grace period for each assignment (**EXCEPT PS0**)
    - 20% penalty

- **After grace period, you get a 0 (no excuses except medical)**
  - Send all medical requests to dean of students (https://studentlife.gatech.edu/)
  - Form: https://gatech-advocate.symplicity.com/care_report/index.php/pid224342

- **DO NOT SEND US ANY MEDICAL INFORMATION!** We do not need any details, just a confirmation from dean of students

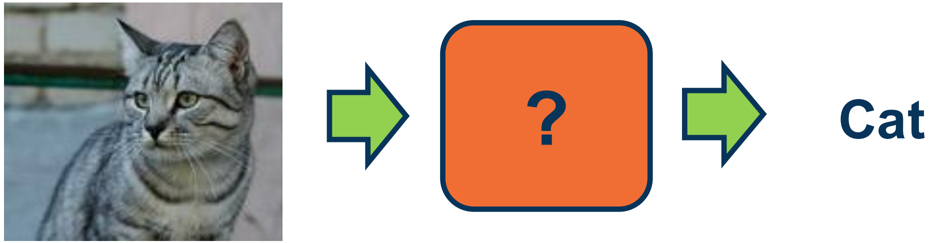
Grace Period

Machine Learning Overview

When is Machine Learning useful?

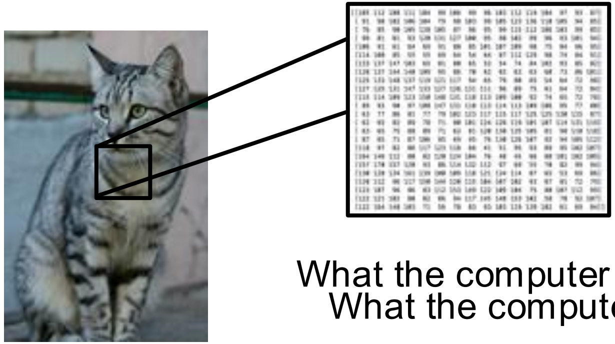
```
algorithm quicksort(A, lo, hi) is
    if lo < hi then
        p := partition(A, lo, hi)
        quicksort(A, lo, p - 1)
        quicksort(A, p + 1, hi)

algorithm partition(A, lo, hi) is
    pivot := A[hi]
    i := lo
    for j := lo to hi do
        if A[j] < pivot then
            swap A[i] with A[j]
            i := i + 1
    swap A[i] with A[hi]
    return i
```



When it’s difficult / infeasible to write a program

Example: Object Recognition



What the computer sees
What the computer sees

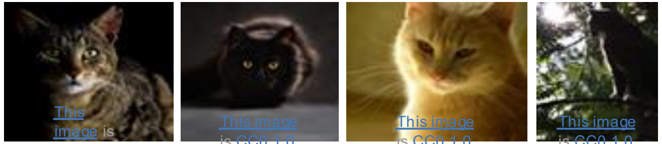
An image is just a big grid of numbers between [0, 255]:

e.g. 800 x 600 x 3
(3 channels RGB)

Viewpoint Changes

All pixels change when the camera moves!

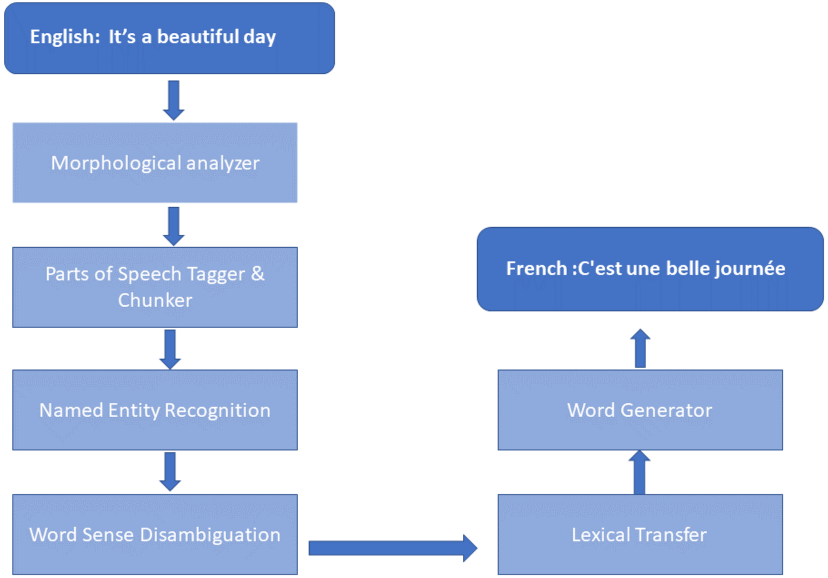
Illumination



Deformation

Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n

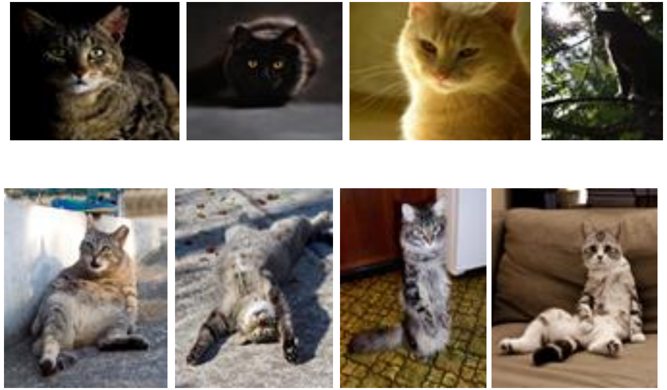
Example: Machine Translation



But what about …

- Word play, jokes, puns, hidden messages
- Concept gaps: go Jackets! George P. Burdell
- Other constraints: lyrics, dubbing, poem, …
- …

The Power of Machine Learning



Model

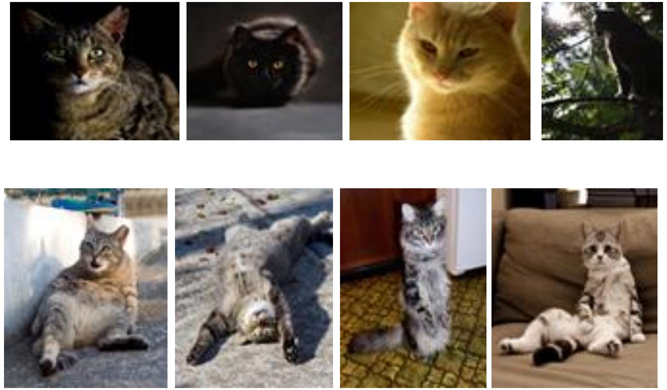
Cat

It's a beautiful day

Model

C'est une belle journée

The Power of Machine Learning



Deep Neural Networks

Cat

It's a beautiful day

Deep Neural Networks

C'est une belle journée

The Power of Machine Learning

The Power of (Deep) Machine Learning

TECHNOLOGY

A Massive Google Network Learns To Identify — Cats

June 26, 2012 · 3:00 PM ET

Heard on All Things Considered

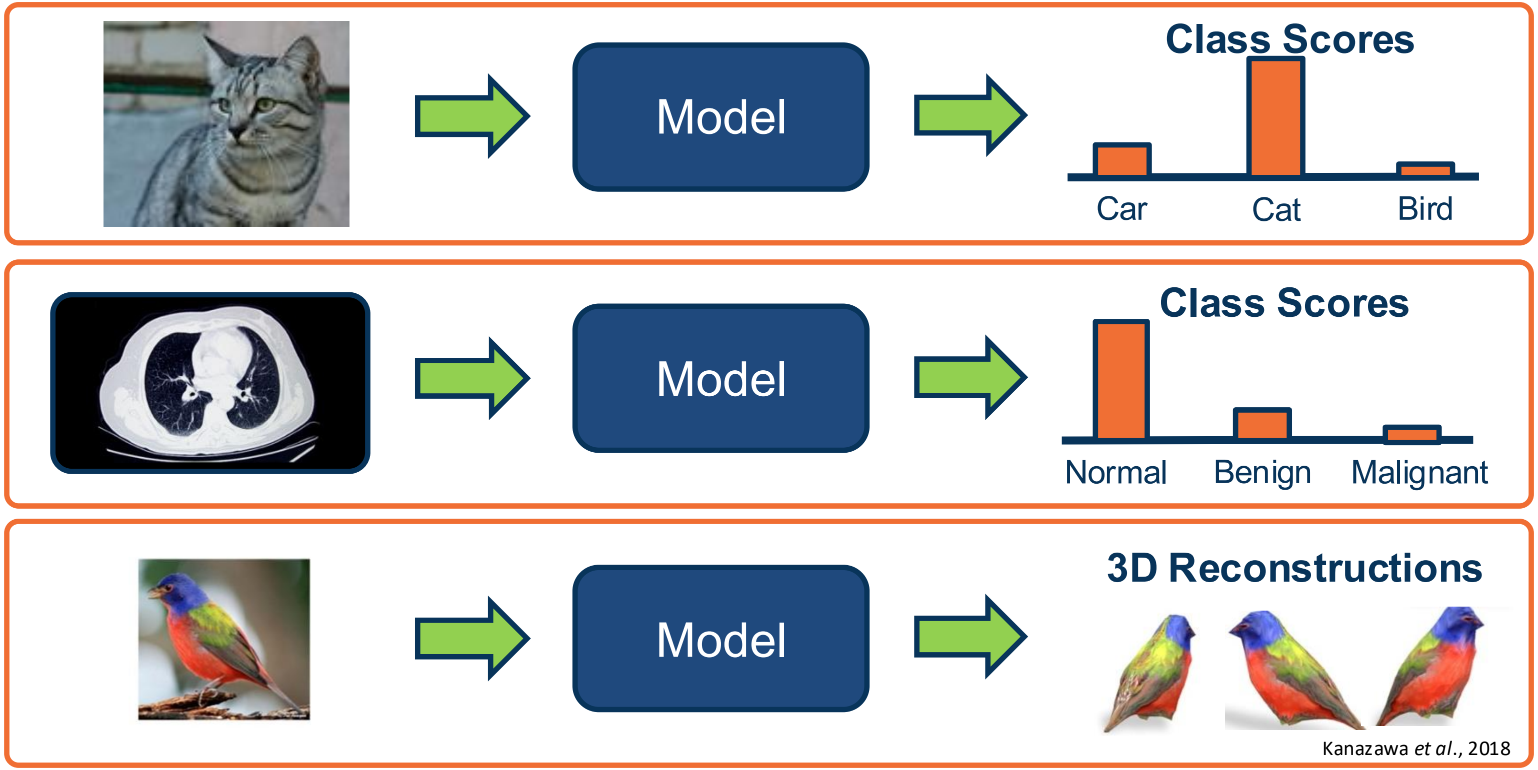
4-Minute Listen

+ PLAYLIST

All Things Considered host Audie Cornish talks with Andrew Ng, Associate Professor of Computer Science at Stanford University. He led a Google research team in creating a neural network out of 16,000 computer processors to try and mimic the functions of the human brain. Given three days on YouTube, the network taught itself how to identify — cats.

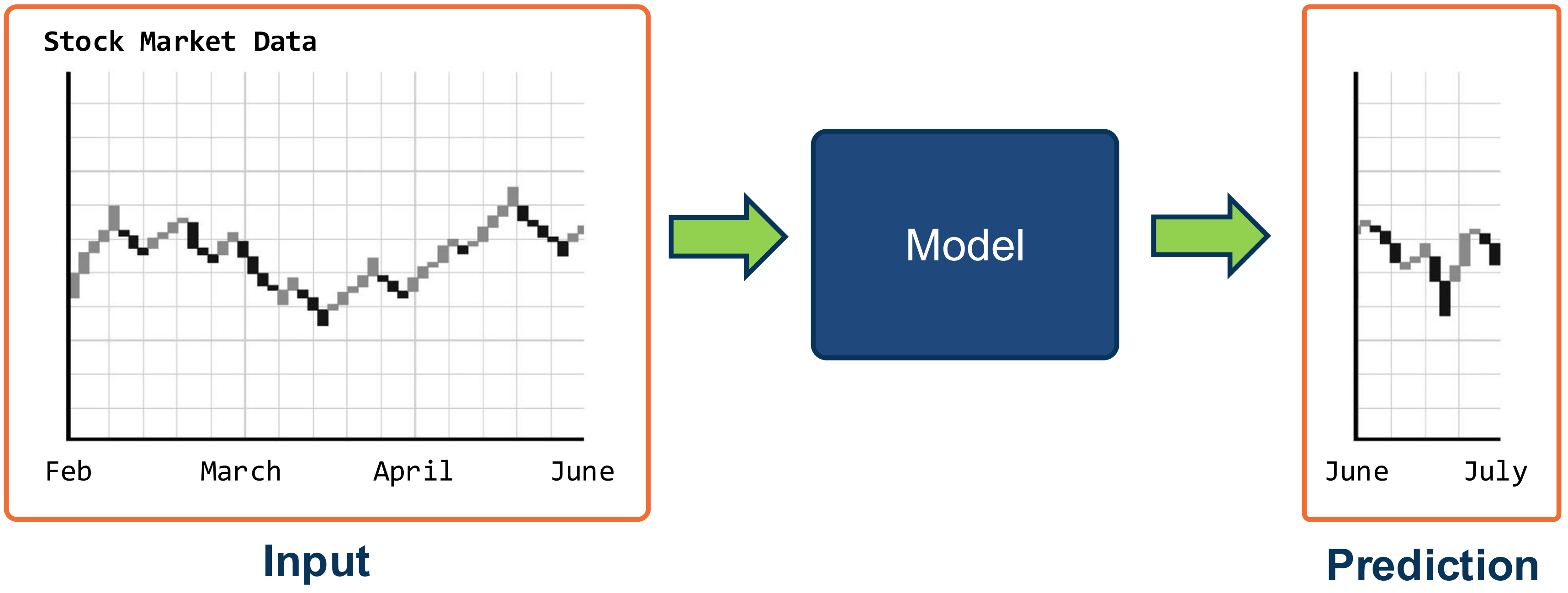
Source: https://www.npr.org/2012/06/26/155792609/a-massive-google-network-learns-to-identify

Application: Computer Vision

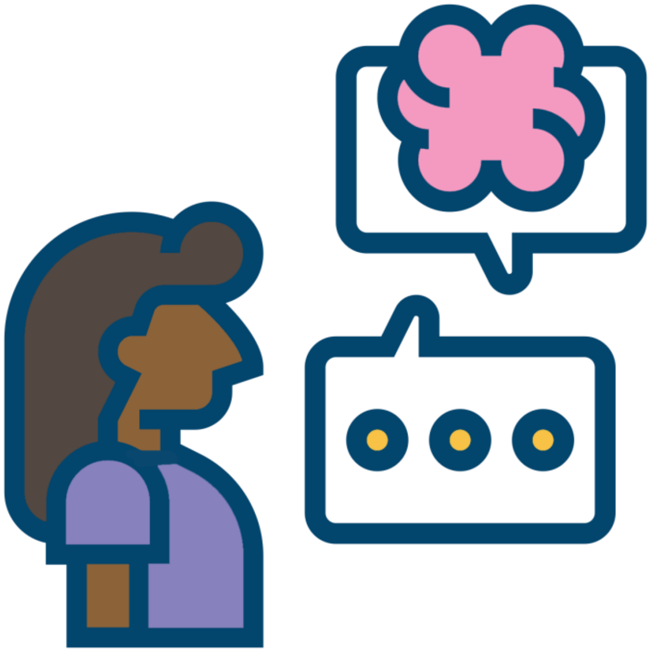


Example: Image Classification

Application: **Time Series Forecasting**



Application: Natural Language Processing (NLP)



Very large number of NLP sub-tasks:

- Syntax Parsing
- Translation
- Named entity recognition
- Summarization
- Generation

Sequence modeling: Variable length sequential inputs and/or outputs

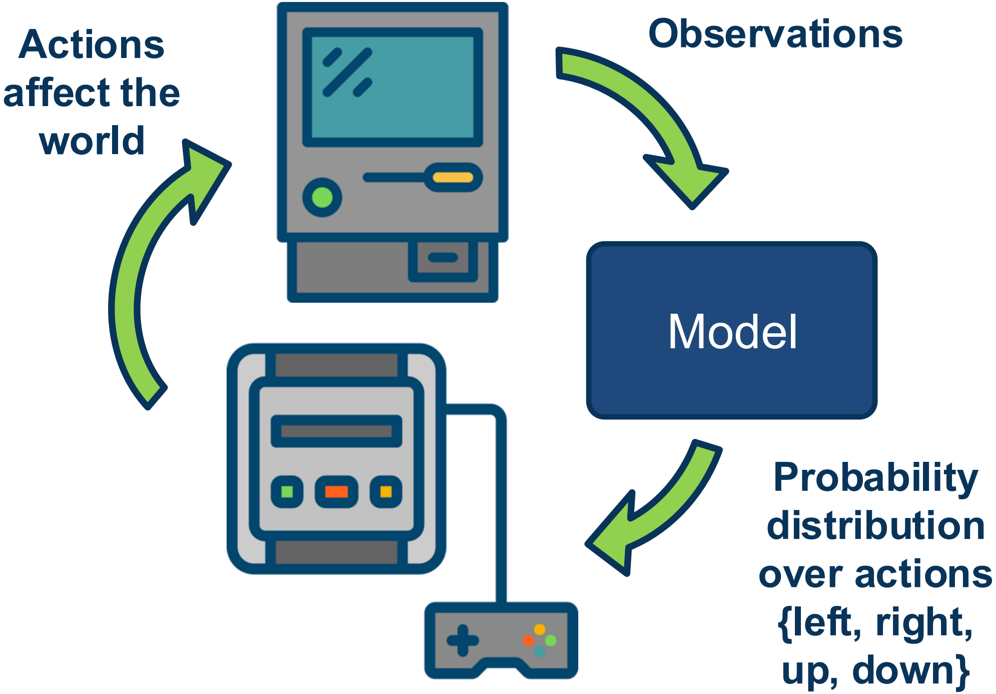
Recent progress: Large-scale Language Models

Application: Decision Making

- Sequence of inputs/outputs
- Actions affect the environment

Examples: Chess / Go, Video Games, Recommendation Systems, Web Agents …

Example: Video Game



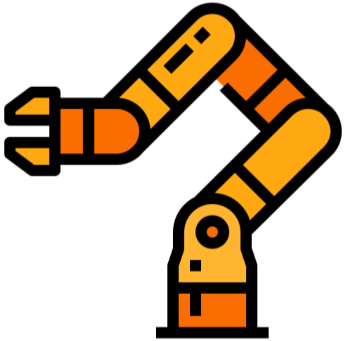
Robotics involves a **combination of AI/ML techniques**:

- **Sense:** Perception
- **Plan:** Planning
- **Act:** Controls

Some things are **learned (perception)**, while others **programmed**

An evolving landscape

Application:



Example: Robotics

Rest of the lecture (also next lecture):

- **Types of Machine Learning Problems**
- **Parametric Models**
- **Linear Classifiers**
- **Gradient Descent**

Supervised Learning

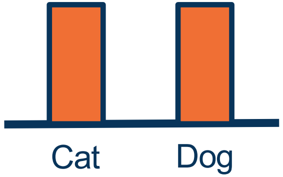
Unsupervised Learning

Reinforcement Learning

Supervised Learning

- Train Input: $\{X, Y\}$
- Learning output: $f : X \to Y$
- Usually f is a **distribution**, e.g. $P(y|x)$

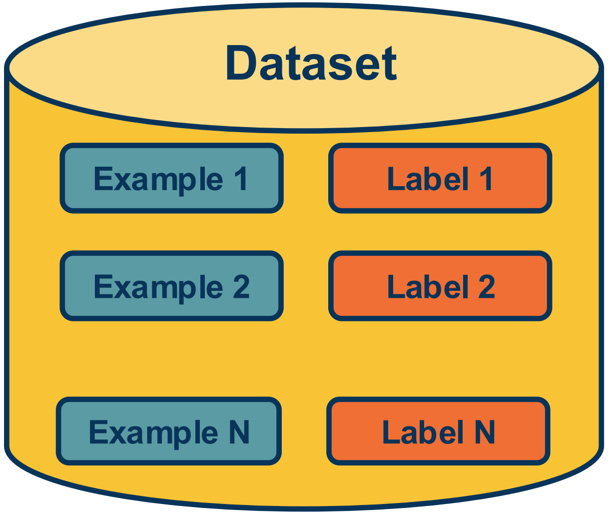
https://en.wikipedia.org/wiki/CatDog



Dataset

$X = \{x_1, x_2, \dots, x_N\}$ *where* $x \in \mathbb{R}^d$ **Examples**

$Y = \{y_1, y_2, \dots, y_N\}$ *where* $y \in \mathbb{R}^c$ **Labels**



Supervised Learning

- Train Input: $\{X, Y\}$
- Learning output: $f : X \rightarrow Y$, e.g. $p(y|x)$

Terminology:

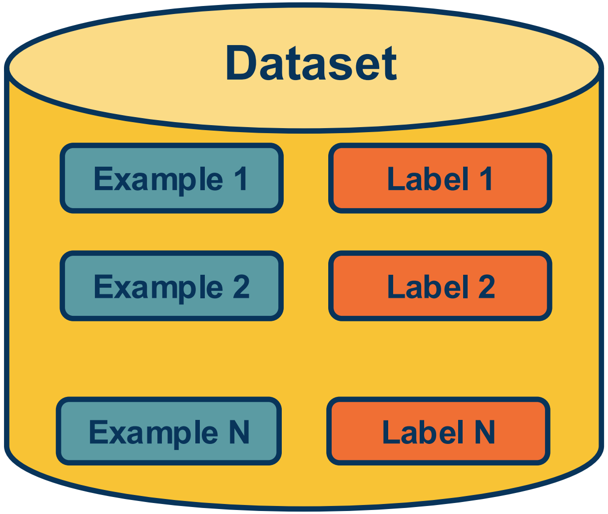
- Model / Hypothesis Class
  - $H: \{f: X \rightarrow Y\}$
  - Learning is search in hypothesis space

E.g., $H = \{f(x) = w^T x \,|w \in \mathbb{R}^d\}$

Dataset

$X = \{x_1, x_2, \dots, x_N\}$ $where\ x \in \mathbb{R}^d$ **Examples**

$Y = \{y_1, y_2, \dots, y_N\}$ $where\ y \in \mathbb{R}^c$ **Labels**

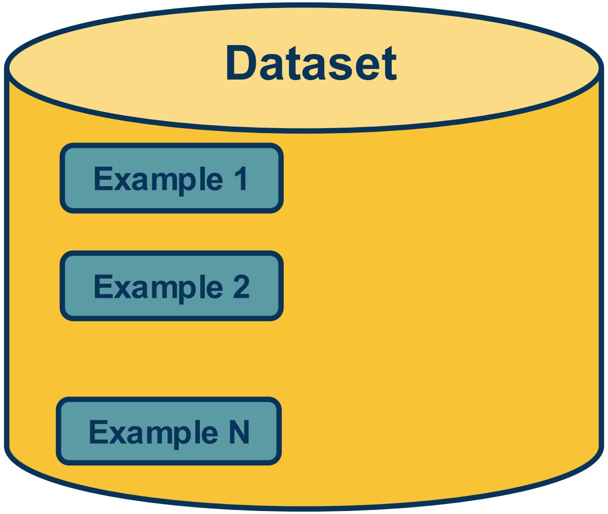


Unsupervised Learning

- Input: $\{X\}$
- Learning output: $p_{data}(x)$
- How likely is x under p_{data}?
- Can we sample from p_{data}?
- Example: Clustering, density estimation, generative modeling, …

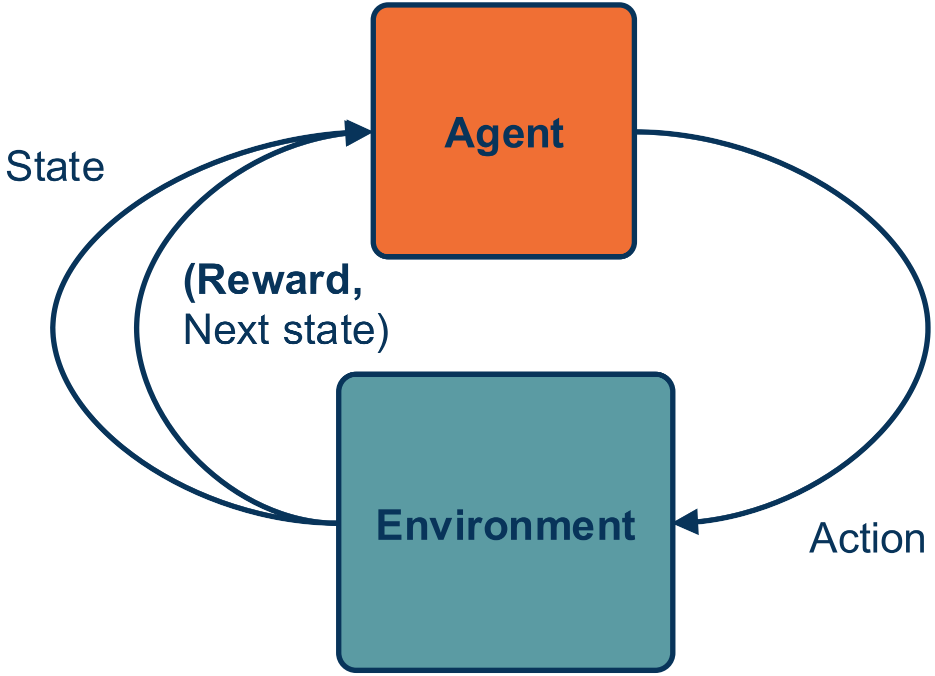
Dataset

$X = \{x_1, x_2, \dots, x_N\}$ $where\ x \in \mathbb{R}^d$ **Examples**



Reinforcement Learning

- Supervision in the form of **reward**
- No supervision on what action to take, but the **expected outcome,** e.g., control a robot to run fast.



Adapted from: http://cs231n.stanford.edu/slides/2020/lecture_17.pdf

Supervised Learning

- Train Input: $\{X, Y\}$
- Learning output: $f : X \rightarrow Y$, e.g. $P(y|x)$

Unsupervised Learning

- Input: $\{X\}$
- Learning output: $P(x)$
- Example: Clustering, density estimation, etc.

Reinforcement Learning

- Supervision in form of **reward**
- No supervision on what action to take

Very often combined, sometimes within the same model!

Types of Machine Learning

What actually defines a machine learning algorithm?

- Model / Hypothesis Class
  - $H: \{f: X \rightarrow Y\}$
  - Learning is search in hypothesis space E.g., $H = \{f(x) = w^T x \mid w \in \mathbb{R}^d\}$

$$\mathcal{D} = \{(x_i, y_i)\}_{i=1}^N, \qquad f^* = \arg\min_{f \in \mathcal{H}} \frac{1}{N} \sum_i \ell(f(x_i), y_i)$$

- It is a combination of:

$$\boxed{\text{data} + \text{hypothesis class} + \text{loss} + \text{optimization}}$$

Questions:

- If H contains **every possible function**, can I get zero training error?
  - No, consider training data {(x=1, y=2), (x=1, y=3)}
- If H contains **only linear functions**, what am I assuming about the world?
  - You are imposing the inductive bias that the input-output relationship can be captured by a linear decision rule:

Rest of the lecture (also next lecture):

- Types of Machine Learning Problems
- **Parametric Models**
- Linear Classifiers
- Gradient Descent

(Recap) Building A Complicated Function

Given a library of simple functions

$\sin(x)$

$\log(x)$

$\cos(x)$

x^3

$\exp(x)$

Compose into a

complex function

Idea 3: Layer Composition

Compose a set of layers **(parametric functions)** through which the input data get transformed.

More layers = "Deeper"

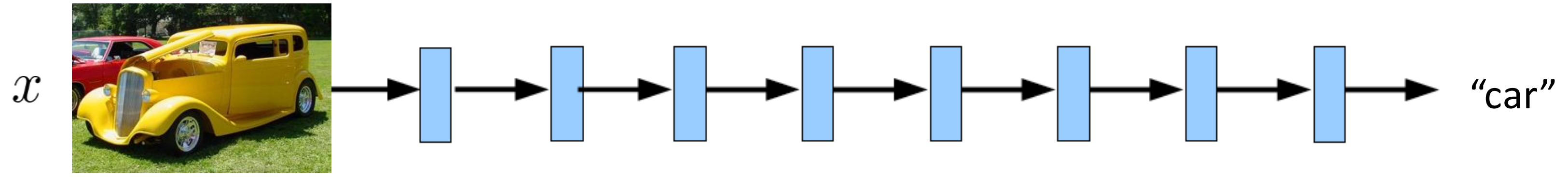
Production models today have **billions** of learnable parameters

Parametric functions

$$f_\theta(x) = \boxed{g_{\theta_n}}(\ldots g_{\theta_2}(g_{\theta_1}(x)\ldots))$$

Linear Layer:
$g(x) = Ax + b$

Nonlinear Layer:
$g(x) = \max(0, x)$



Slide Credit: Marc'Aurelio Ranzato, Yann LeCun

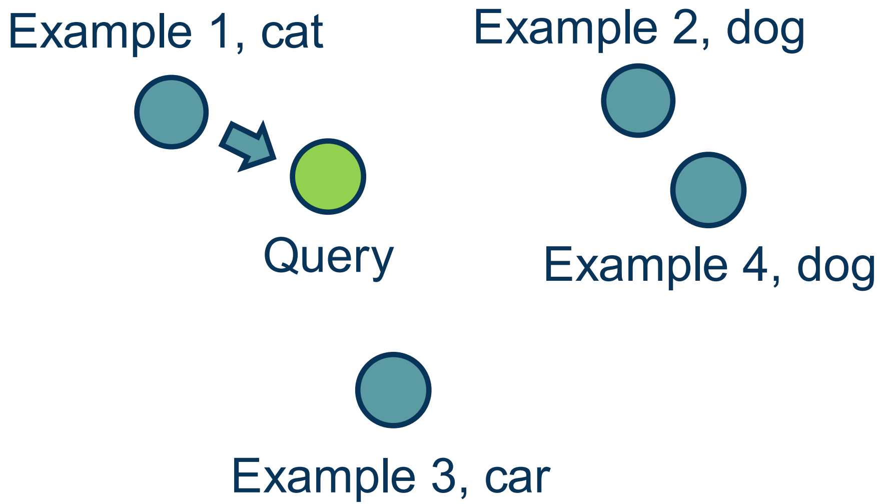
Non-Parametric Model

No explicit model for the function, **examples**:

- **Nearest neighbor classifier**
- Decision tree

Hypothesis class changes with the number of data points

Non-Parametric – Nearest Neighbor



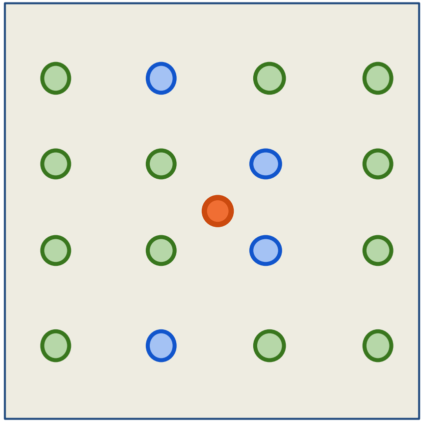
Procedure: Take label of nearest example

k-Nearest Neighbor on high-dimensional data (e.g., images) is *almost never* used.

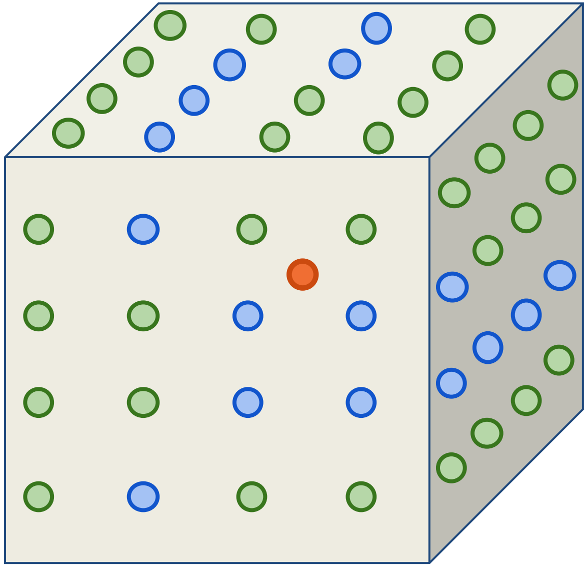
Curse of dimensionality:
Each added dimension creates the **opportunity** for two data points to be far away from each other.

Dimensions = 1
Points = 4

Dimensions = 2
Points = 4^2



Dimensions = 3
Points = 4^3



Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n

Problems with Instance-Based Learning

- **Curse of Dimensionality**
  - Data required increases exponentially with the number of dimensions
- **Doesn’t work well when large number of irrelevant features**
  - Distances overwhelmed by noisy features
- **Expensive**
  - No Learning: most real work done during testing
  - For every test sample, must search through all dataset – very slow!
  - Must use tricks like approximate nearest neighbor search

Parametric Model

Explicitly model the function $f : X \to Y$ in the form of a parametrized function $f(x, W) = y$, **examples**:

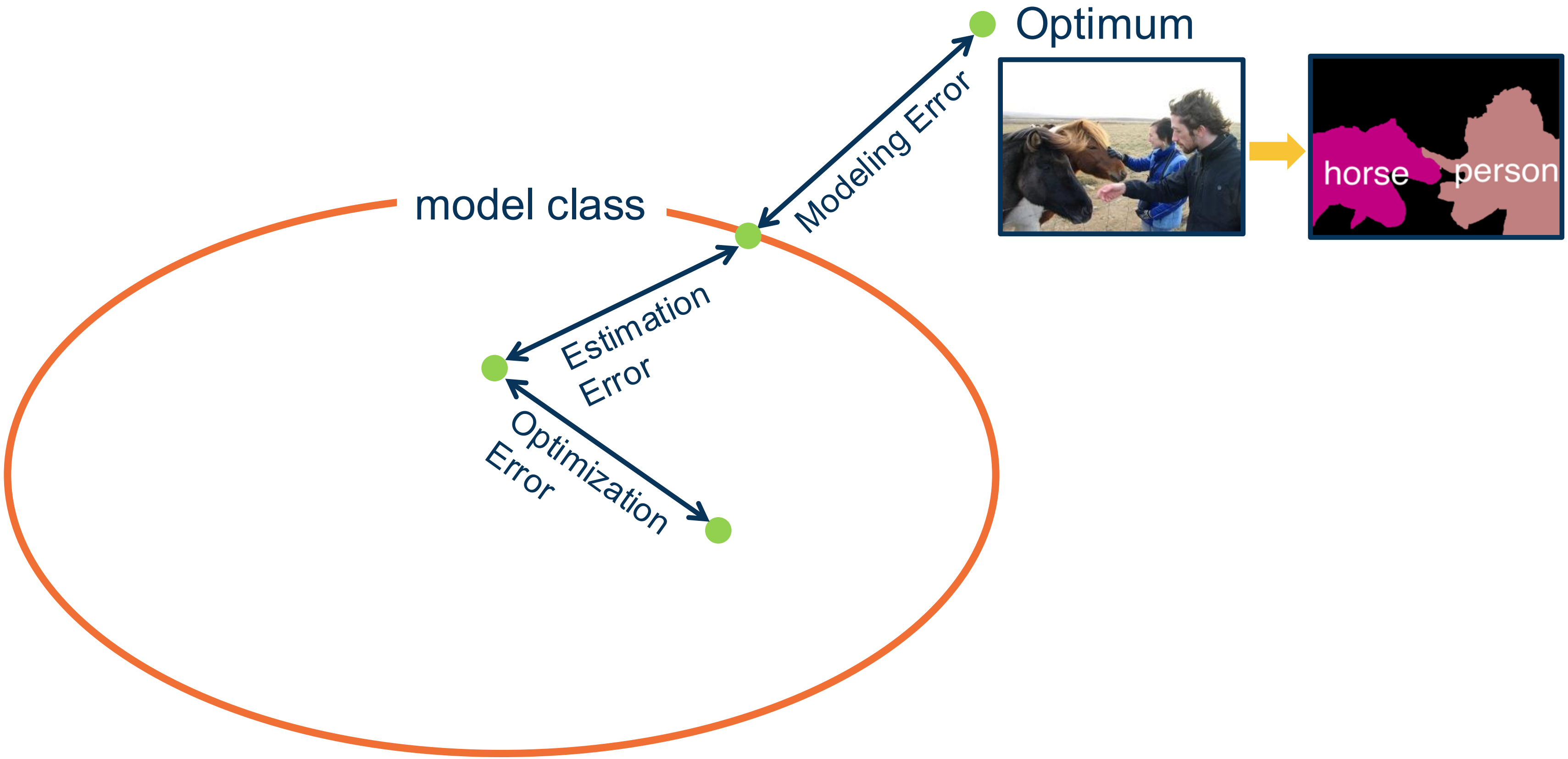
- Linear classifier
  - Number of parameters grows **linearly** with the number of dimensions!
- Neural networks

Parametric – Linear Classifier

$$f(x, W) = Wx + b$$

Q: How many parameters to classify ***N***-dimensional data?
A: N + 1

Hypothesis classes doesn't change: we are simply searching for the optimal value for each parameter



From: slides by Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n

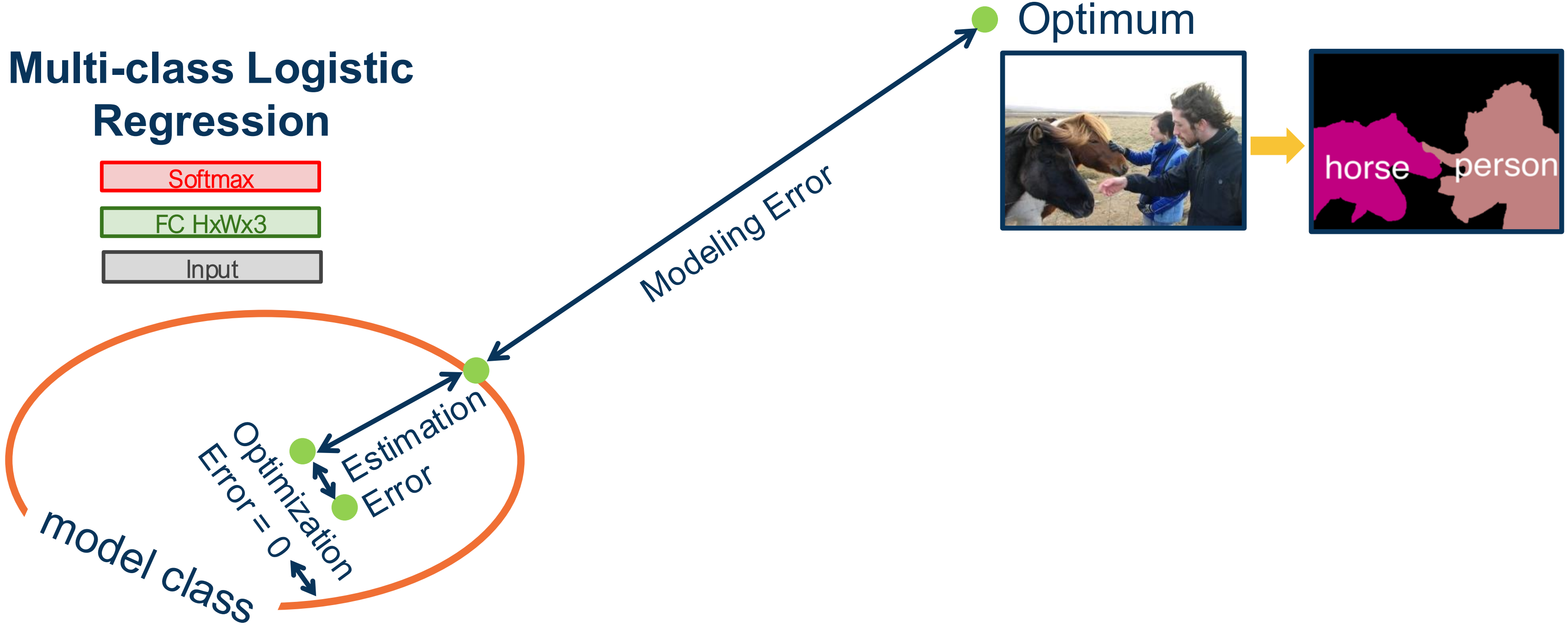
Types of Errors and Generalization

Multi-class Logistic Regression

Softmax

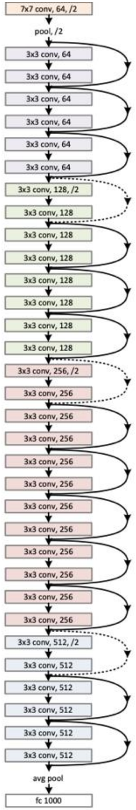
FC HxWx3

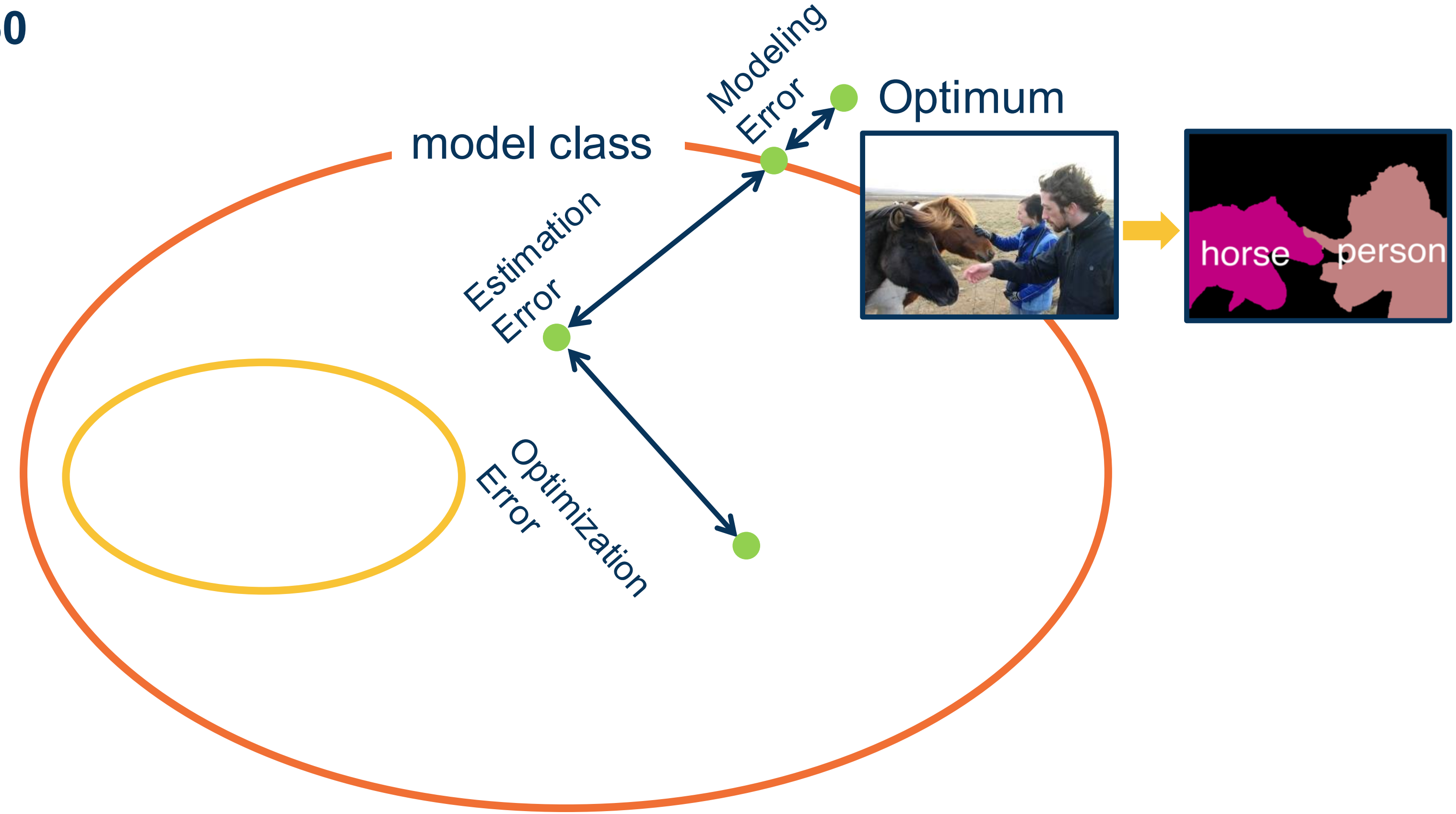
Input



From: slides by Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n

ResNet50





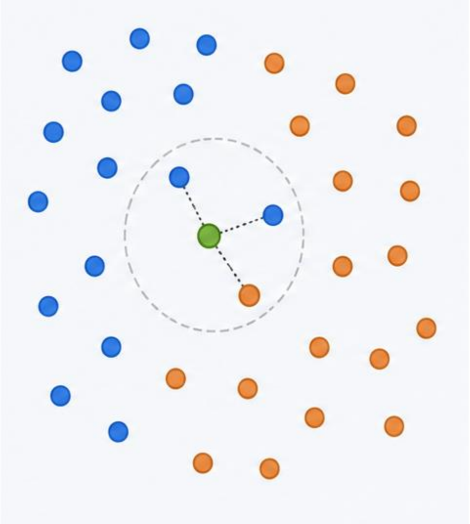
From: slides by Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n

Types of Errors and Generalization

Model classes and assumptions

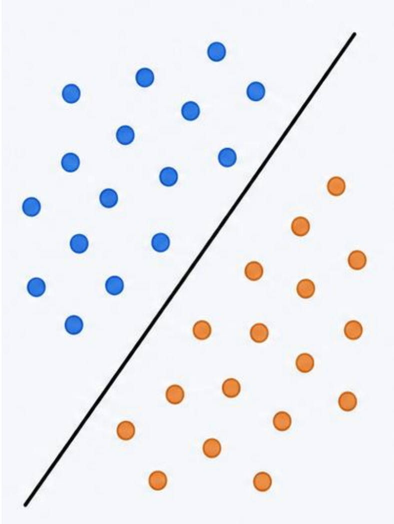
Each model class has its own **inductive biases / assumptions** of the data domain

K-NN



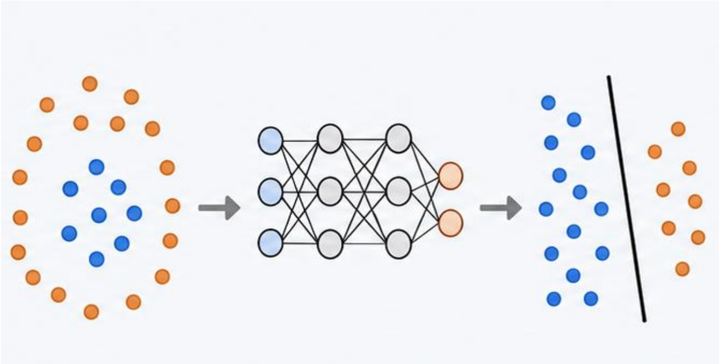
Nearby examples should have similar labels.

Linear Classifier



A single global linear rule can separate the classes.

Neural Nets

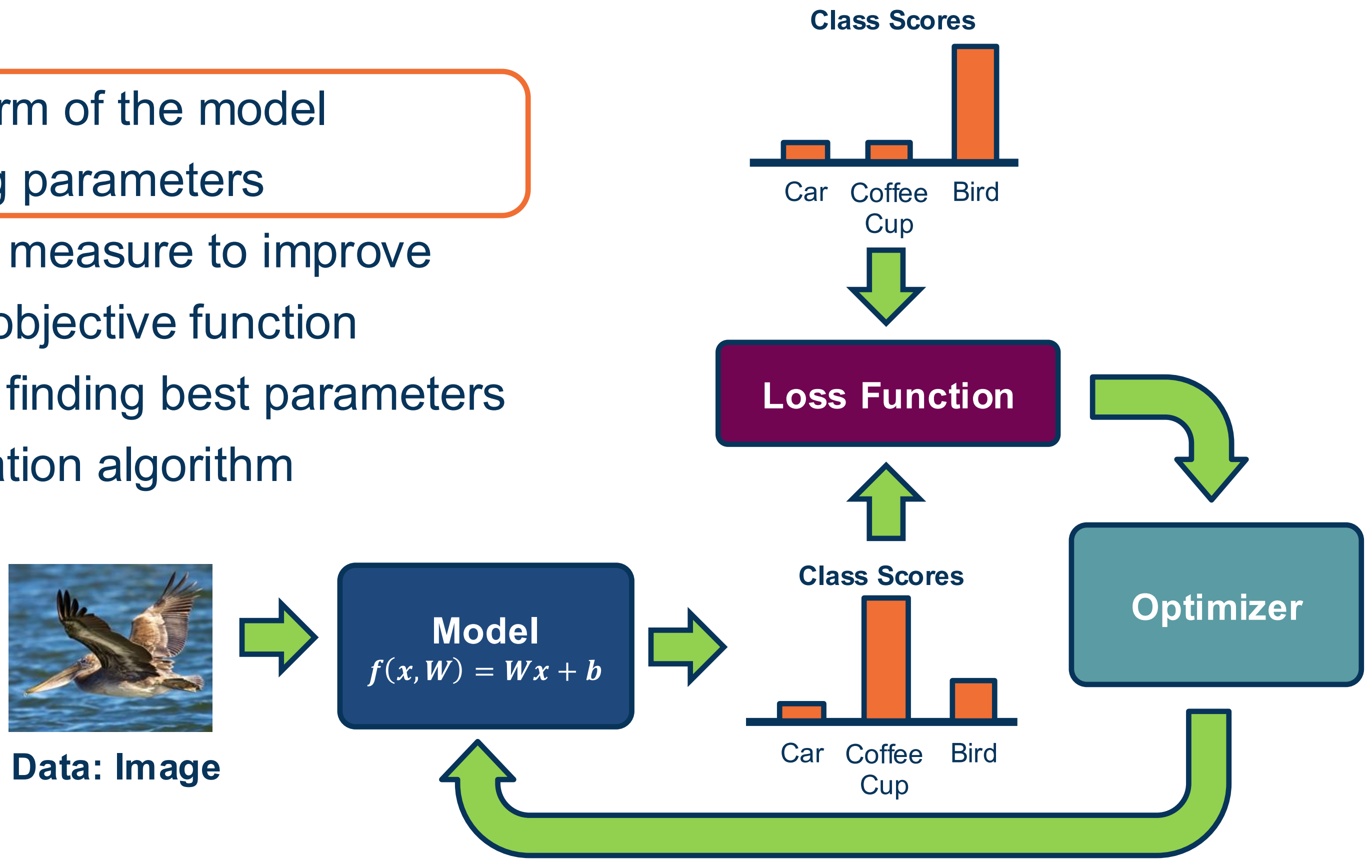


Can learn a representation space where the task becomes simple.

Rest of the lecture (also next lecture):

- Types of Machine Learning Problems
- Parametric Models
- **Linear Classifiers**
- Gradient Descent

- Input
- Functional form of the model
  - Including parameters
- Performance measure to improve
  - Loss or objective function
- Algorithm for finding best parameters
  - Optimization algorithm



Components of a Parametric Model

What is the **simplest function** you can think of?

Our model is:

$$f(x, w) = w \cdot x + b$$

Classifier Result — f

Weights — w

Input — x

Bias (scalar) — b

(Note if $\boldsymbol{w}$ and $\mathbf{x}$ are column vectors we often show this as $\boldsymbol{w}^T\boldsymbol{x}$)

Simple Function

Linear Classification and Regression

Simple linear classifier:

- Calculate score:
  $$f(x, w) = w \cdot x + b$$
- Binary classification rule (w is a vector):
  $$y = \begin{cases} 1 & \text{if } f(x, w) >= 0 \\ 0 & \text{otherwise} \end{cases}$$
- For multi-class classifier take class with highest (max) score
  $$f(x, W) = Wx + b$$

Data: Image

Model

$$f(x, W) = Wx + b$$

Class Scores

$$x = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1n} \\ x_{21} & x_{22} & \cdots & x_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{nn} \end{bmatrix}$$

Flatten

$$x = \begin{bmatrix} x_{11} \\ x_{12} \\ \vdots \\ x_{21} \\ x_{22} \\ \vdots \\ x_{n1} \\ \vdots \\ x_{nn} \end{bmatrix}$$

To simplify notation we will refer to inputs as $x_1 \cdots x_m$ where $m = n \times n$

Model
$$f(x, W) = Wx + b$$

Classifier for class 1
Classifier for class 2
Classifier for class 3

$$\begin{bmatrix} w_{11} & w_{12} & \cdots & w_{1m} \\ w_{21} & w_{22} & \cdots & w_{2m} \\ w_{31} & w_{32} & \cdots & w_{3m} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_m \end{bmatrix} + \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}$$

$\boldsymbol{W}$ $\boldsymbol{x}$ $\boldsymbol{b}$

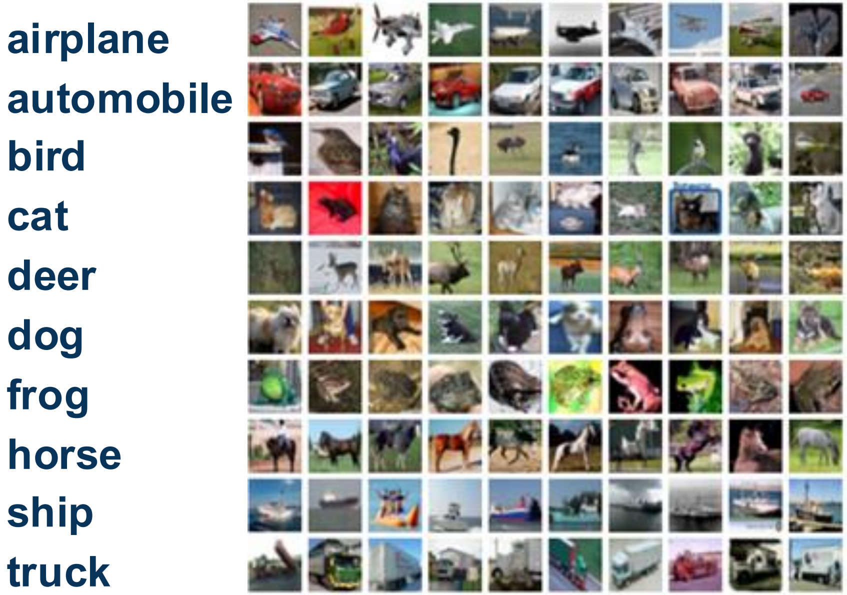
(Note that in practice, implementations can use xW instead, assuming a different shape for W. That is just a different convention and is equivalent.)

Weights

- We can move the bias term into the weight matrix, and a "1" at the end of the input
- Results in **one matrix-vector multiplication**!

Model
$$f(x, W) = Wx + b$$

$$\underbrace{\begin{bmatrix} w_{11} & w_{12} & \cdots & w_{1m} & b_1 \\ w_{21} & w_{22} & \cdots & w_{2m} & b_2 \\ w_{31} & w_{32} & \cdots & w_{3m} & b_3 \end{bmatrix}}_{\boldsymbol{W}} \underbrace{\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_m \\ 1 \end{bmatrix}}_{\boldsymbol{x}}$$

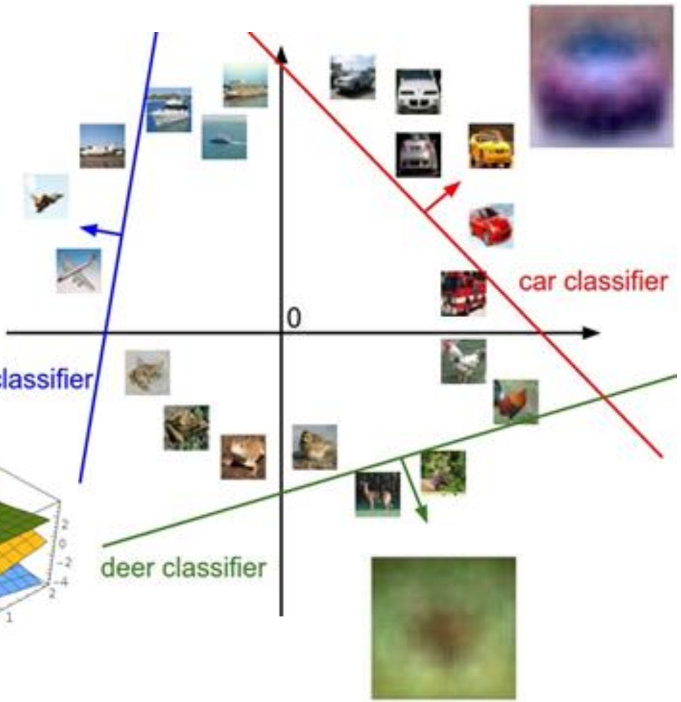


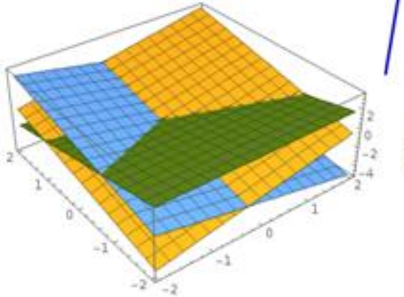
Visual Viewpoint

We can convert the weight vector back into the shape of the image and visualize

Adapted from slides by Fei-Fei Li, Justin Johnson, Serena Yeung, from CS 231n

Geometric Viewpoint





Plot created using Wolfram Cloud

$$f(x, W) = Wx + b$$

Recall: signed distance from point to plane

$$\frac{ax_1 + bx_2 + cx_3 + d}{\sqrt{a^2 + b^2 + c^2}}$$

Adapted from slides by Fei-Fei Li, Justin Johnson, Serena Yeung, from CS 231n

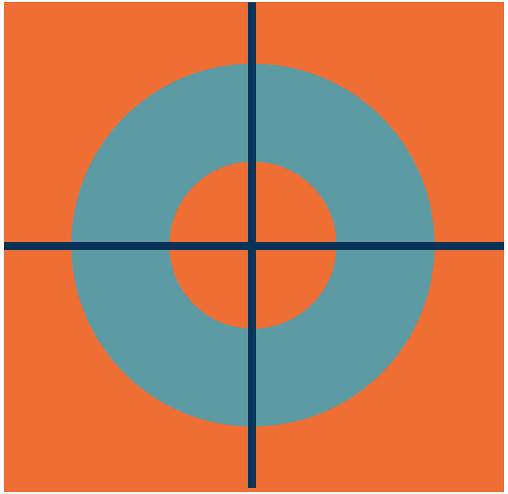
Hard Cases for a Linear Classifier

Class 1:
number of pixels > 0 odd

Class 2:
number of pixels > 0 even

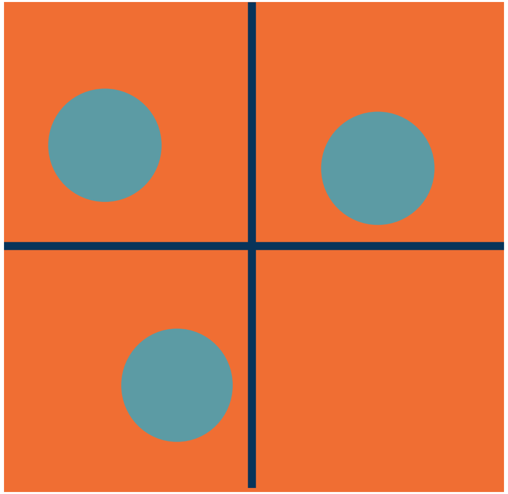
Class 1:
1 < = L2 norm < = 2

Class 2:
Everything else



Class 1:
Three modes

Class 2:
Everything else



Adapted from slides by Fei-Fei Li, Justin Johnson, Serena Yeung, from CS 231n

Neural Network

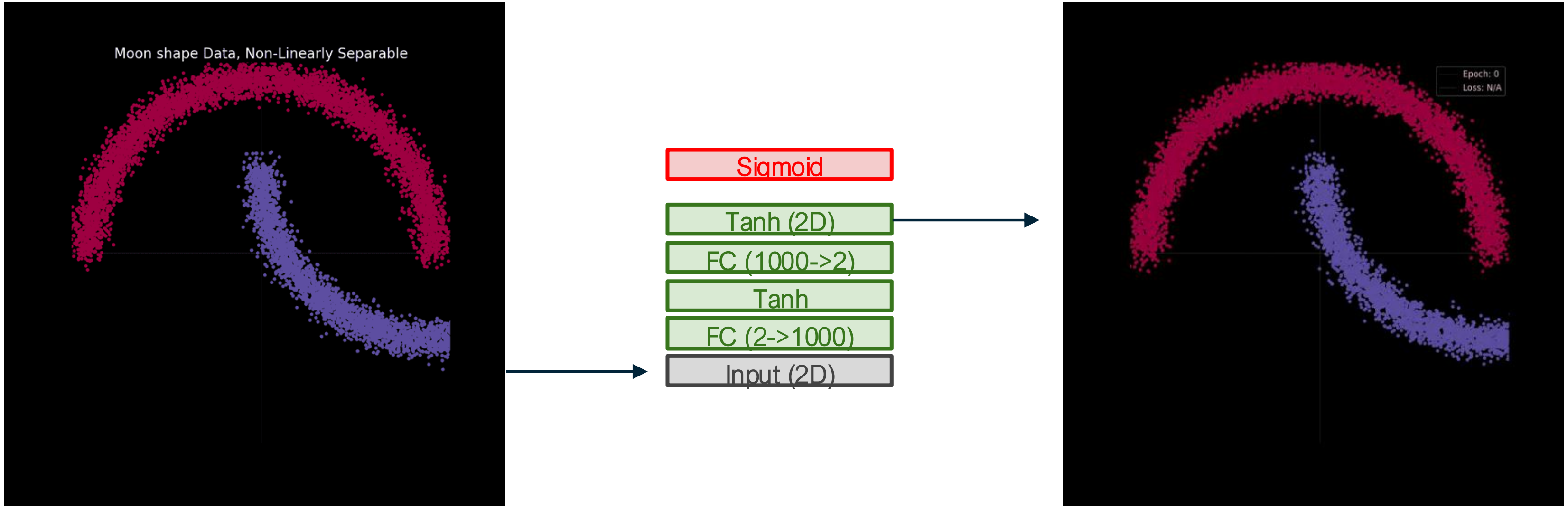
Linear classifier

This image is CC0 1.0 public domain

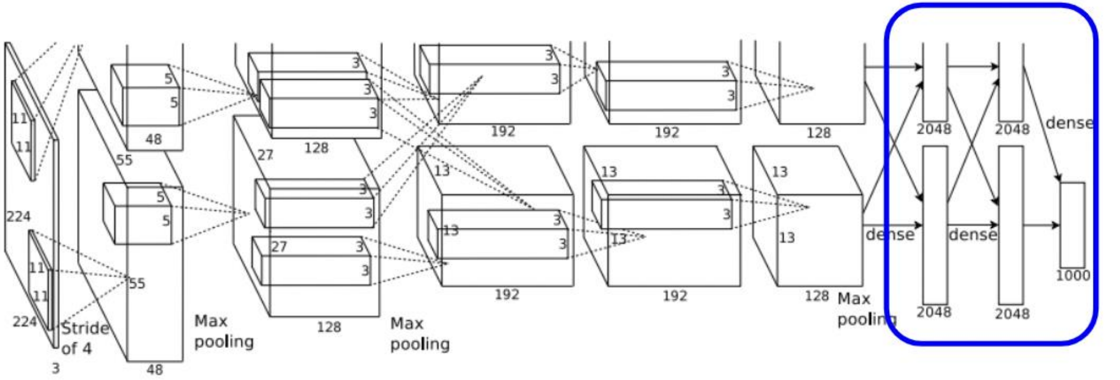
Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n

(Deep) Representation Learning for Classification

A function that transforms raw data space into a linearly-separable space

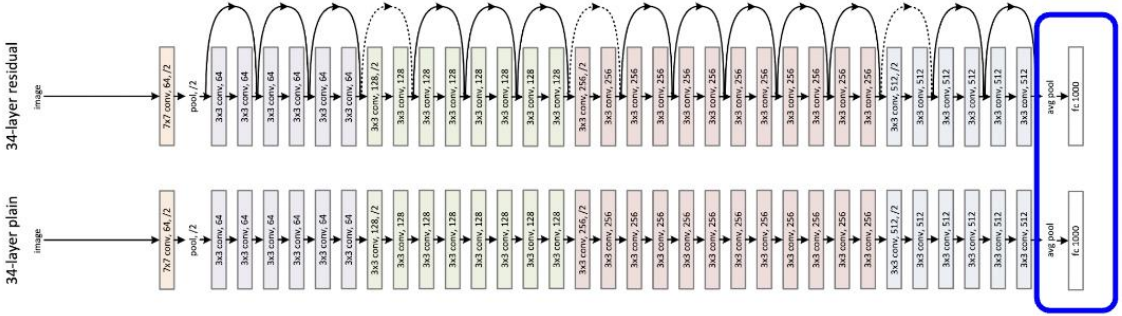


https://khalidsaifullaah.github.io/neural-networks-from-linear-algebraic-perspective



[Krizhevsky et al. 2012]

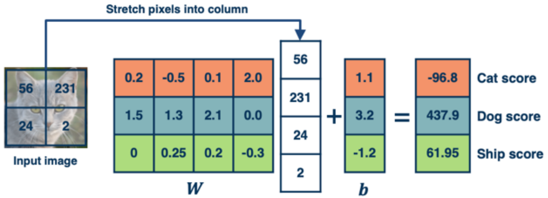
Linear layers



[He et al. 2015]

Algebraic Viewpoint

$$f(x, W) = Wx$$

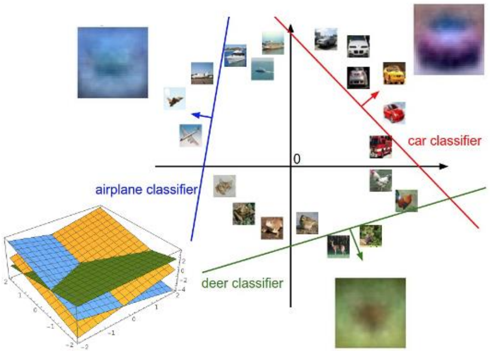


Visual Viewpoint

One template per class

Geometric Viewpoint

Hyperplanes cutting up space



Adapted from slides by Fei-Fei Li, Justin Johnson, Serena Yeung, from CS 231n

Next time:

- Input (and representation)
- Functional form of the model
  - Including parameters
- **Performance measure to improve**
  - **Loss or objective function**
- Algorithm for finding best parameters
  - Optimization algorithm