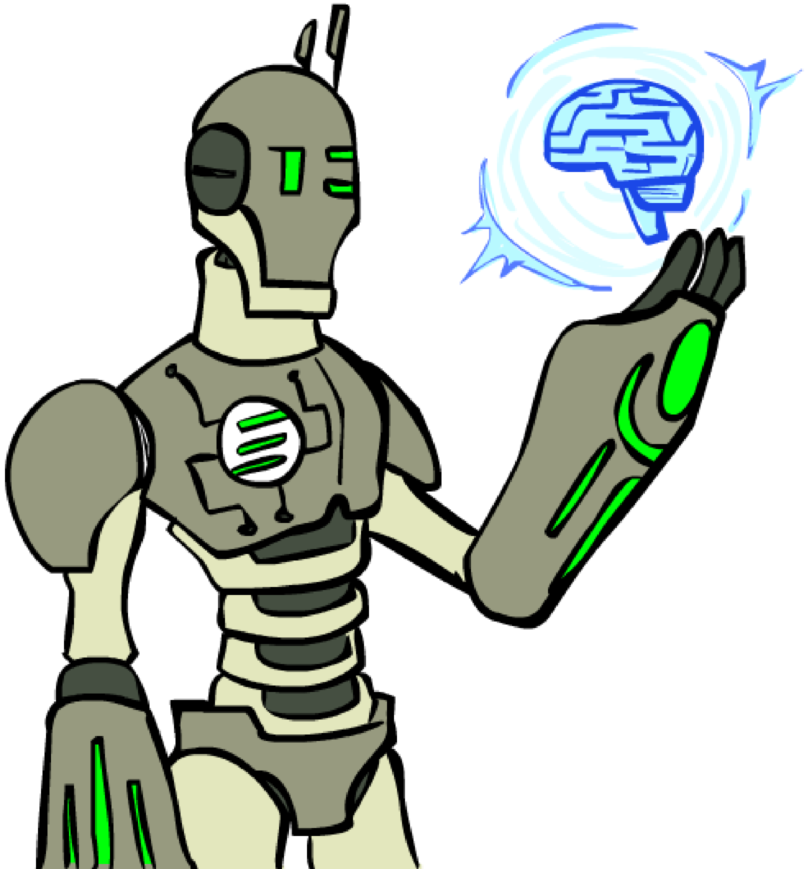
As you walk in

cs.cmu.edu/~10607

Welcome!

1) Sit at a table next to another student
2) Make name plate
   - Fold paper in half
   - Write preferred name
   - Below write you favorite fictional AI/robot



Images: ai.berkeley.edu

10-607 Computational Foundations for Machine Learning

Instructor: Pat Virtue

Today

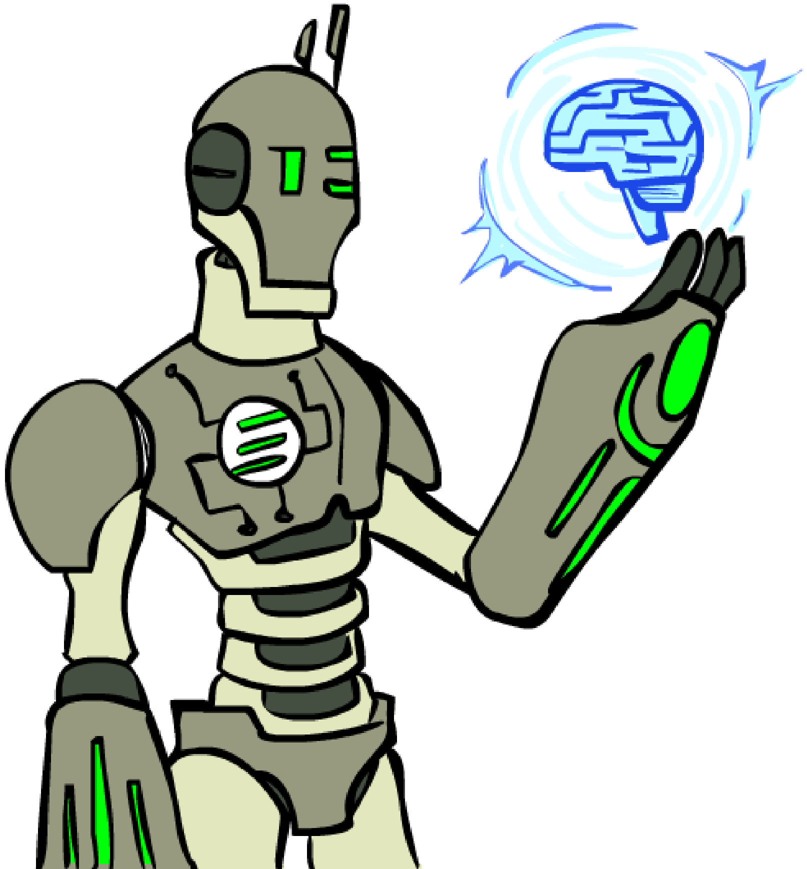
Course Info

Warm-up exercise

Propositional Logic and Proofs

ML and 606/607 Intro

More Course Info



Images: ai.berkeley.edu

Course Team

Instructor

Pat
Virtue
pvirtue

Teaching Assistants

Kellen
Gibson
kagibson

Ian
Char
ichar

Course Team

Students!!

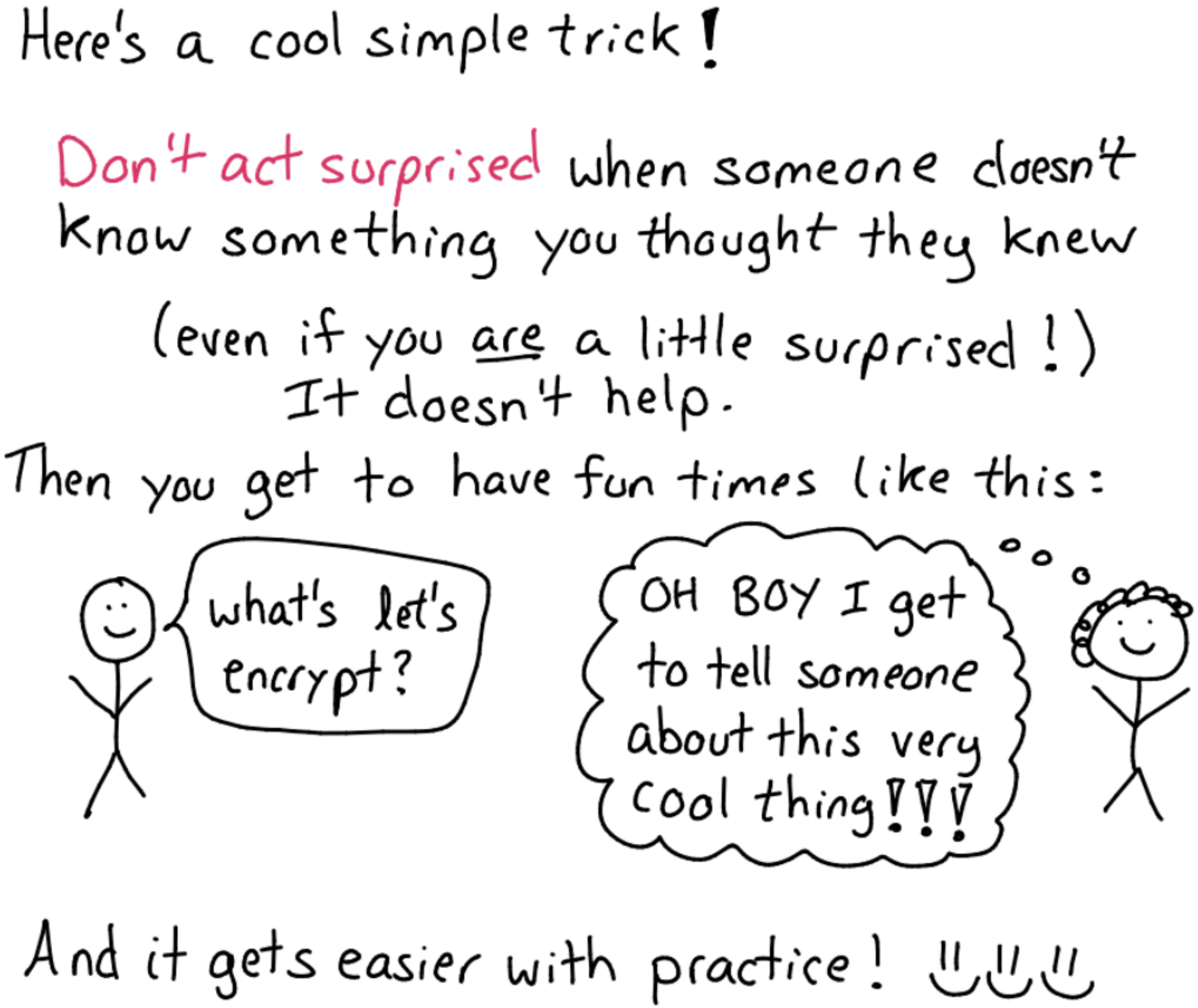
Team Tips

Try not to act surprised

https://jvns.ca/blog/2017/04/27/no-feigning-surprise/

Team Tips

Try not to act surprised



https://jvns.ca/blog/2017/04/27/no-feigning-surprise/

Two-column Proof

A: Socrates is human
$A \Rightarrow B$: human the mortal

Give an explicit justification for each statement based on previous statements

Prove Socrates is mortal

	Statement	Justification
1	A	Assumption
2	$A \Rightarrow B$	Assumption
3	B	Modus Ponens

(1) & (2)

Notation Alert!

Modus Ponens

$$\frac{\alpha \Rightarrow \beta, \quad \alpha}{\beta}$$

Warm-up Exercise

Propositional logic inference rules

- *modus ponens*: from premesis p and $p \Rightarrow q$, conclude q
- $\wedge$ introduction: if we separately prove p and q, then that constitutes a proof of $p \wedge q$.
- $\wedge$ elimination: from $p \wedge q$ we can conclude either of p and q separately.
- $\vee$ introduction: from p we can conclude $p \vee q$ for any q.
- $\vee$ elimination (also called proof by cases): if we know $p \vee q$ (the cases) and we have both $p \vee r$ and $p \vee r$ (the case-specific proofs), then we can conclude r.
- T introduction: we can conclude T from no assumptions.
- F elimination: from F we can conclude an arbitrary formula p.
- Associativity: both $\wedge$ and $\vee$ are associative: it doesn't matter how we parenthesize an expression like $a \wedge b \wedge c \wedge d$. (So in fact we often just leave the parentheses out in such cases. But when having $\vee$ and $\wedge$ together, it's a good idea to keep the parentheses.)
- Distributivity: $\wedge$ and $\vee$ distribute over one another; for example, $a \vee (b \wedge c)$ is equivalent to $(a \vee b) \wedge (a \vee c)$ and $a \wedge (b \vee c)$ is equivalent to $(a \wedge b) \vee (a \wedge c)$.
- Commutativity: both $\wedge$ and $\vee$ are commutative (symmetric in the order of their arguments), so we can re-order their arguments however we please. For example, $a \wedge b \wedge c$ is equivalent to $c \wedge b \wedge a$.

Warm-up Exercise

Use the propositional logic inference rules provided to prove:

$$(a \land b) \Rightarrow (b \land a)$$

However, you cannot use the commutativity rule.

Write your proof in two-column format, i.e., give an explicit justification for each statement based on previous statements

1	$a \land b$	Assumption
2	a	$\land$ Elim (1)
3	b	$\land$ Elim (1)
4	$b \land a$	$\land$ Intro (2) (3)
5	$a \land b \Rightarrow b \land a$	$\Rightarrow$ Intro 1-4

Warm-up Exercise

Use the propositional logic inference rules provided to prove:

$$(a \wedge b) \Rightarrow (b \wedge a)$$

However, you cannot use the commutativity rule.

Write your proof in two-column format, i.e., give an explicit justification for each statement based on previous statements

Proof by Cases

Goal is to prove χ

1 $\phi \vee \psi$ Assume

2 Case 1: ϕ

a) ______

b) ______

c) $\phi \Rightarrow \chi$ ______

3 Case 2: ψ

a) ______

b) ______

c) ______

d) $\psi \Rightarrow \chi$ ______

4 χ by $\vee$ elimination

Today

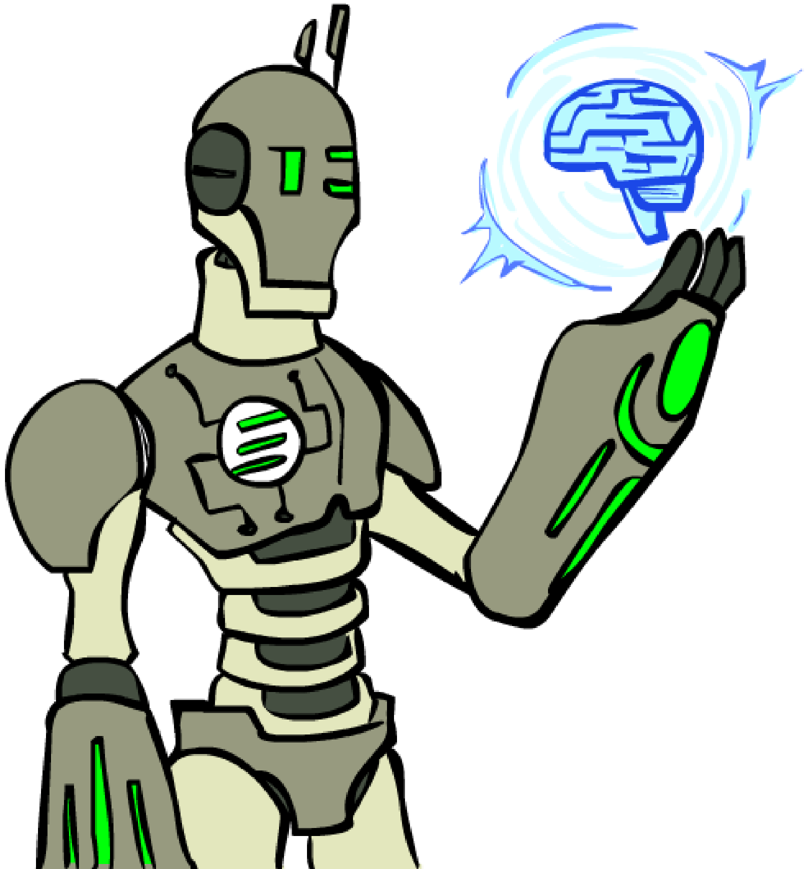
Course Info

Warm-up exercise

Propositional Logic and Proofs

ML and 606/607 Intro

More Course Info



Images: ai.berkeley.edu

Analysis: Perceptron

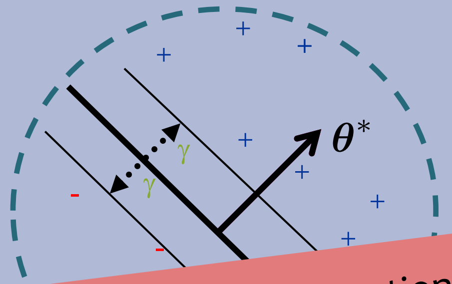
Perceptron Mistake Bound

Theorem 0.1 (Block (1962), Novikoff (1962)).
Given dataset: $\mathcal{D} = \{(\mathbf{x}^{(i)}, y^{(i)})\}_{i=1}^{N}$.
Suppose:

1. *Finite size inputs:* $||x^{(i)}|| \leq R$
2. *Linearly separable data:* $\exists \boldsymbol{\theta}^*$ *s.t.* $||\boldsymbol{\theta}^*|| = 1$ *and* $y^{(i)}(\boldsymbol{\theta}^* \cdot \mathbf{x}^{(i)}) \geq \gamma, \forall i$

Then: The number of mistakes made by the Perceptron algorithm on this dataset is

$$k \leq (R/\gamma)^2$$



Note: This is just motivation – we'll cover the background need to understand these topics later!

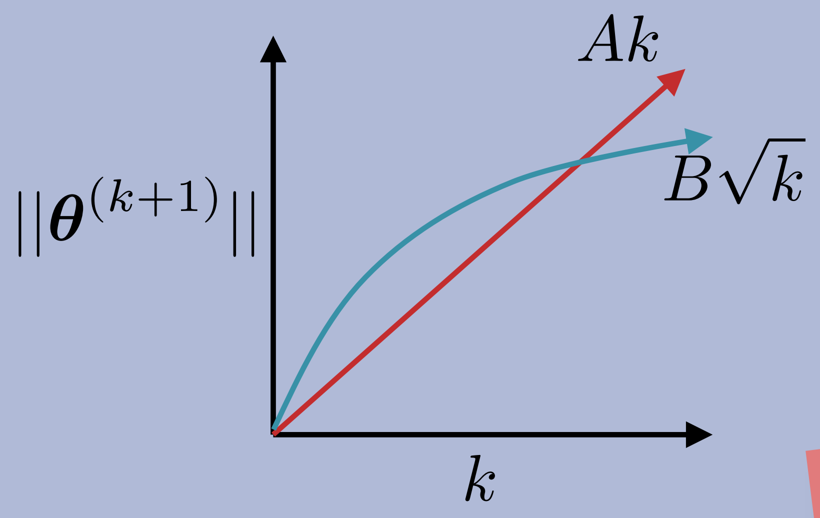
Figure from Nina Balcan

Analysis: Perceptron

Proof of Perceptron Mistake Bound:

We will show that there exist constants A and B s.t.

$$Ak \leq ||\boldsymbol{\theta}^{(k+1)}|| \leq B\sqrt{k}$$



Note: This is just motivation – we'll cover the background need to understand these topics later!

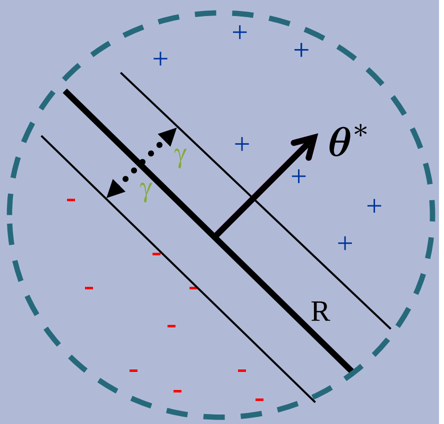
Analysis: Perceptron

Theorem 0.1 (Block (1962), Novikoff (1962)).
Given dataset: $\mathcal{D} = \{(\mathbf{x}^{(i)}, y^{(i)})\}_{i=1}^{N}$.
Suppose:

1. *Finite size inputs:* $||x^{(i)}|| \leq R$
2. *Linearly separable data:* $\exists \boldsymbol{\theta}^*$ s.t. $||\boldsymbol{\theta}^*|| = 1$ *and* $y^{(i)}(\boldsymbol{\theta}^* \cdot \mathbf{x}^{(i)}) \geq \gamma, \forall i$

Then: The number of mistakes made by the Perceptron algorithm on this dataset is

$$k \leq (R/\gamma)^2$$



Algorithm 1 Perceptron Learning Algorithm (Online)

```
procedure PERCEPTRON(D = {(x^(1), y^(1)), (x^(2), y^(2)), ...})
    θ ← 0, k = 1                              ▷ Initialize parameters
    for i ∈ {1, 2, ...} do                    ▷ For each example
        if y^(i)(θ^(k) · x^(i)) ≤ 0 then      ▷ If mistak
            θ^(k+1) ← θ^(k) + y^(i) x^(i)     ▷ Update param
            k ← k + 1
    return θ
```

Note: This is just motivation – we'll cover the background need to understand these topics later!

Analysis: Perceptron

Proof of Perceptron Mistake Bound:

Part 1: for some A, $Ak \leq ||\boldsymbol{\theta}^{(k+1)}||$

$$\boldsymbol{\theta}^{(k+1)} \cdot \boldsymbol{\theta}^* = (\boldsymbol{\theta}^{(k)} + y^{(i)}\mathbf{x}^{(i)})\boldsymbol{\theta}^*$$

by Perceptron algorithm update

$$= \boldsymbol{\theta}^{(k)} \cdot \boldsymbol{\theta}^* + y^{(i)}(\boldsymbol{\theta}^* \cdot \mathbf{x}^{(i)})$$

$$\geq \boldsymbol{\theta}^{(k)} \cdot \boldsymbol{\theta}^* + \gamma$$

by assumption

$$\Rightarrow \boldsymbol{\theta}^{(k+1)} \cdot \boldsymbol{\theta}^* \geq k\gamma$$

by induction on k since $\theta^{(1)} = \mathbf{0}$

$$\Rightarrow ||\boldsymbol{\theta}^{(k+1)}|| \geq k\gamma$$

since $\underbrace{||\mathbf{w}|| \times ||\mathbf{u}|| \geq \mathbf{w} \cdot \mathbf{u}}_{\text{Cauchy-Schwartz inequality}}$ and $||\theta^*|| = 1$

Note: This is just motivation – we'll cover the background need to understand these topics later!

Analysis: Perceptron

Proof of Perceptron Mistake Bound:

Part 2: for some B, $||\boldsymbol{\theta}^{(k+1)}|| \leq B\sqrt{k}$

$$||\boldsymbol{\theta}^{(k+1)}||^2 = ||\boldsymbol{\theta}^{(k)} + y^{(i)}\mathbf{x}^{(i)}||^2$$

by Perceptron algorithm update

$$= ||\boldsymbol{\theta}^{(k)}||^2 + (y^{(i)})^2||\mathbf{x}^{(i)}||^2 + 2y^{(i)}(\boldsymbol{\theta}^{(k)} \cdot \mathbf{x}^{(i)})$$

$$\leq ||\boldsymbol{\theta}^{(k)}||^2 + (y^{(i)})^2||\mathbf{x}^{(i)}||^2$$

since kth mistake $\Rightarrow y^{(i)}(\boldsymbol{\theta}^{(k)} \cdot \mathbf{x}^{(i)}) \leq 0$

$$= ||\boldsymbol{\theta}^{(k)}||^2 + R^2$$

since $(y^{(i)})^2||\mathbf{x}^{(i)}||^2 = ||\mathbf{x}^{(i)}||^2 = R^2$ by assumption and $(y^{(i)})^2 = 1$

$$\Rightarrow ||\boldsymbol{\theta}^{(k+1)}||^2 \leq kR^2$$

by induction on k since $(\theta^{(1)})^2 = 0$

$$\Rightarrow ||\boldsymbol{\theta}^{(k+1)}|| \leq \sqrt{k}R$$

Note: This is just motivation – we'll cover the background need to understand these topics later!

Analysis: Perceptron

Proof of Perceptron Mistake Bound:

Part 3: Combining the bounds finishes the proof.

$$k\gamma \leq ||\boldsymbol{\theta}^{(k+1)}|| \leq \sqrt{k}R$$

$$\Rightarrow k \leq \underbrace{(R/\gamma)^2}_{\text{The total number of mistakes must be less than this}}$$

Note: This is just motivation – we'll cover the background need to understand these topics later!

Logic Language

Natural language?

Propositional logic

- Syntax: $P \lor (\neg Q \land R)$; $X_1 \Leftrightarrow (\text{Raining} \Rightarrow \text{Sunny})$
- Possible world: {P=true, Q=true, R=false, S=true} or 1101
- Semantics: $\alpha \land \beta$ is true in a world iff is α true and β is true (etc.)

First-order logic

- Syntax: $\forall x\ \exists y\ P(x,y) \land \neg Q(Joe,f(x)) \Rightarrow f(x)=f(y)$
- Possible world: Objects o_1, o_2, o_3; P holds for <o_1,o_2>; Q holds for <o_3>; $f(o_1)=o_1$; $Joe=o_3$; etc.
- Semantics: $\phi(\sigma)$ is true in a world if $\sigma=o_j$ and ϕ holds for o_j; etc.

Propositional Logic

Propositional Logic

A	B	$A \Rightarrow B$
T	T	T
T	F	F
F	T	T
F	F	T

Symbol:

- Variable that can be true or false
- We'll try to use capital letters, e.g. A, B, $P_{1,2}$
- Often include True and False

Operators:

- $\neg A$: not A
- $A \wedge B$: A and B (conjunction)
- $A \vee B$: A or B (disjunction) Note: this is not an "exclusive or"
- $A \Rightarrow B$: A implies B (implication). If A then B
- $A \Leftrightarrow B$: A if and only if B (biconditional)

Sentences

Poll 1

If we know that $A \vee B$ and $\neg B \vee C$ are true,

what do we know about $A \vee C$?

i. $A \vee C$ is guaranteed to be true
ii. $A \vee C$ is guaranteed to be false
iii. We don't have enough information to say anything definitive about $A \vee C$

Poll 1

If we know that $A \vee B$ and $\neg B \vee C$ are true, what do we know about $A \vee C$?

A	B	C	$A \vee B$	$\neg B \vee C$	$A \vee C$
false	false	false	false	true	false
false	false	true	false	true	true
false	true	false	true	false	false
false	true	true	true	true	true
true	false	false	true	true	true
true	false	true	true	true	true
true	true	false	true	false	true
true	true	true	true	true	true

Poll 1

If we know that $A \vee B$ and $\neg B \vee C$ are true, what do we know about $A \vee C$?

A	B	C	$A \vee B$	$\neg B \vee C$	$A \vee C$
false	false	false	false	true	false
false	false	true	false	true	true
false	true	false	true	false	false
false	true	true	true	true	true
true	false	false	true	true	true
true	false	true	true	true	true
true	true	false	true	false	true
true	true	true	true	true	true

Poll 1

If we know that $A \lor B$ and $\neg B \lor C$ are true,

what do we know about $A \lor C$?

i. $A \lor C$ is guaranteed to be true
ii. $A \lor C$ is guaranteed to be false
iii. We don't have enough information to say anything definitive about $A \lor C$

Poll 2

If we know that $A \vee B$ and $\neg B \vee C$ are true,

what do we know about A?

i. A is guaranteed to be true
ii. A is guaranteed to be false
iii. We don't have enough information to say anything definitive about A

Poll 2

If we know that $A \vee B$ and $\neg B \vee C$ are true, what do we know about A?

A	B	C	$A \vee B$	$\neg B \vee C$	$A \vee C$
false	false	false	false	true	false
false	false	true	false	true	true
false	true	false	true	false	false
false	true	true	true	true	true
true	false	false	true	true	true
true	false	true	true	true	true
true	true	false	true	false	true
true	true	true	true	true	true

Poll 2

If we know that $A \vee B$ and $\neg B \vee C$ are true,

what do we know about A?

i. A is guaranteed to be true

ii. A is guaranteed to be false

iii. We don't have enough information to say anything definitive about A

Propositional Logic

Symbol:

- Variable that can be true or false
- We'll try to use capital letters, e.g. A, B, $P_{1,2}$
- Often include True and False

Operators:

- $\neg A$: not A
- $A \wedge B$: A and B (conjunction)
- $A \vee B$: A or B (disjunction) Note: this is not an "exclusive or"
- $A \Rightarrow B$: A implies B (implication). If A then B
- $A \Leftrightarrow B$: A if and only if B (biconditional)

Sentences

Propositional Logic Syntax

Given: a set of proposition symbols $\{X_1, X_2, \ldots, X_n\}$

- (we often add True and False for convenience)

X_i is a sentence

If α is a sentence then $\neg\alpha$ is a sentence

If α and β are sentences then $\alpha \wedge \beta$ is a sentence

If α and β are sentences then $\alpha \vee \beta$ is a sentence

If α and β are sentences then $\alpha \Rightarrow \beta$ is a sentence

If α and β are sentences then $\alpha \Leftrightarrow \beta$ is a sentence

And p.s. there are no other sentences!

Notes on Operators

$\boldsymbol{\alpha} \vee \boldsymbol{\beta}$ is <u>inclusive or</u>, not exclusive

Truth Tables

$\boldsymbol{\alpha} \vee \boldsymbol{\beta}$ is inclusive or, not exclusive

$\boldsymbol{\alpha}$	$\boldsymbol{\beta}$	$\boldsymbol{\alpha} \wedge \boldsymbol{\beta}$
F	F	F
F	T	F
T	F	F
T	T	T

$\boldsymbol{\alpha}$	$\boldsymbol{\beta}$	$\boldsymbol{\alpha} \vee \boldsymbol{\beta}$
F	F	F
F	T	T
T	F	T
T	T	T

Notes on Operators

$\boldsymbol{\alpha} \lor \boldsymbol{\beta}$ is inclusive or, not exclusive

$\boldsymbol{\alpha} \Rightarrow \boldsymbol{\beta}$ is equivalent to $\neg\boldsymbol{\alpha} \lor \boldsymbol{\beta}$

- Says who?

Truth Tables

$\alpha \Rightarrow \beta$ is equivalent to $\neg\alpha \vee \beta$

α	β	$\alpha \Rightarrow \beta$	$\neg\alpha$	$\neg\alpha \vee \beta$
F	F	T	T	T
F	T	T	T	T
T	F	F	F	F
T	T	T	F	T

Notes on Operators

$\alpha \vee \beta$ is inclusive or, not exclusive

$\alpha \Rightarrow \beta$ is equivalent to $\neg\alpha \vee \beta$

- Says who?

$\alpha \Leftrightarrow \beta$ is equivalent to $(\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha)$

- Prove it!

Truth Tables

$\alpha \Leftrightarrow \beta$ is equivalent to $(\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha)$

α	β	$\alpha \Leftrightarrow \beta$	$\alpha \Rightarrow \beta$	$\beta \Rightarrow \alpha$	$(\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha)$
F	F	T	T	T	T
F	T	F	T	F	F
T	F	F	F	T	F
T	T	T	T	T	T

Equivalence: it's true in all models. Expressed as a logical sentence:

$$(\alpha \Leftrightarrow \beta) \Leftrightarrow [(\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha)]$$

Inference Rules

Notation Alert!

Modus Ponens

$$\frac{\alpha \Rightarrow \beta, \quad \alpha}{\beta}$$

Unit Resolution

$$\frac{a \vee b, \quad \neg b \vee c}{a \vee c}$$

General Resolution

$$\frac{a_1 \vee \cdots \vee a_m \vee b, \quad \neg b \vee c_1 \vee \cdots \vee c_n}{a_1 \vee \cdots \vee a_m \vee c_1 \vee \cdots \vee c_n}$$

Propositional Logic

Check if sentence is true in given model

In other words, does the model *satisfy* the sentence?

function PL-TRUE?(α,model) returns true or false

if α is a symbol then return Lookup(α, model)

if Op(α) = $\neg$ then return **not**(PL-TRUE?(Arg1(α),model))

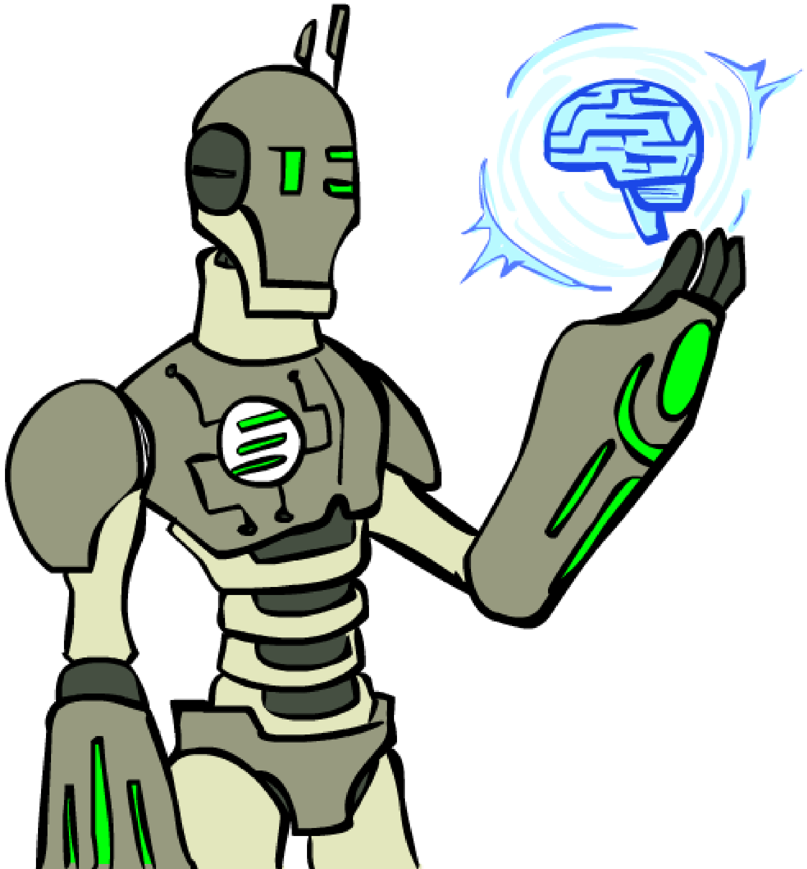
if Op(α) = $\wedge$ then return **and**(PL-TRUE?(Arg1(α),model),
PL-TRUE?(Arg2(α),model))

etc.

(Sometimes called "recursion over syntax")

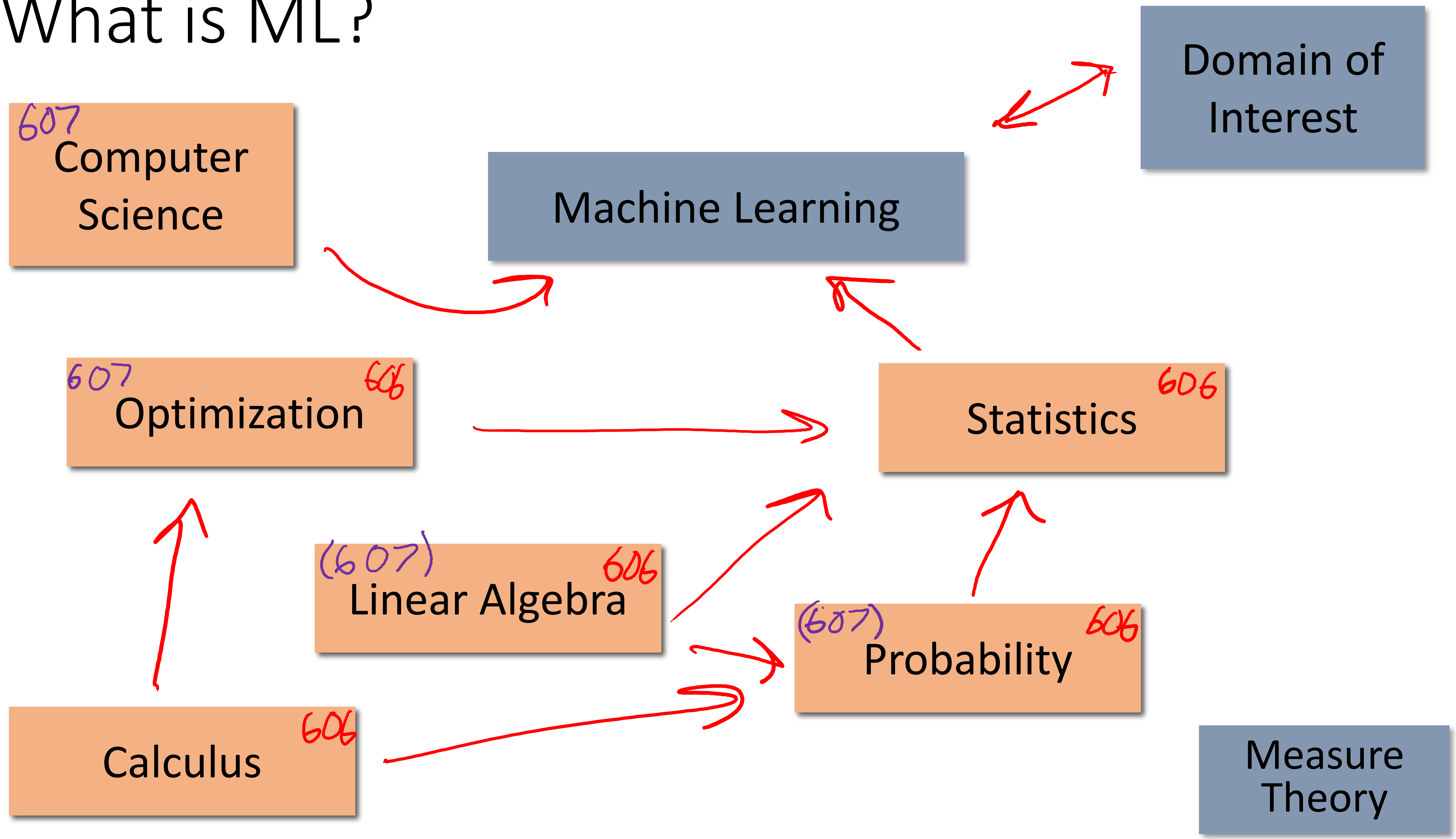
Today

- Course Info
- Warm-up exercise
- Propositional Logic and Proofs
- ML and 606/607 Intro
- More Course Info



Images: ai.berkeley.edu

What is ML?


Domain of Interest
607
Computer Science
Machine Learning
607
Optimization
606
Statistics
606
(607)
Linear Algebra
606
(607)
Probability
606
Calculus
606
Measure Theory

Why Computer Science for ML?

10-607

To best understand __A__ we need __B__

A	B

Slide credit: CMU MLD Matt Gormley

Why Computer Science for ML?

To best understand __A__ we need __B__

A	B
Analysis of Exact Inference in Graphical Models	Computation • Computational Complexity • Recursion; Dynamic Programming • Data Structures for ML Algorithms

Factor Graph Notation

- Variables:

$$\mathcal{X} = \{X_1, \ldots, X_i, \ldots, X_n\}$$

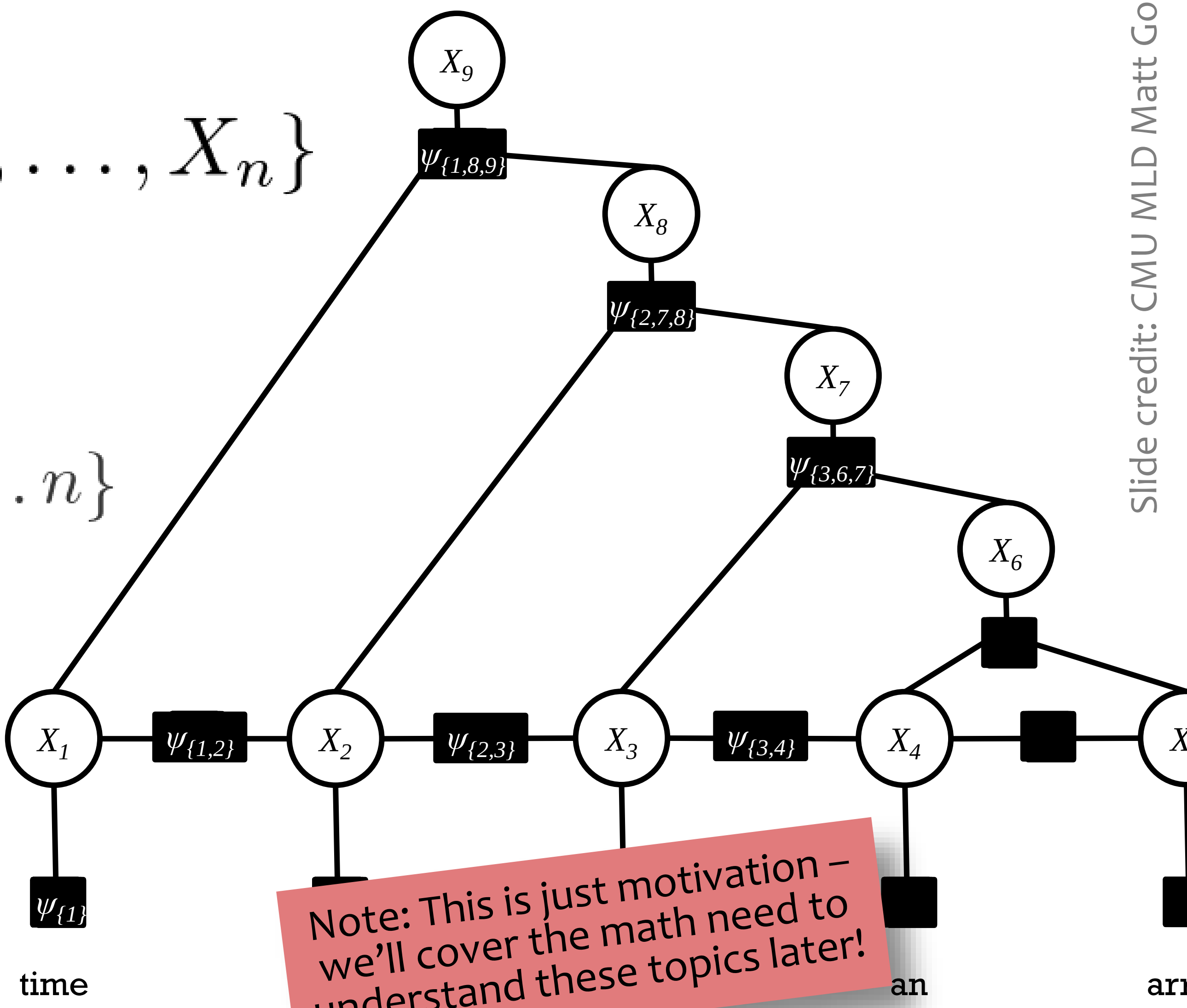
- Factors:

$$\psi_\alpha, \psi_\beta, \psi_\gamma, \ldots$$

$$\text{where } \alpha, \beta, \gamma, \ldots \subseteq \{1, \ldots n\}$$

Joint Distribution

$$p(\boldsymbol{x}) = \frac{1}{Z} \prod_\alpha \psi_\alpha(\boldsymbol{x}_{\boldsymbol{\alpha}})$$



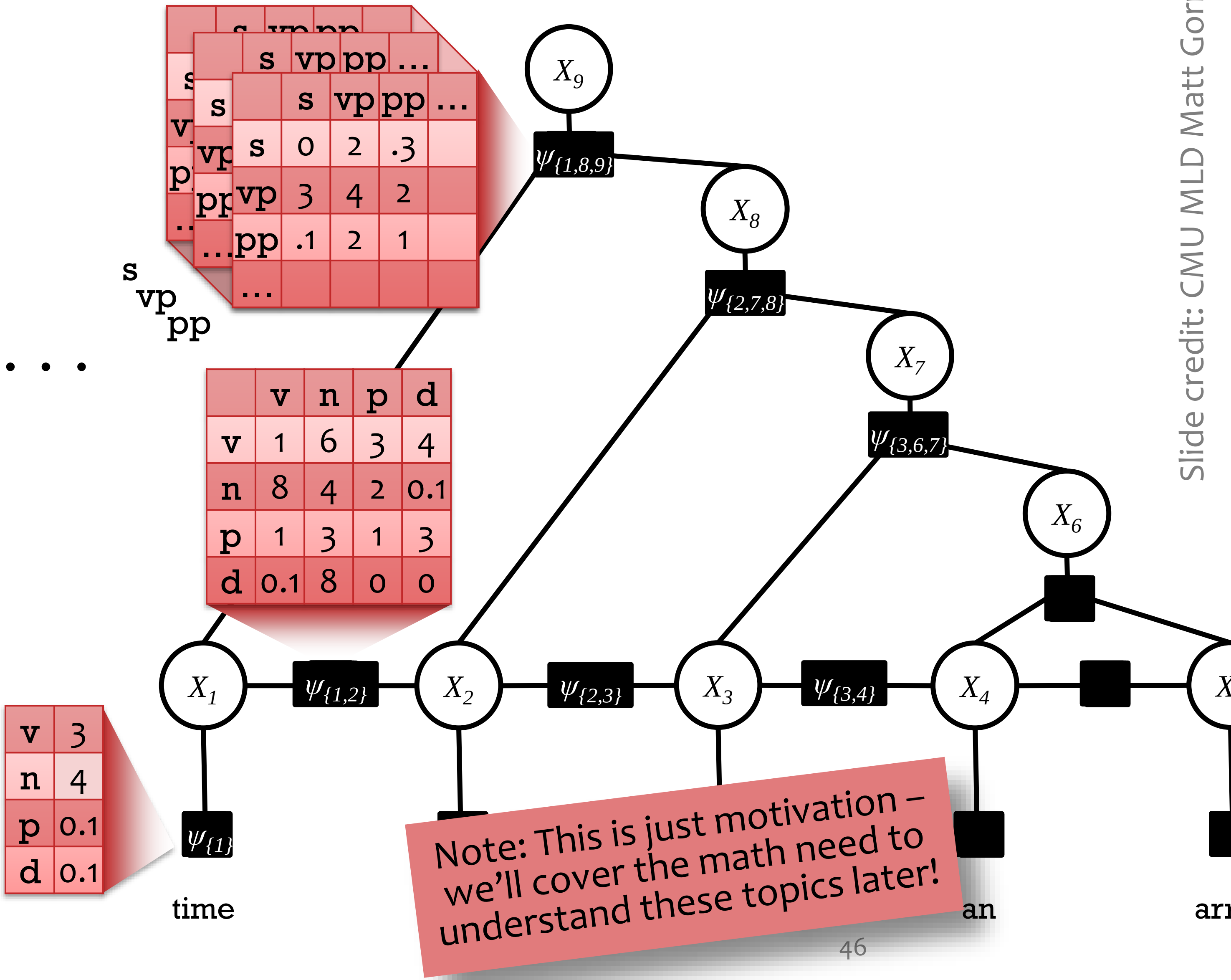
Note: This is just motivation – we'll cover the math need to understand these topics later!

Slide credit: CMU MLD Matt Gormley

Factors are Tensors

- Factors:

$$\psi_\alpha, \psi_\beta, \psi_\gamma, \ldots$$



Note: This is just motivation – we'll cover the math need to understand these topics later!

Slide credit: CMU MLD Matt Gormley

Inference

Given a factor graph, two common tasks …

- Compute the most likely joint assignment,
  $\boldsymbol{x}^* = \text{argmax}_{\boldsymbol{x}}\ p(\boldsymbol{X}=\boldsymbol{x})$
- Compute the marginal distribution of variable X_i:
  $p(X_i=x_i)$ for each value x_i

Both consider *all* joint assignments.

Both are NP-Hard in general.

So, we turn to **approximations**.

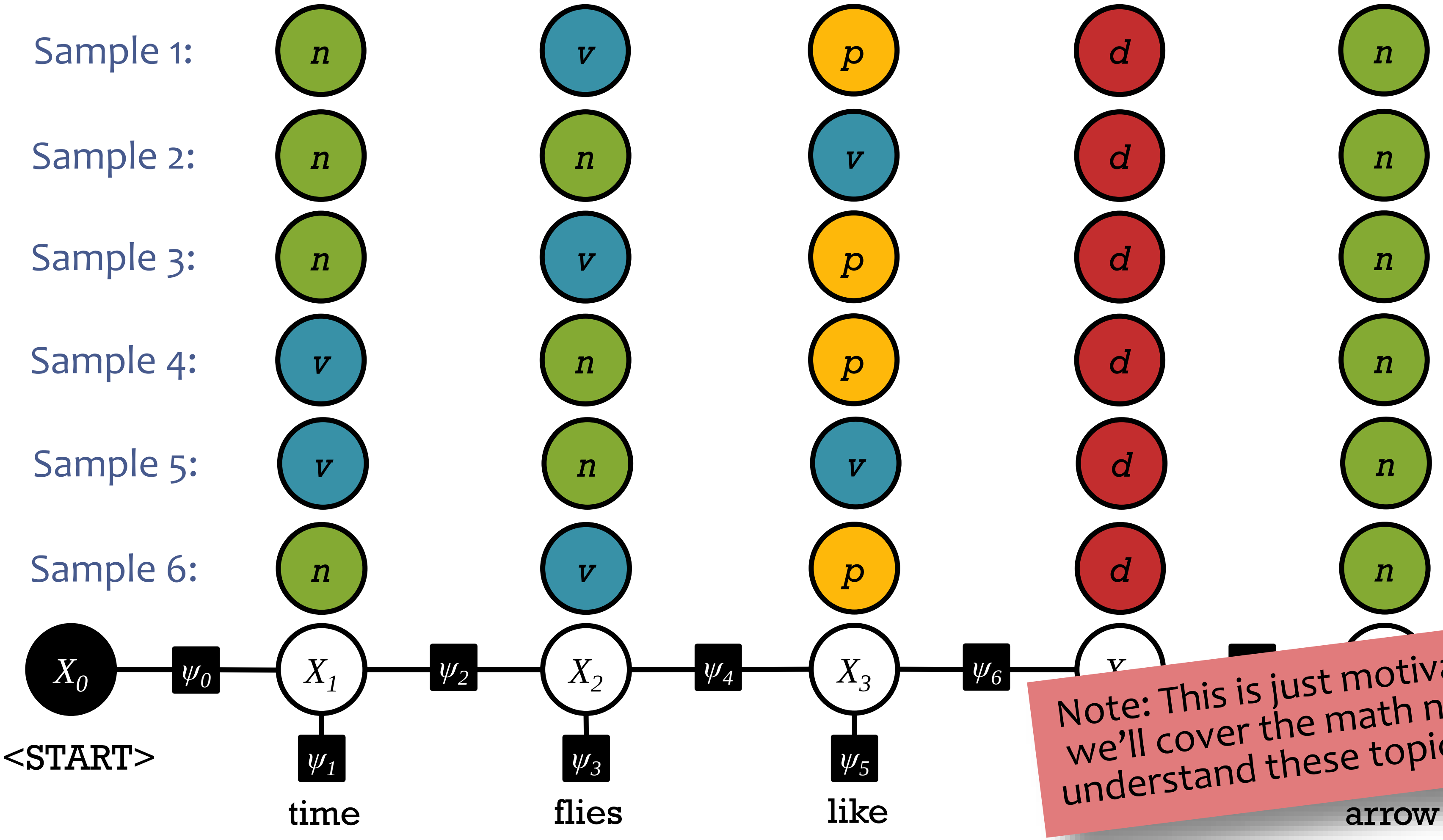
$p(X_i=x_i)$ = sum of $p(\boldsymbol{X}=\boldsymbol{x})$ over joint

Note: This is just motivation – we'll cover the math need to understand these topics later!

Slide credit: CMU MLD Matt Gormley

Marginals by Sampling on Factor Graph

Suppose we took many samples from the distribution over taggings: $p(\boldsymbol{x}) = \frac{1}{Z}\prod_{\alpha} \psi_\alpha(\boldsymbol{x}_{\boldsymbol{\alpha}})$

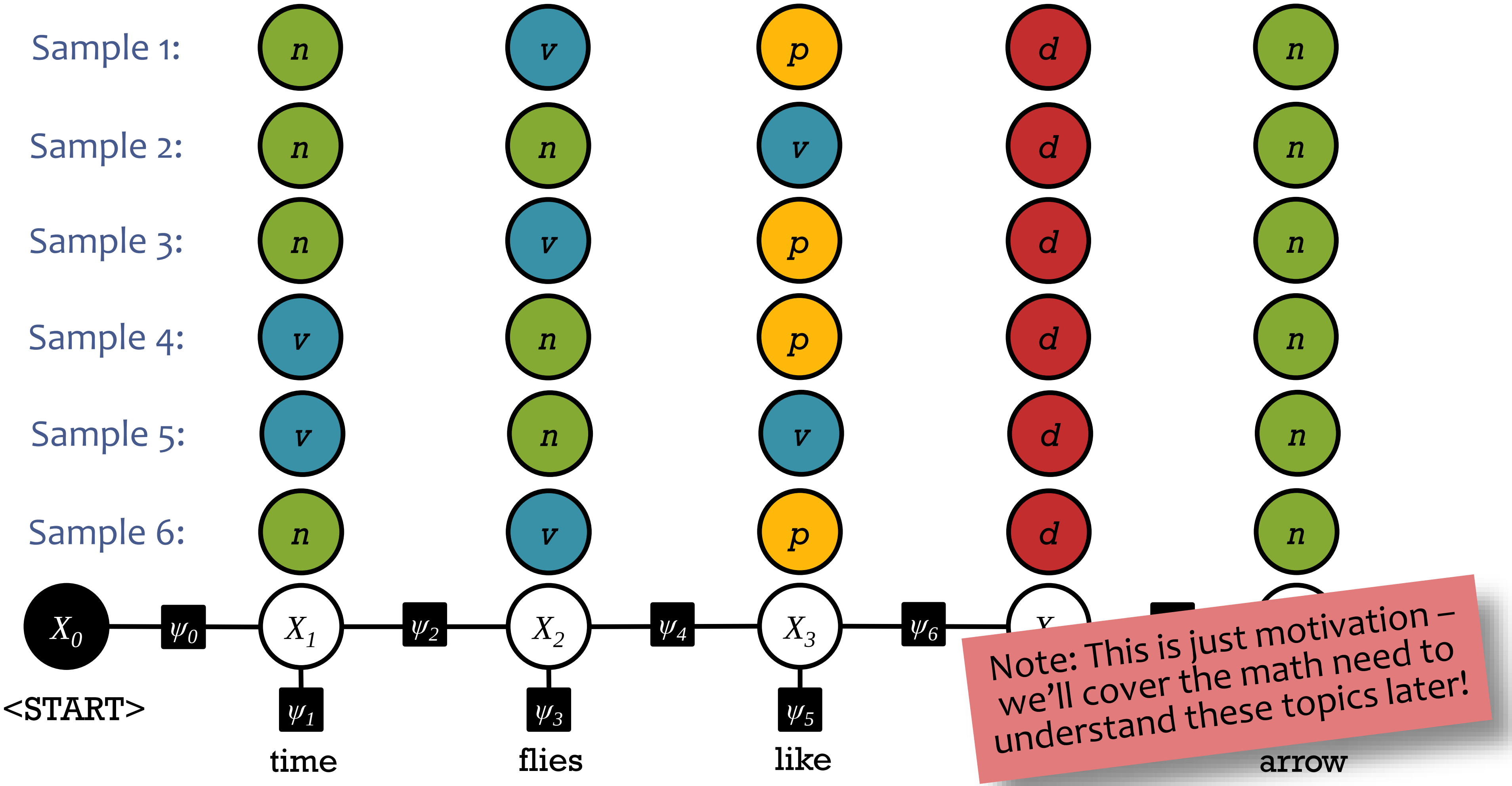


Note: This is just motivation – we'll cover the math need to understand these topics later!

Slide credit: CMU MLD Matt Gormley

Marginals by Sampling on Factor Graph

The marginal $p(X_i = x_i)$ gives the probability that variable X_i takes value x_i in a random sample

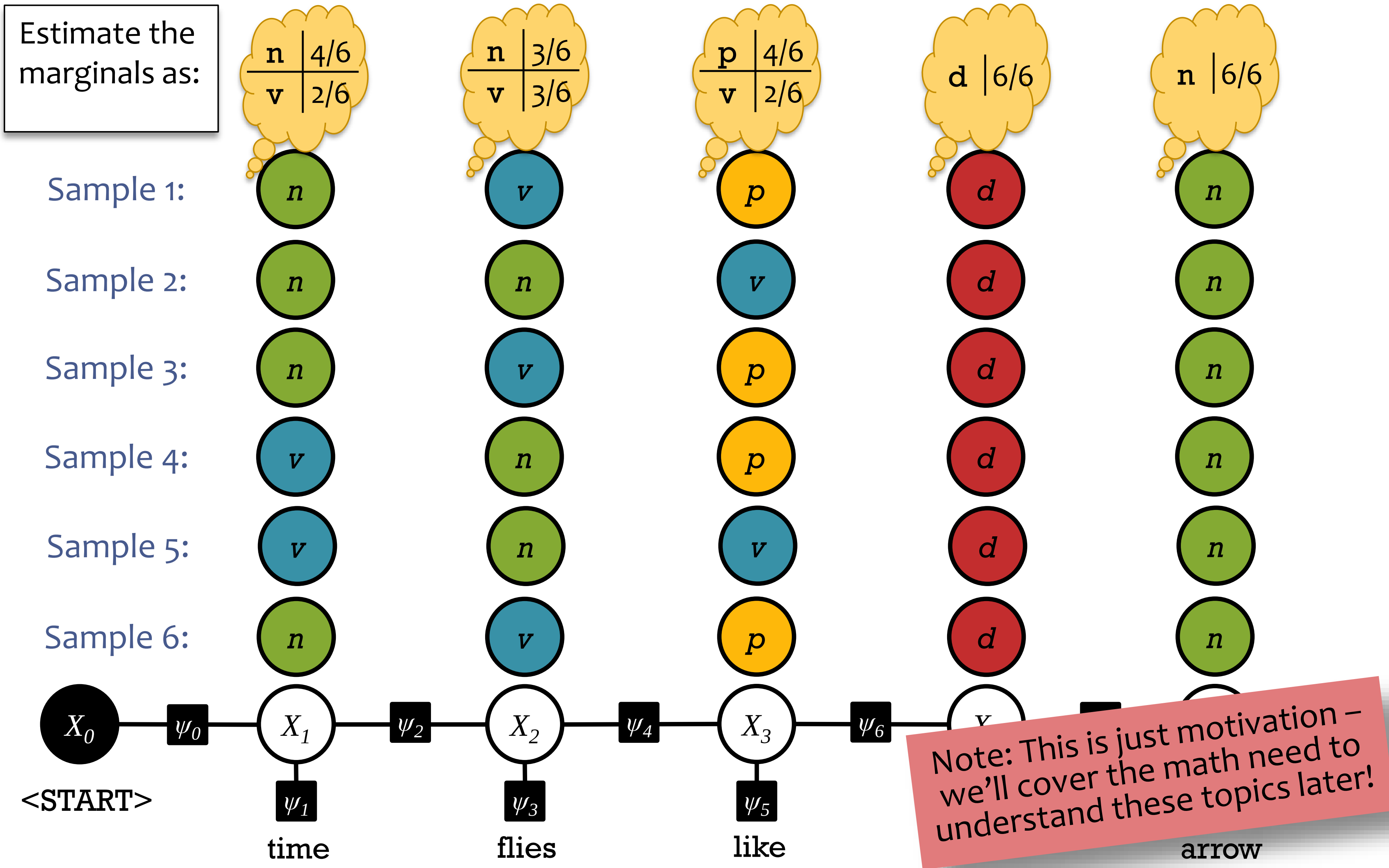


Note: This is just motivation – we'll cover the math need to understand these topics later!

Slide credit: CMU MLD Matt Gormley

Marginals by Sampling on Factor Graph

Estimate the marginals as:



Note: This is just motivation – we'll cover the math need to understand these topics later!

Slide credit: CMU MLD Matt Gormley

Why Computer Science for ML?

10-607

To best understand __A__ we need __B__

A	B
Analysis of Exact Inference in Graphical Models	Computation • Computational Complexity • Recursion; Dynamic Programming • Data Structures for ML Algorithms
Implementation Design of a Deep Learning Library	Programming & Efficiency • Debugging for Machine Learning • Efficient Implementation / Profiling ML Algorithms

Slide credit: CMU MLD Matt Gormley

Finite Difference Method

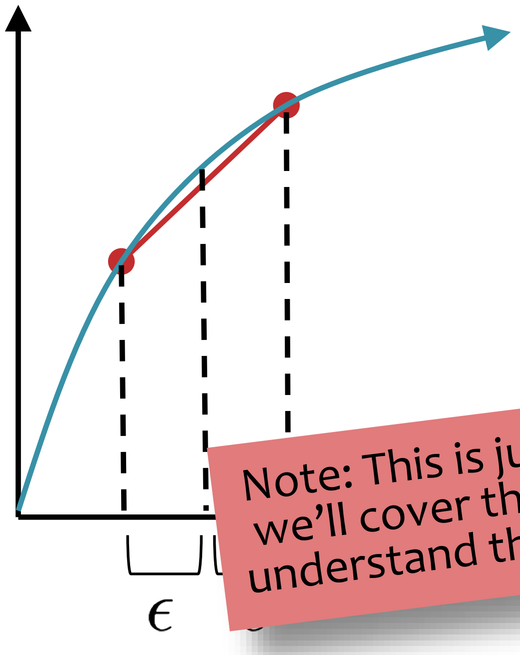
The *centered* finite difference approximation is:

$$\frac{\partial}{\partial \theta_i} J(\boldsymbol{\theta}) \approx \frac{(J(\boldsymbol{\theta} + \epsilon \cdot \boldsymbol{d}_i) - J(\boldsymbol{\theta} - \epsilon \cdot \boldsymbol{d}_i))}{2\epsilon} \tag{1}$$

where $\boldsymbol{d}_i$ is a 1-hot vector consisting of all zeros except for the ith entry of $\boldsymbol{d}_i$, which has value 1.

Notes:

- Suffers from issues of floating point precision, in practice
- Typically only appropriate to use on small examples with an appropriately chosen epsilon



Note: This is just motivation – we'll cover the math need to understand these topics later!

Slide credit: CMU MLD Matt Gormley

Differentiation

Chain Rule Quiz #1:

Suppose x = 2 and z = 3, what are dy/dx and dy/dz for the function below?

$$y = \exp(xz) + \frac{xz}{\log(x)} + \frac{\sin(\log(x))}{\exp(xz)}$$

Finite Difference Solution:

```
from math import *

# Define function
def f(x, z):
    return exp(x*z) + x*z/log(x) + sin(log(x)) / exp(x*z)

# Inputs
x = 2; z = 3; e = 1e-8

# Finite difference check
dydx = (f(x+e, z) - f(x-e, z)) / (2*e
dydz = (f(x, z+e) - f(x, z-e)) / (2*e
print "dydx =", dydx
print "dydz =", dydz
```

Note: This is just motivation – we'll cover the math need to understand these topics later!

Slide credit: CMU MLD Matt Gormley

Backpropagation

Automatic Differentiation – Reverse Mode (aka. Backpropagation)

Forward Computation

1. Write an **algorithm** for evaluating the function $y = f(\mathbf{x})$. The algorithm defines a **directed acyclic graph,** where each variable is a node (i.e. the "**computation graph**")
2. Visit each node in **topological order**.
   For variable u_i with inputs $v_1,\ldots, v_N$
   a. Compute $u_i = g_i(v_1,\ldots, v_N)$
   b. Store the result at the node

Backward Computation

1. **Initialize** all partial derivatives dy/du_j to 0 and $dy/dy = 1$.
2. Visit each node in **reverse topological order**.
   For variable $u_i = g_i(v_1,\ldots, v_N)$
   a. We already know dy/du_i
   b. Increment dy/dv_j by $(dy/du_i)(du_i/dv_j)$
   (Choice of algorithm ensures computing $(d$

Return partial derivatives dy/du_i for all variables

Note: This is just motivation – we'll cover the math need to understand these topics later!

Slide credit: CMU MLD Matt Gormley

Why Computer Science for ML?

To best understand __A__ we need __B__

A	B
Analysis of Exact Inference in Graphical Models	Computation • Computational Complexity • Recursion; Dynamic Programming • Data Structures for ML Algorithms
Implementation Design of a Deep Learning Library	Programming & Efficiency • Debugging for Machine Learning • Efficient Implementation / Profiling ML Algorithms
Optimization for Support Vector Machines (SVMs)	Optimization • Unconstrained Optimization • Preconditioning • Constrained Optimization

Slide credit: CMU MLD Matt Gormley

Support Vector Machines (SVMs)

Hard-margin SVM (Primal)

$$\min_{\mathbf{w},b} \frac{1}{2}\|\mathbf{w}\|_2^2$$

$$\text{s.t. } y^{(i)}(\mathbf{w}^T\mathbf{x}^{(i)} + b) \geq 1, \quad \forall i = 1, \ldots, N$$

Hard-margin SVM (Lagrangian Dual)

$$\max_{\boldsymbol{\alpha}} \sum_{i=1}^{N} \alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i \alpha_j y^{(i)} y^{(j)} \mathbf{x}^{(i)} \cdot \mathbf{x}^{(j)}$$

$$\text{s.t. } \alpha_i \geq 0, \quad \forall i = 1, \ldots, N$$

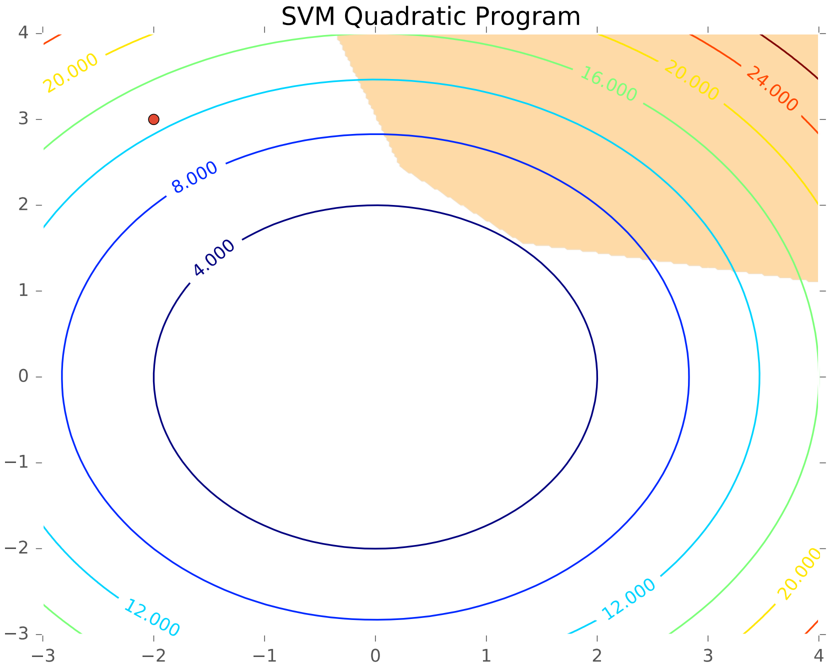
$$\sum_{i=1}^{N} \alpha_i y^{(i)} = 0$$

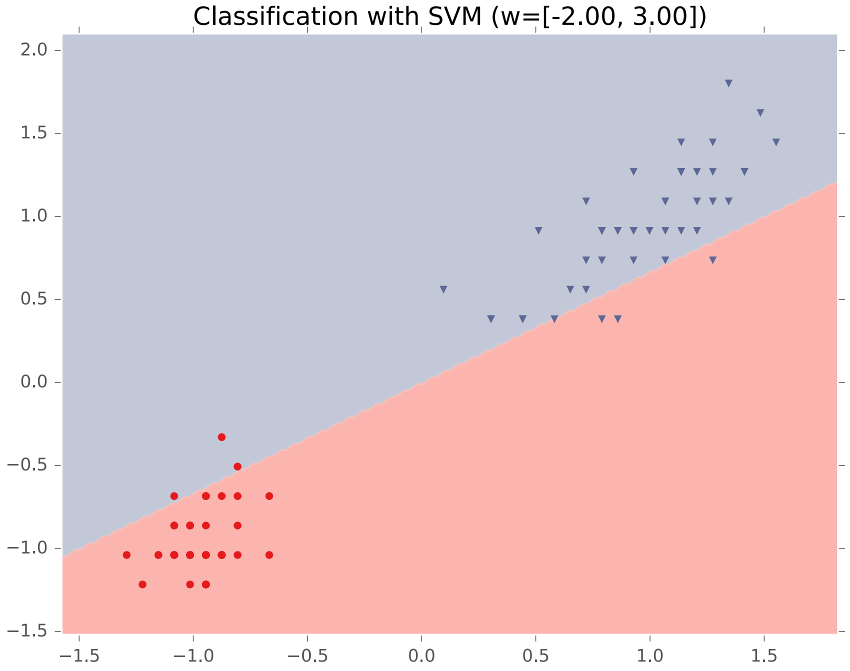
- Instead of minimizing the primal, we can maximize the dual problem
- For the SVM, these two problems give the same answer (i.e. the minimum of one is the maximum of the other)
- *Definition*: **support vectors** are those points for which $\alpha^{(i)} \neq 0$

Note: This is just motivation – we'll cover the math need to understand these topics later!

Slide credit: CMU MLD Matt Gormley

SVM QP

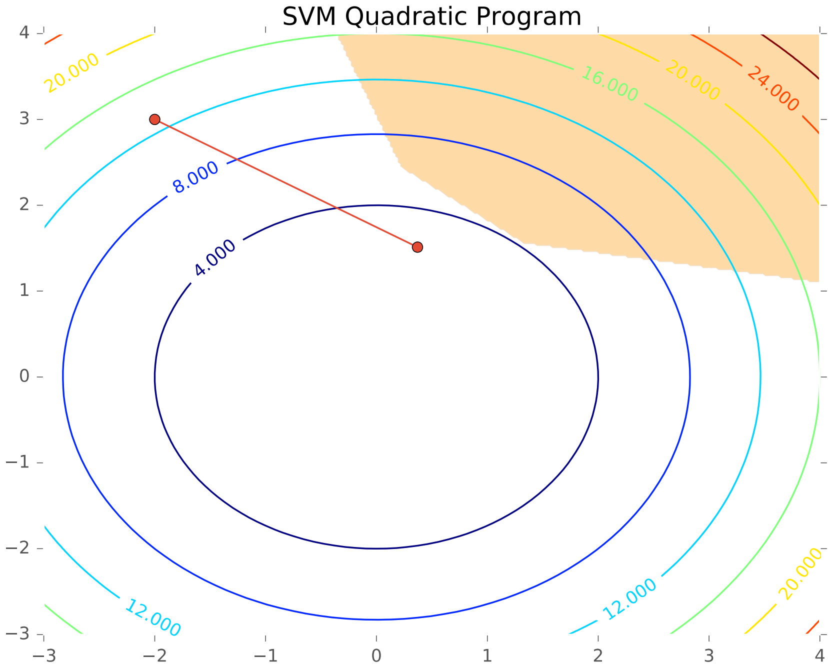


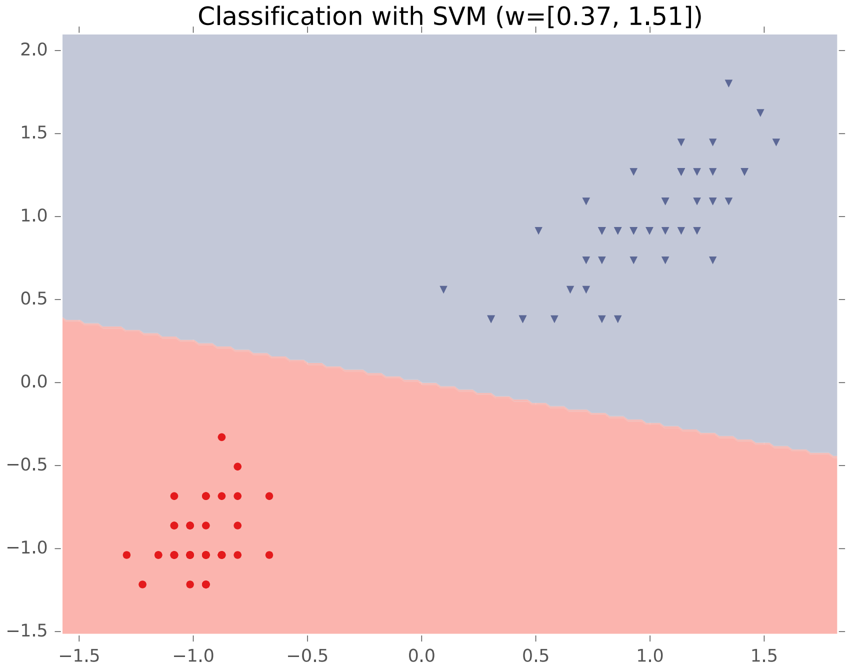


Note: This is just motivation – we'll cover the math need to understand these topics later!

Slide credit: CMU MLD Matt Gormley

SVM QP

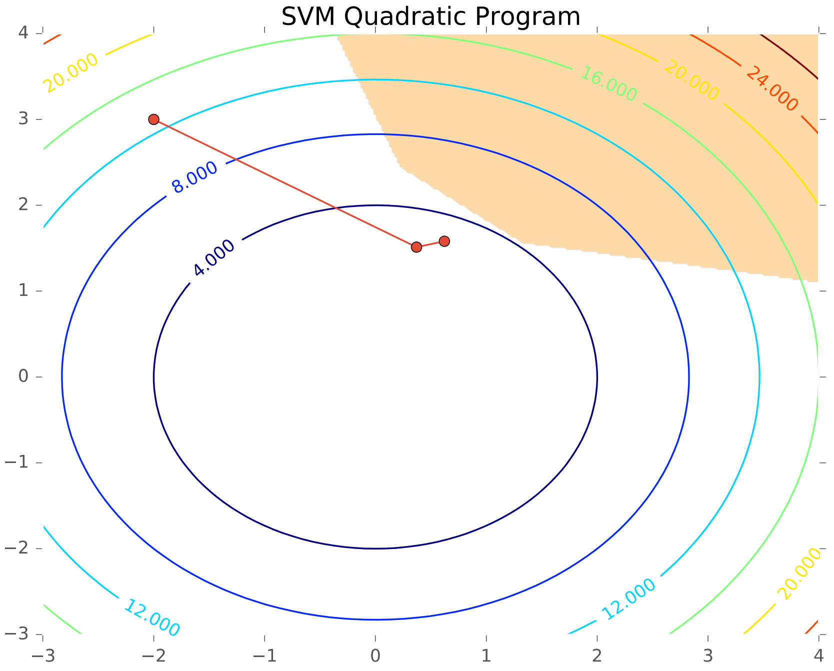


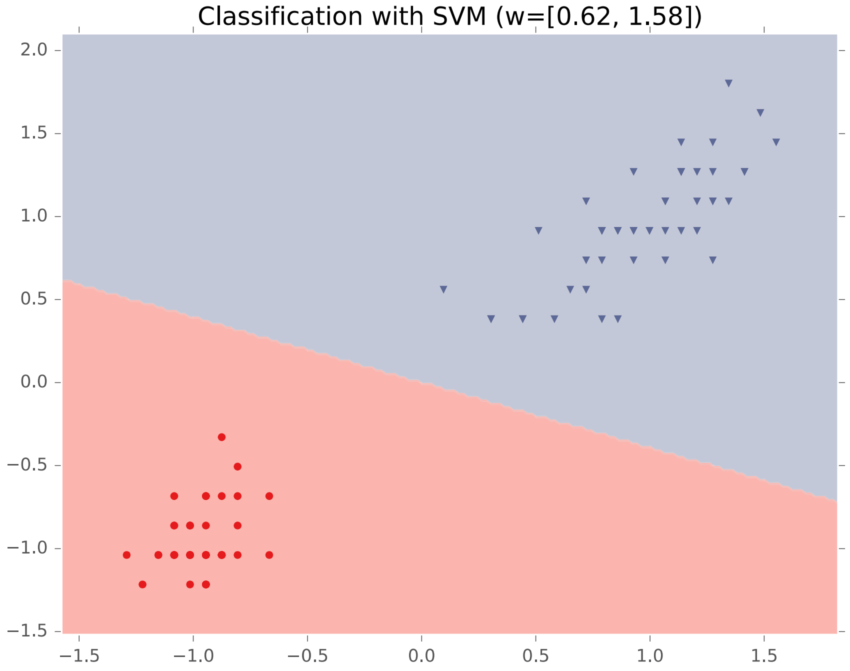


Note: This is just motivation – we'll cover the math need to understand these topics later!

Slide credit: CMU MLD Matt Gormley

SVM QP

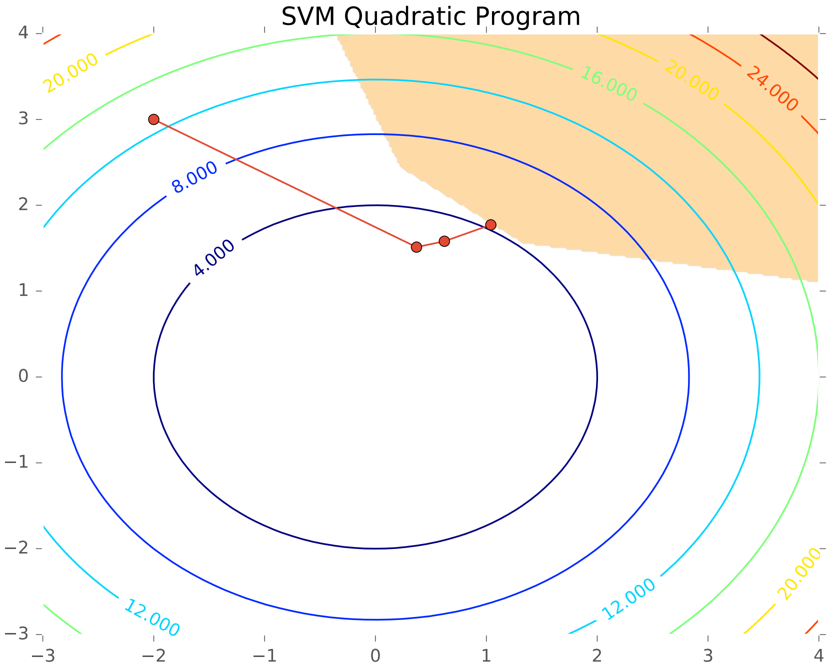


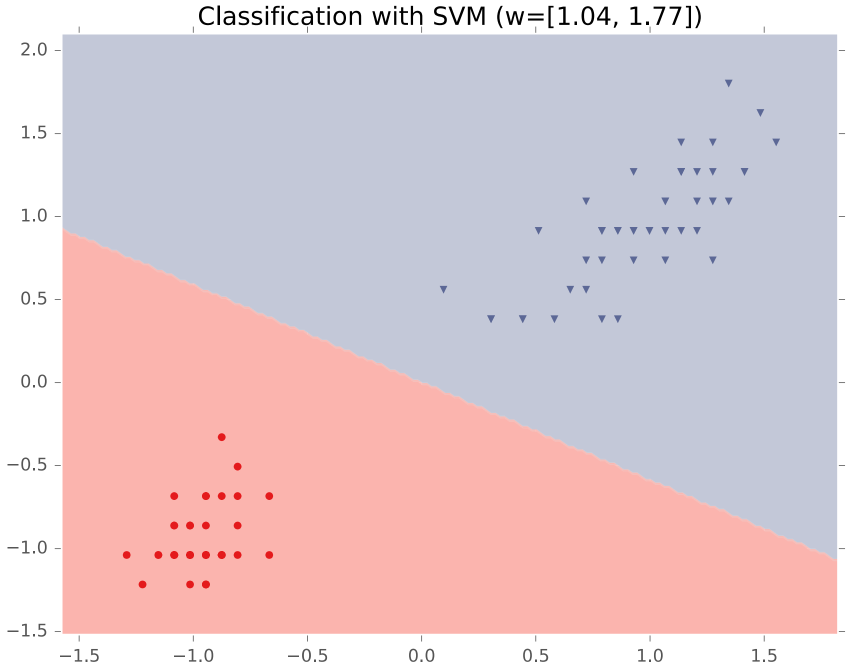


Note: This is just motivation – we'll cover the math need to understand these topics later!

Slide credit: CMU MLD Matt Gormley

SVM QP

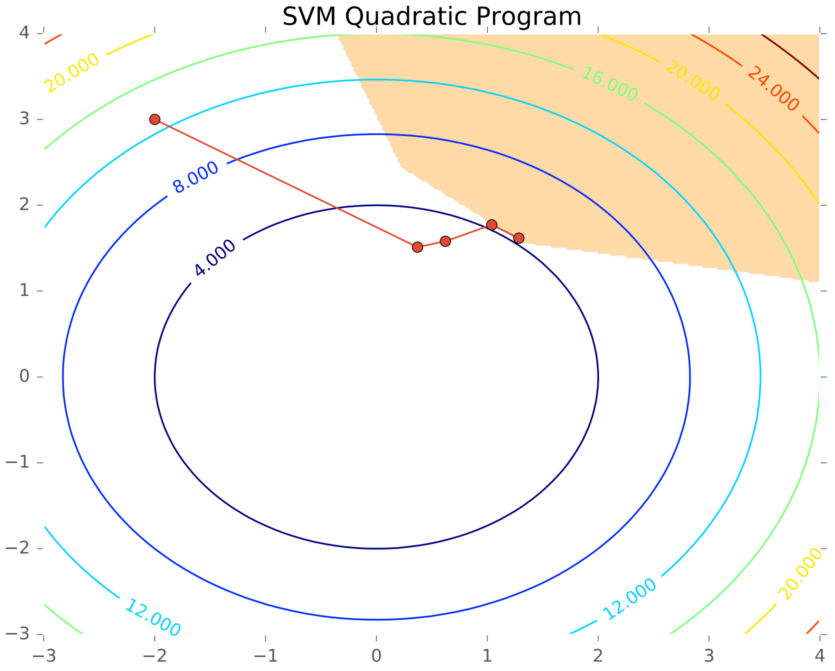


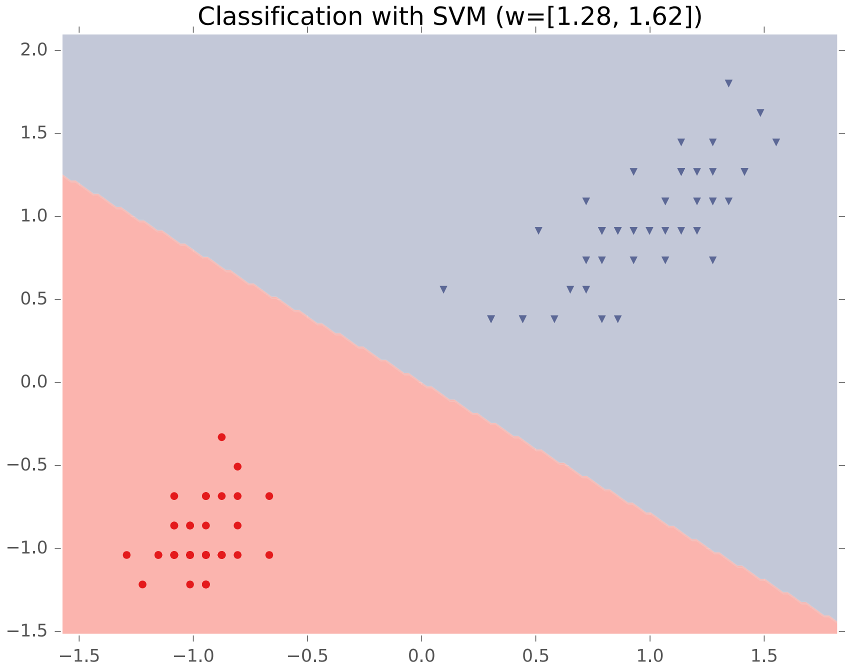


Note: This is just motivation – we'll cover the math need to understand these topics later!

Slide credit: CMU MLD Matt Gormley

SVM QP

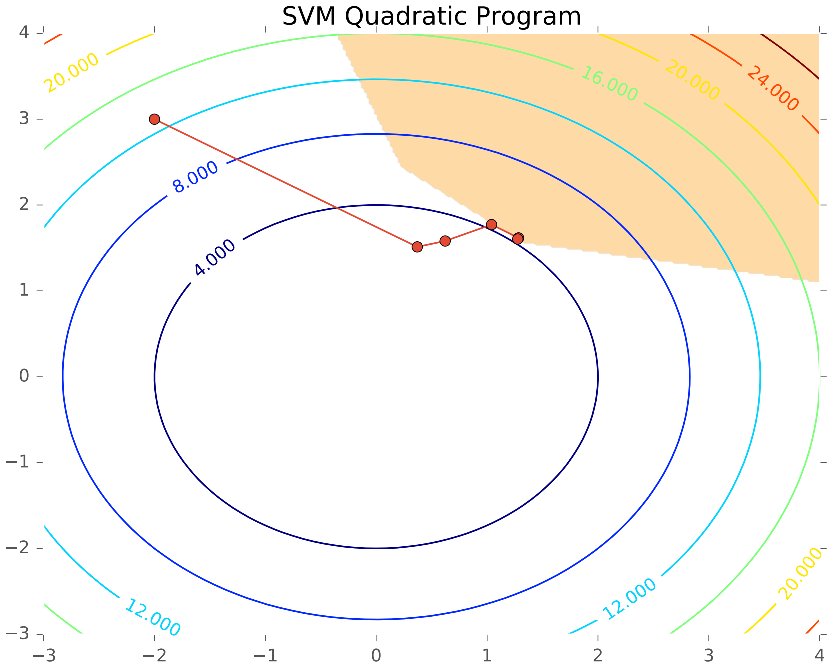


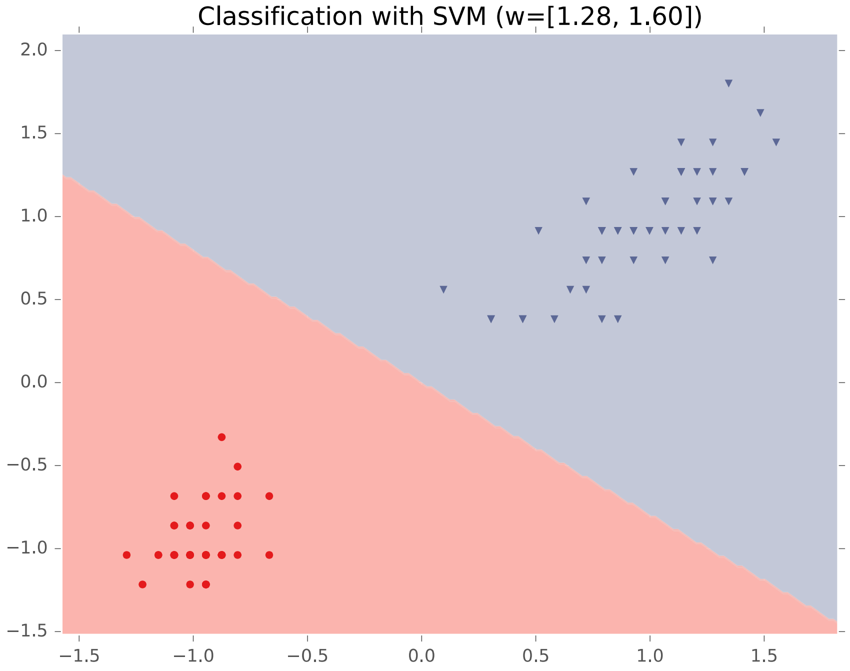


Note: This is just motivation – we'll cover the math need to understand these topics later!

Slide credit: CMU MLD Matt Gormley

SVM QP





Note: This is just motivation – we'll cover the math need to understand these topics later!

Slide credit: CMU MLD Matt Gormley

Why Computer Science for ML?

10-607

To best understand __A__ we need __B__

A	B
Analysis of Exact Inference in Graphical Models	Computation • Computational Complexity • Recursion; Dynamic Programming • Data Structures for ML Algorithms
Implementation Design of a Deep Learning Library	Programming & Efficiency • Debugging for Machine Learning • Efficient Implementation / Profiling ML Algorithms
Optimization for Support Vector Machines (SVMs)	Optimization • Unconstrained Optimization • Preconditioning • Constrained Optimization

The core content for this course is the **computer science** (Column B), but you will apply what you learn to **real problems in machine learning** (Column A)

Slide credit: CMU MLD Matt Gormley

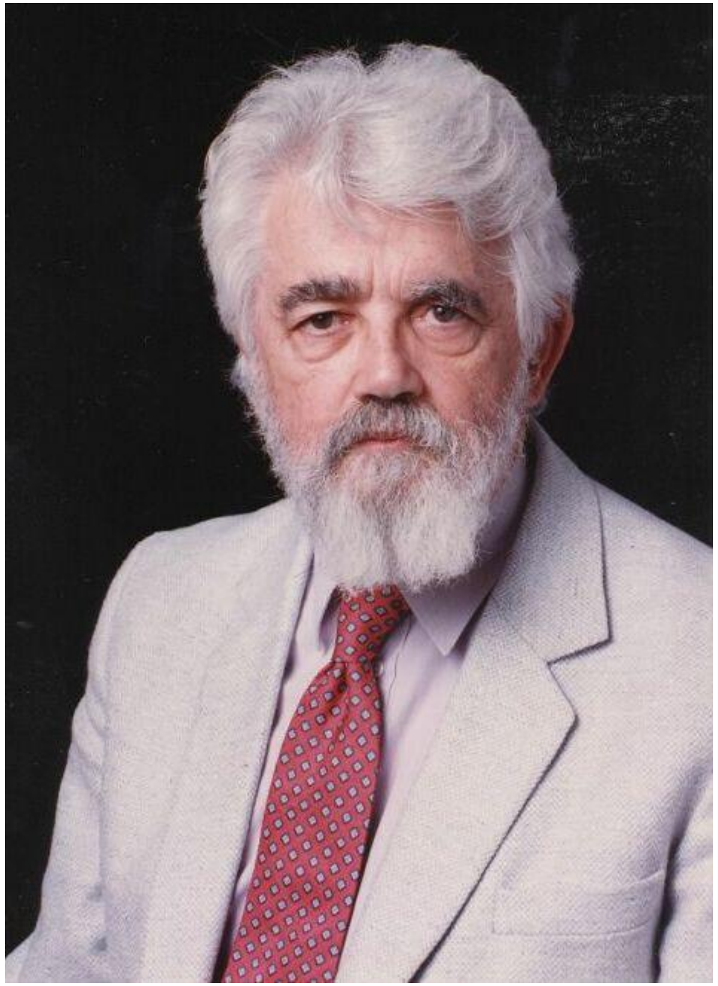
AI Definition by John McCarthy

What is artificial intelligence

- It is the science and engineering of making intelligent machines, especially intelligent computer programs

What is intelligence

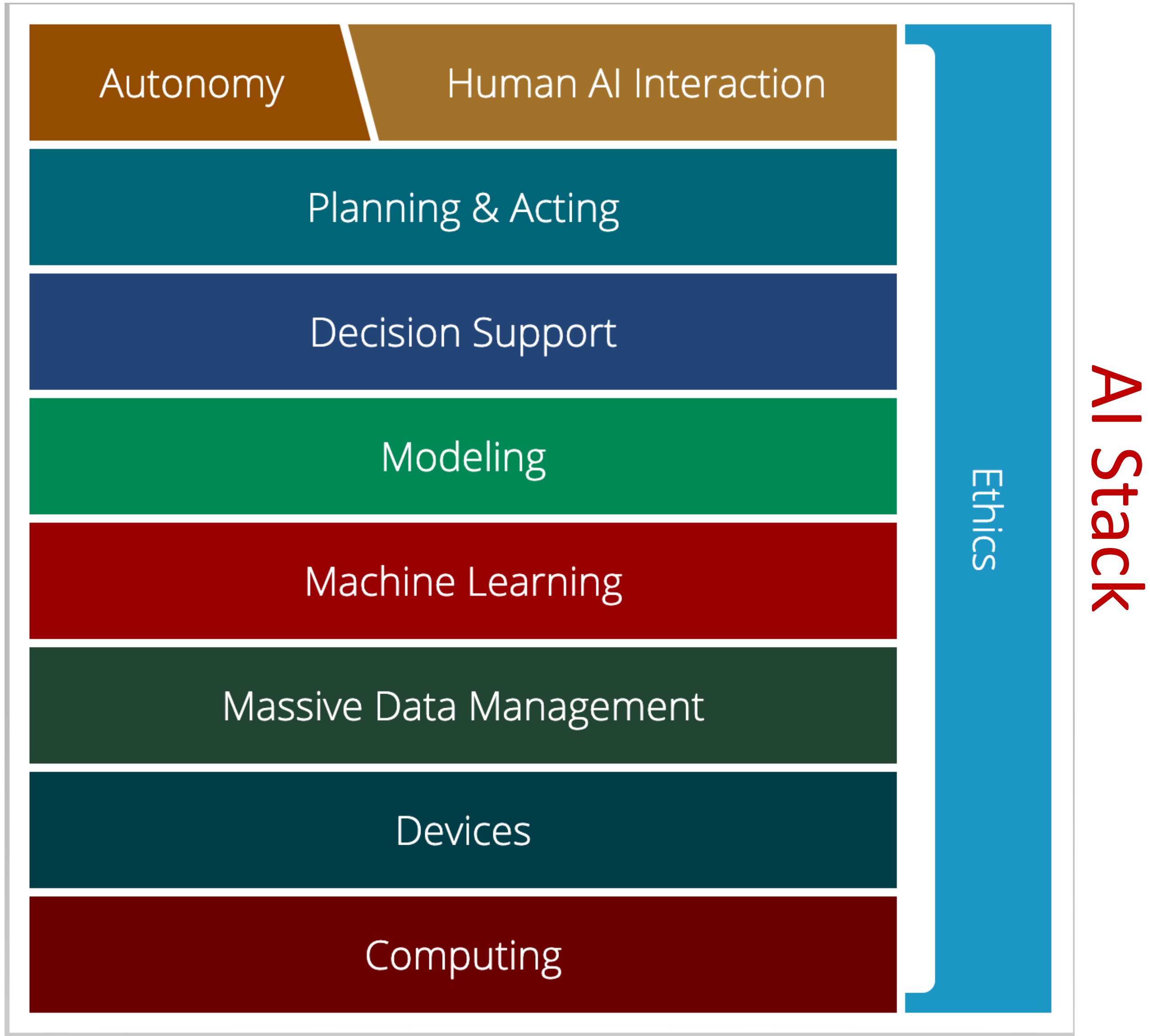
- Intelligence is the computational part of the ability to achieve goals in the world



http://www-formal.stanford.edu/jmc/whatisai/whatisai.html

AI Stack for CMU AI

"AI must understand the human needs and it must make smart design decisions based on that understanding"

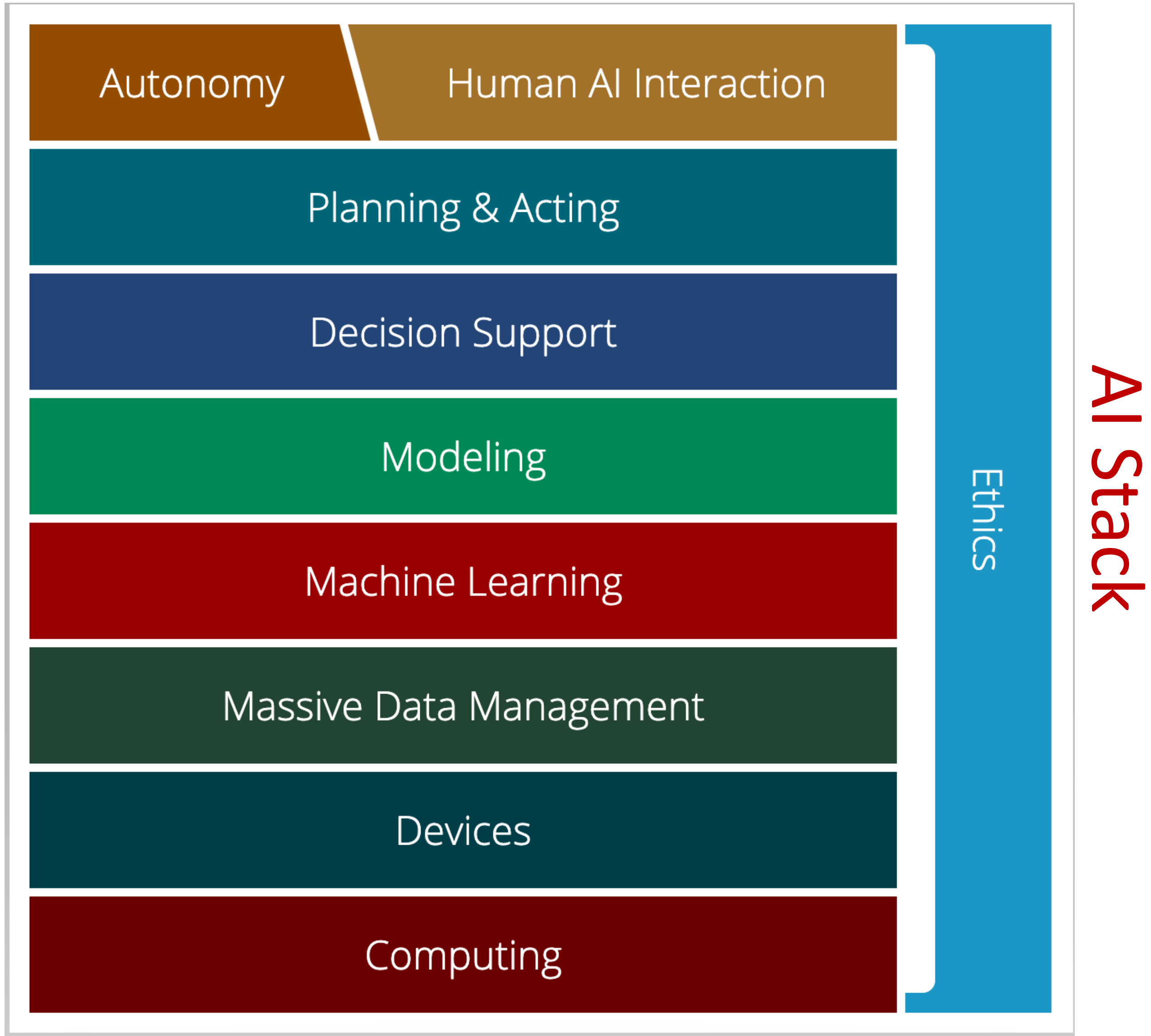


https://ai.cs.cmu.edu/about

AI Stack for CMU AI

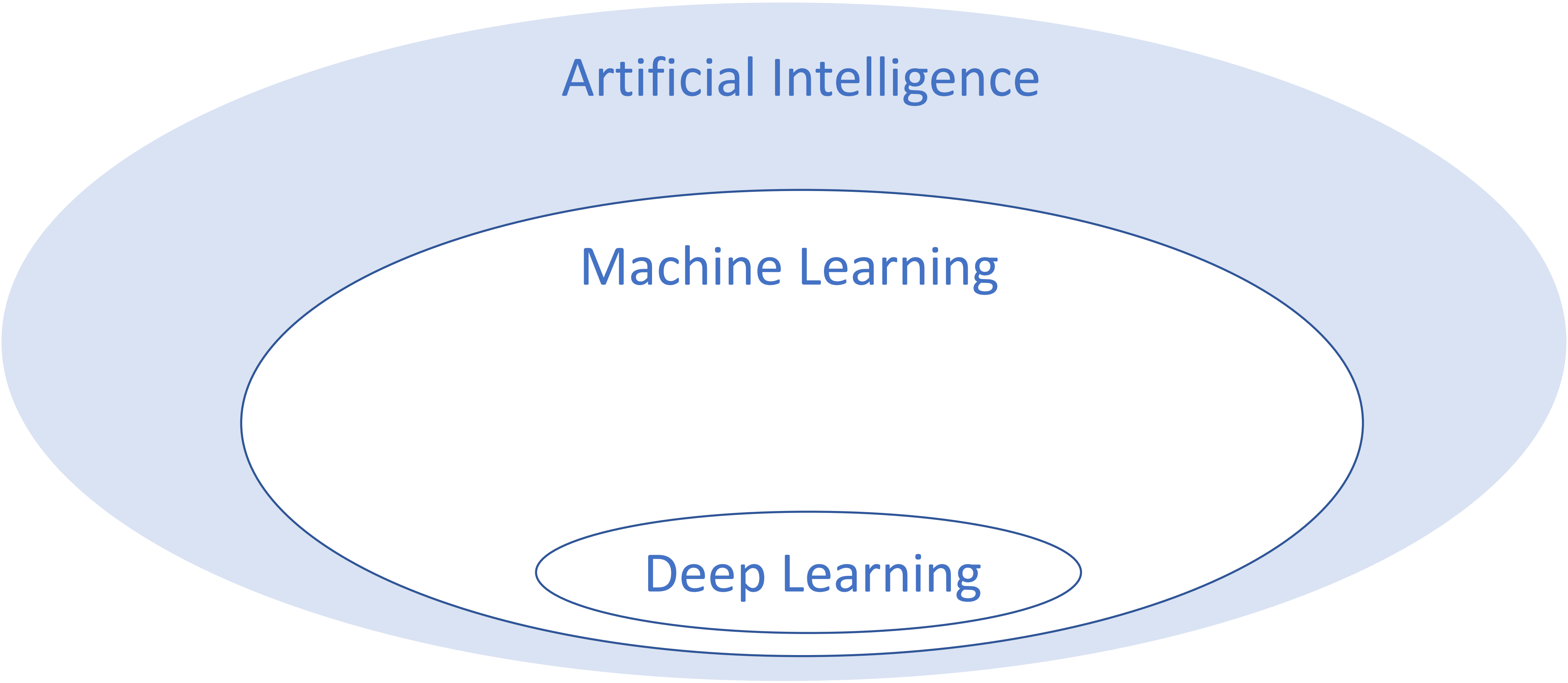
“Machine learning focuses on creating programs that learn from experience.”

“It advances computing through exposure to new scenarios, testing and adaptation, while using pattern- and trend-detection to help the computer make better decisions in similar, subsequent situations.”

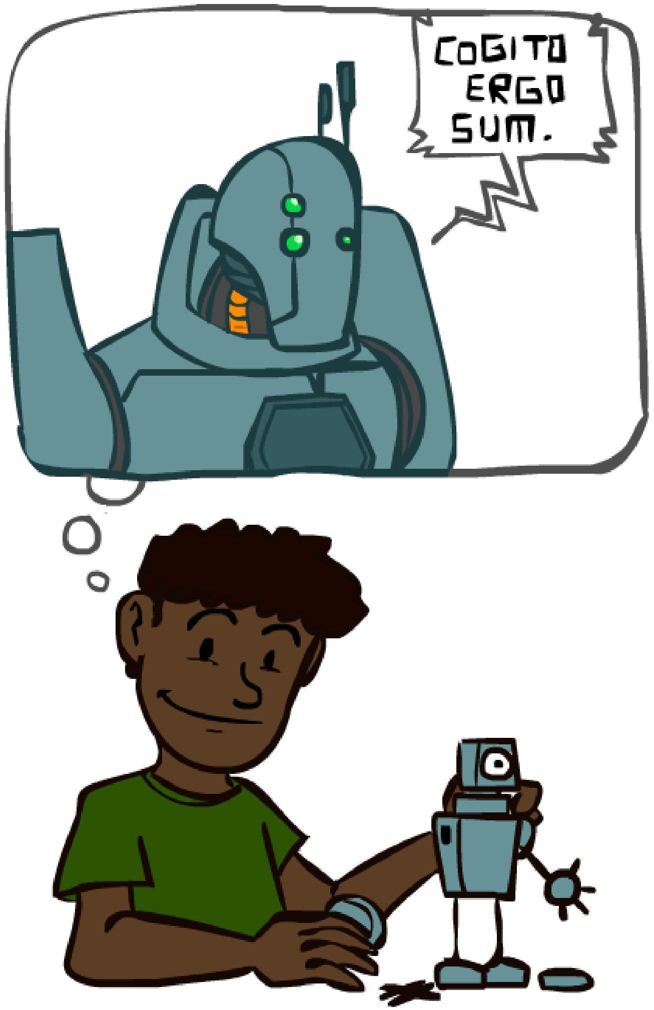


https://ai.cs.cmu.edu/about

Artificial Intelligence vs Machine Learning?

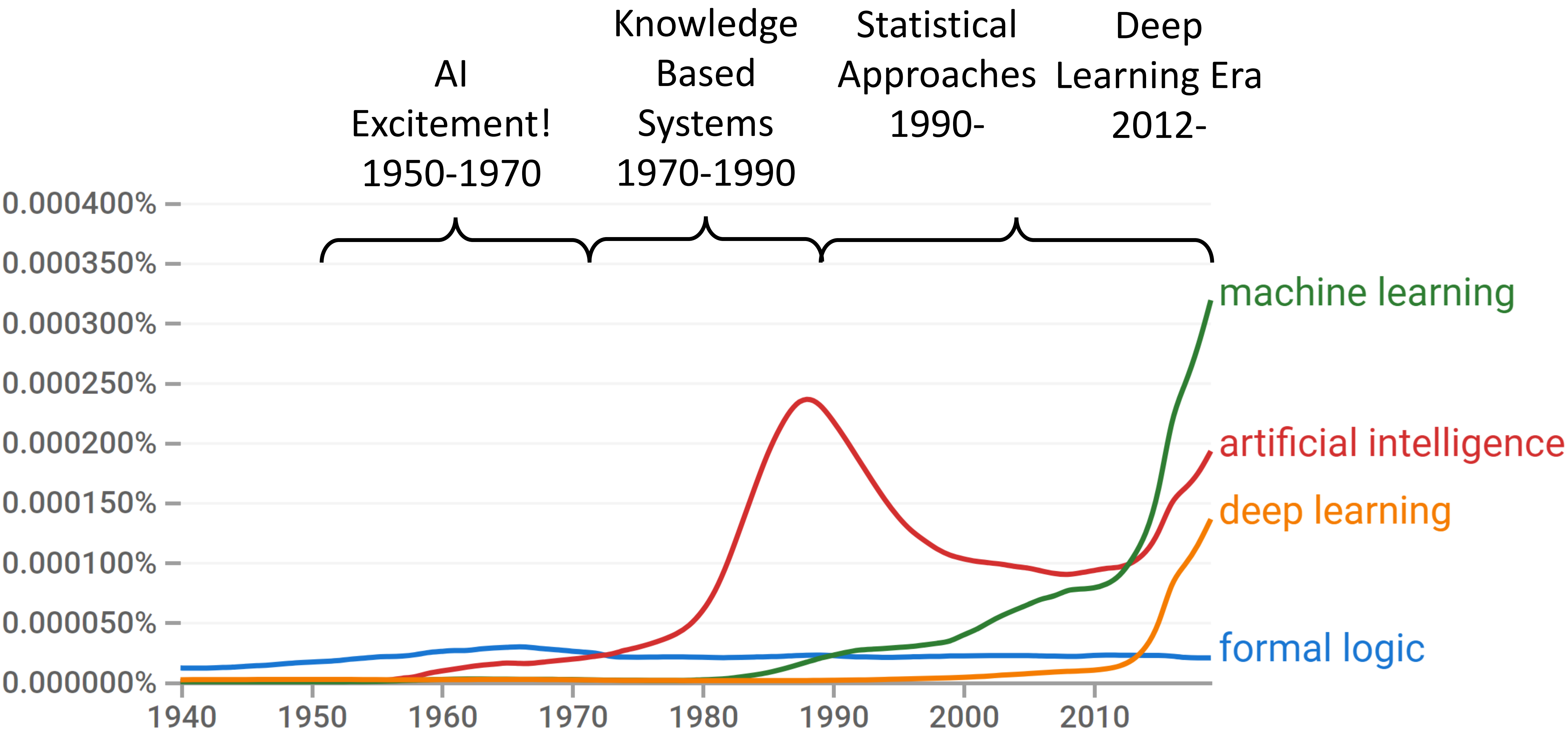


A Brief History of AI



Images: ai.berkeley.edu

A Brief History of AI



https://books.google.com/ngrams

A Brief History of AI

1940-1950: Early days

- 1943: McCulloch & Pitts: Boolean circuit model of brain
- 1950: Turing's “Computing Machinery and Intelligence”

1950—70: Excitement: Look, Ma, no hands!

- 1950s: Early AI programs, including Samuel's checkers program, Newell & Simon's Logic Theorist, Gelernter's Geometry Engine
- 1956: Dartmouth meeting: “Artificial Intelligence” adopted

1970—90: Knowledge-based approaches

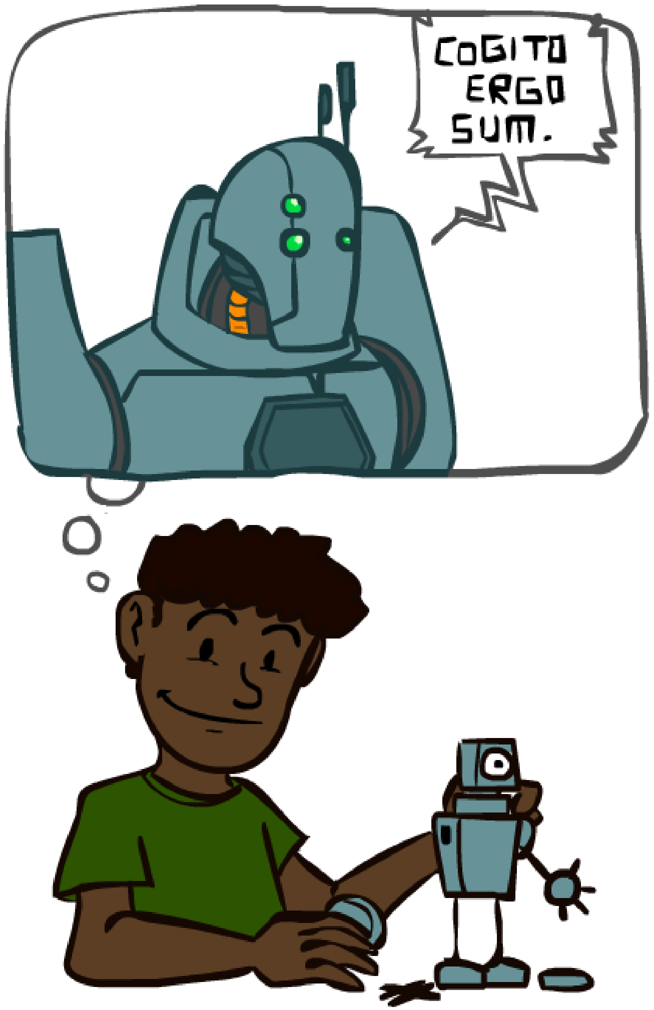
- 1969—79: Early development of knowledge-based systems
- 1980—88: Expert systems industry booms
- 1988—93: Expert systems industry busts: “AI Winter”

1990—: Statistical approaches

- Resurgence of probability, focus on uncertainty
- General increase in technical depth
- Agents and learning systems… “AI Spring”?

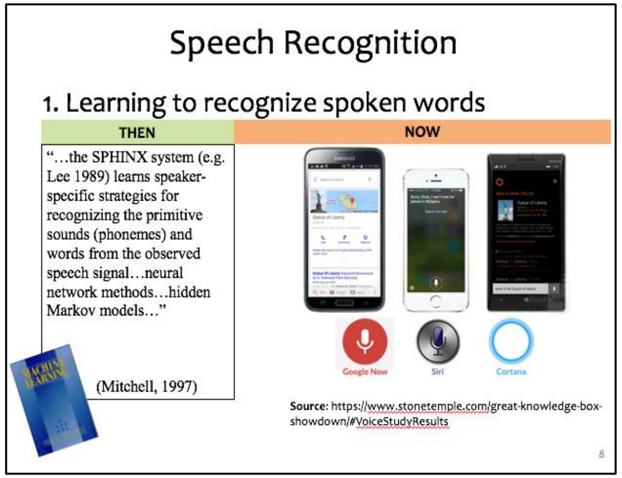
2012—: Deep learning

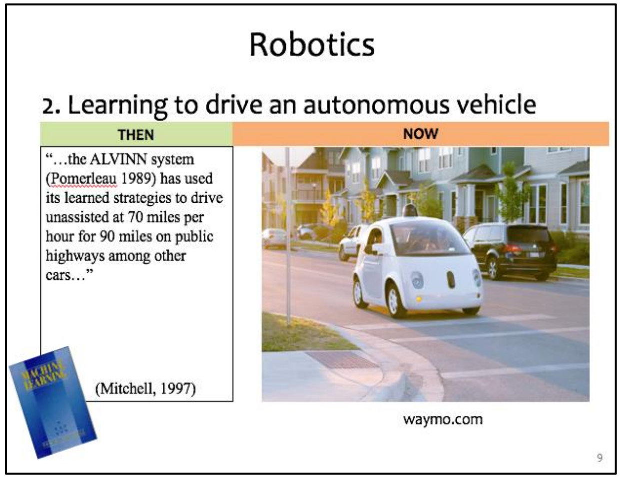
- 2012: ImageNet & AlexNet

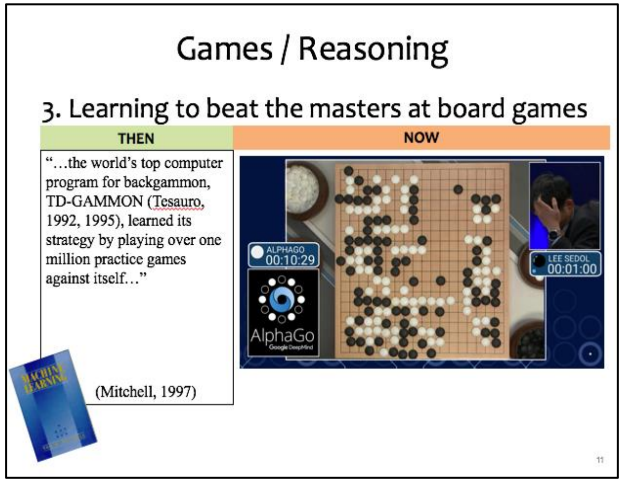


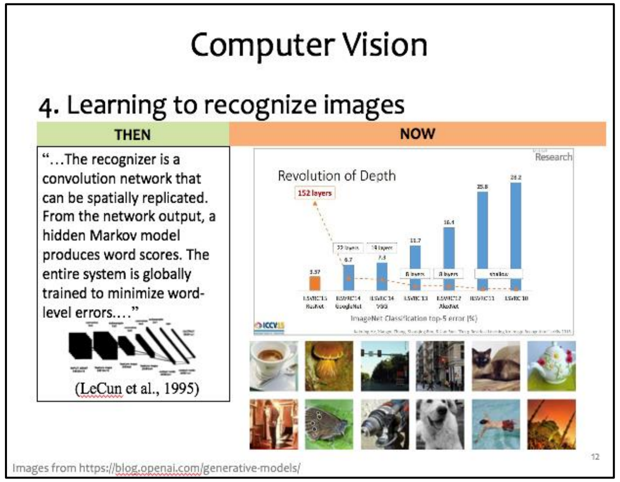
Images: ai.berkeley.edu

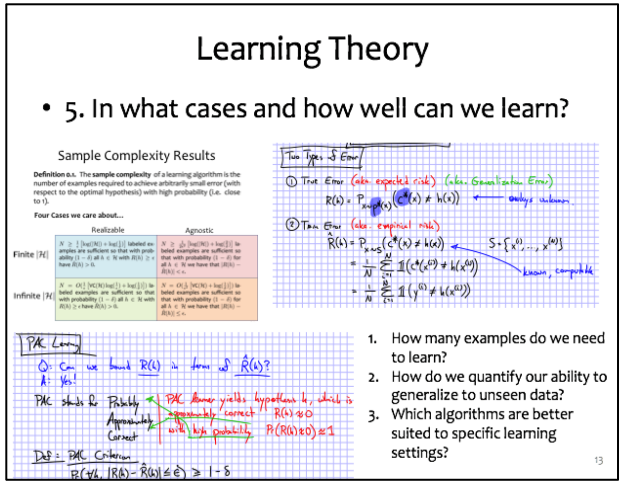
ML Applications?





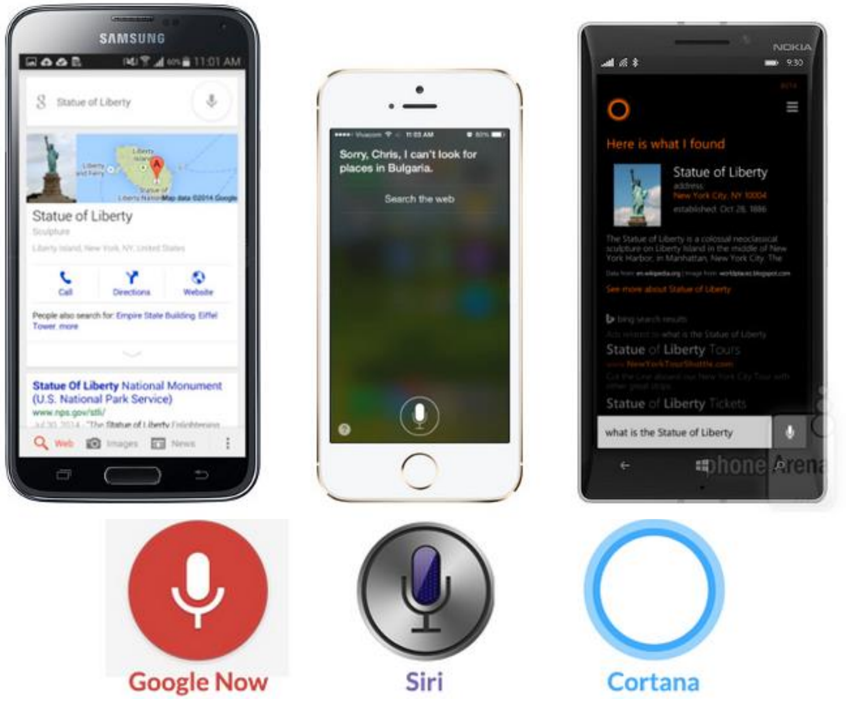


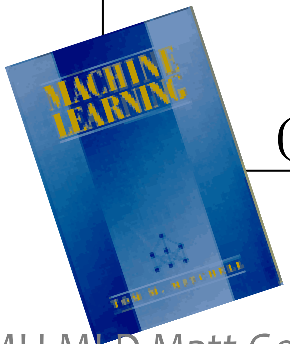




Speech Recognition

1. Learning to recognize spoken words

THEN	NOW
"…the SPHINX system (e.g. Lee 1989) learns speaker-specific strategies for recognizing the primitive sounds (phonemes) and words from the observed speech signal…neural network methods…hidden Markov models…" (Mitchell, 1997)	



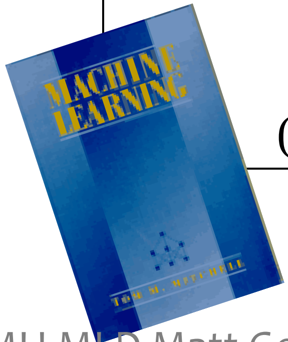
Source: https://www.stonetemple.com/great-knowledge-box-showdown/#VoiceStudyResults

Robotics

2. Learning to drive an autonomous vehicle

THEN	NOW
"…the ALVINN system (Pomerleau 1989) has used its learned strategies to drive unassisted at 70 miles per hour for 90 miles on public highways among other cars…" (Mitchell, 1997)	

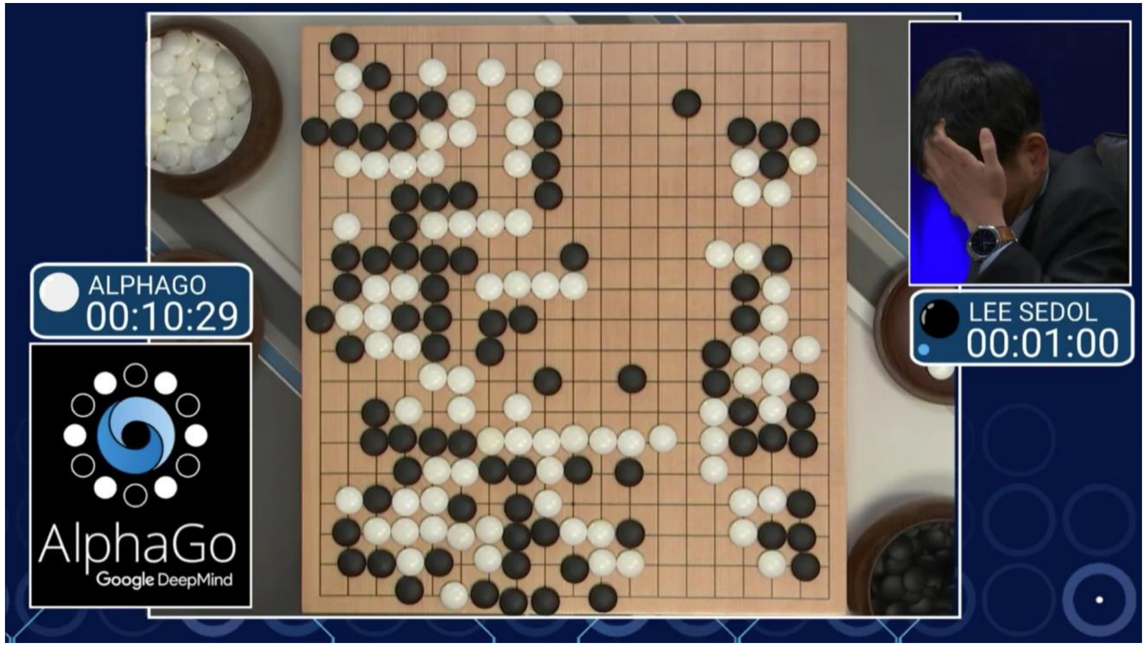
https://www.geek.com/wp-content/uploads/2016/03/uber.jpg

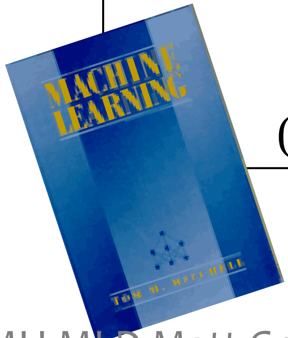


Slide credit: CMU MLD Matt Gormley

Games / Reasoning

3. Learning to beat the masters at board games

THEN	NOW
"…the world's top computer program for backgammon, TD-GAMMON (Tesauro, 1992, 1995), learned its strategy by playing over one million practice games against itself…" (Mitchell, 1997)	



Computer Vision

4. Learning to recognize images

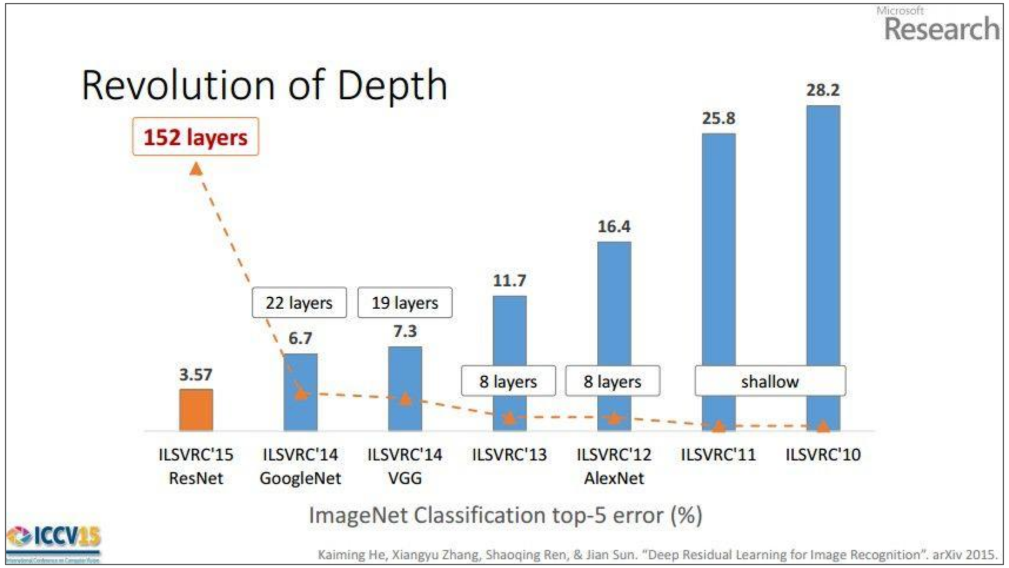
THEN	NOW

"…The recognizer is a convolution network that can be spatially replicated. From the network output, a hidden Markov model produces word scores. The entire system is globally trained to minimize word-level errors.…"

Figure 2: Convolutional neural network character recognizer. This architecture is robust to local translations and distortions, with subsampling, shared weights, and local receptive fields.

number of subsampling layers and the sizes of the kernels are chosen, the sizes of all the layers, including the input, are determined unambiguously. The only architectural parameters that remain to be selected are the number of feature maps in each layer, and the information as to what feature map is connected to what other feature map. In our case, the subsampling rates were chosen as small as possible (2 × 2), and the kernels as small as possible in the first layer (3 × 3) to limit the total number of connections. Kernel sizes in the upper layers are chosen to be as small as

(LeCun et al., 1995)



Slide credit: CMU MLD Matt Gormley

Images from https://blog.openai.com/generative-models/

Learning Theory

- 5. In what cases and how well can we learn?

Sample Complexity Results

Definition 0.1. The **sample complexity** of a learning algorithm is the number of examples required to achieve arbitrarily small error (with respect to the optimal hypothesis) with high probability (i.e. close to 1).

Four Cases we care about…

	Realizable	Agnostic
Finite $\lvert\mathcal{H}\rvert$	$N \geq \frac{1}{\epsilon}[\log(\lvert\mathcal{H}\rvert) + \log(\frac{1}{\delta})]$ labeled examples are sufficient so that with probability $(1-\delta)$ all $h \in \mathcal{H}$ with $R(h) \geq \epsilon$ have $\hat{R}(h) > 0$.	$N \geq \frac{1}{2\epsilon^2}[\log(\lvert\mathcal{H}\rvert) + \log(\frac{2}{\delta})]$ labeled examples are sufficient so that with probability $(1-\delta)$ for all $h \in \mathcal{H}$ we have that $\lvert R(h) - \hat{R}(h)\rvert < \epsilon$.
Infinite $\lvert\mathcal{H}\rvert$	$N = O(\frac{1}{\epsilon}[\text{VC}(\mathcal{H})\log(\frac{1}{\epsilon}) + \log(\frac{1}{\delta})])$ labeled examples are sufficient so that with probability $(1-\delta)$ all $h \in \mathcal{H}$ with $R(h) \geq \epsilon$ have $\hat{R}(h) > 0$.	$N = O(\frac{1}{\epsilon^2}[\text{VC}(\mathcal{H}) + \log(\frac{1}{\delta})])$ labeled examples are sufficient so that with probability $(1-\delta)$ for all $h \in \mathcal{H}$ we have that $\lvert R(h) - \hat{R}(h)\rvert \leq \epsilon$.

Two Types of Error

① True Error (aka. expected risk) (aka. Generalization Error)

$$R(h) = P_{x \sim p^*(x)}(c^*(x) \neq h(x))$$ ← always unknown.

② Train Error (aka. empirical risk)

$$\hat{R}(h) = P_{x \sim S}(c^*(x) \neq h(x))$$

$S = \{x^{(1)}, \ldots, x^{(N)}\}$

$$= \frac{1}{N}\sum_{i=1}^{N} \mathbb{1}(c^*(x^{(i)}) \neq h(x^{(i)}))$$

known, computable

$$= \frac{1}{N}\sum_{i=1}^{N} \mathbb{1}(y^{(i)} \neq h(x^{(i)}))$$

PAC Learning

Q: Can we bound $R(h)$ in terms of $\hat{R}(h)$?

A: Yes!

PAC stands for Probably Approximately Correct

PAC learner yields hypothesis h, which is approximately correct $R(h) \approx 0$ with high probability $\Pr(R(h) \approx 0) \approx 1$

Def: PAC Criterion

$$\Pr(\forall h, |R(h) - \hat{R}(h)| \leq \epsilon) \geq 1 - \delta$$

1. How many examples do we need to learn?
2. How do we quantify our ability to generalize to unseen data?
3. Which algorithms are better suited to specific learning settings?

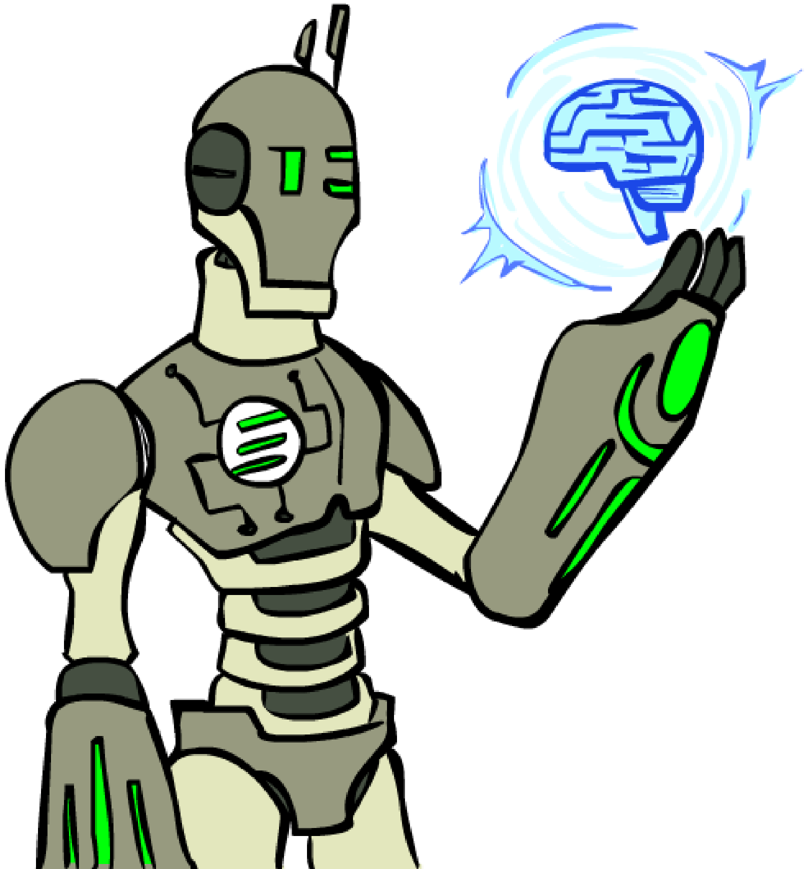
Slide credit: CMU MLD Matt Gormley

10-606 and 10-607

- Mini Courses
  - 10-606
  - 10-607
- Intro ML Courses
  - 10-315
  - 10-301/601
  - 10-701
  - 10-715
- Prerequisites

Today

- Course Info
- Warm-up exercise
- Propositional Logic and Proofs
- ML and 606/607 Intro
- More Course Info



Images: ai.berkeley.edu

Course Information

Website: https://www.cs.cmu.edu/~10607

Canvas: canvas.cmu.edu

Gradescope: gradescope.com

Communication:

- piazza.com
- E-mail (if piazza doesn't work): pvirtue@andrew.cmu.edu

Course Information

Lectures

- Lectures are recorded
  - Shared with our course and ML course staff only
- Participation point earned by answering Piazza polls in lecture
- Quizzes will in lecture, announced two days ahead of time
- Slides will be posted

Recitations

- Recommended attendance
- No plans to record at this point
- No participation points in recitation
- Recitation materials are in-scope for quizzes and exams

Course Information

Office Hours

- OH calendar on course website
- OH-by-appointment requests are certainly welcome

Mental Health