CS189: Intro to Machine Learning
Summer 2018

Lecture 0: Introduction

Ke Li & Josh Tobin
UC Berkeley EECS

Outline for today

- Administrative stuff
- Overview of the course
- ML in the context of AI
- Ordinary Least Squares

Outline for today

- **Administrative stuff**
- Overview of the course
- ML in the context of AI
- Ordinary Least Squares

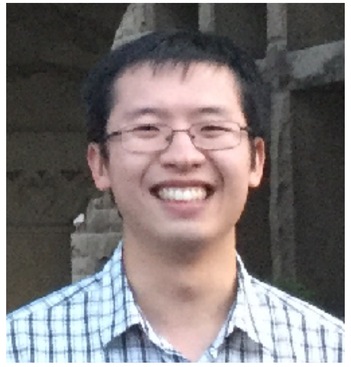
Administrative stuff

Rules for lecture

- No technology!
- No lecture recordings

Course Staff

Instructors



Ke Li

Josh Tobin

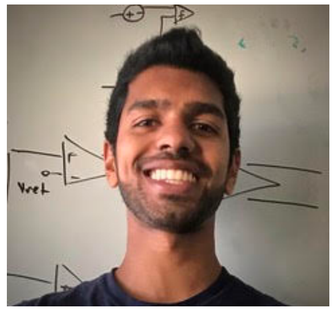
TAs

Sid Reddy

Vedran Hadziosmanovic

Jonathan Lee

Zipeng Qin



Chandan Singh

Head TA

Alvin Wan

www

Website: www.eecs189.org

Piazza: piazza.com/berkeley/summer2018/eecs189289a

Gradescope: https://gradescope.com/

Other course meetings

Discussion Sections

- Schedule on website
- Welcome to go to any that you want!

Review lectures

- Fridays during normal lecture time, room TBD
- THIS WEEK ONLY: Friday 5pm - 7pm at The Woz

Resources for the course

- No textbook
- Useful references:
  - *Elements of Statistical Learning* (Hastie, Tibshirani, Friedman)
  - *Previous semesters course notes* (See website)

Grading

- Weekly HW **30%**
- Midterm (July 19th) **30%**
- Final (August 9th) **40%**

Homeworks

- 7 Total
- Assigned Monday at 10pm & Due following Monday at 10pm *on Gradescope*
- HW0 and HW1 released today (*HW0 Due Friday!)*

Academic honesty

- We take this very seriously
- Simple rule:
  HW groups and Piazza are great, but **no other online communications about course content**

Outline for today

- Administrative stuff
- **Overview of the course**
- ML in the context of AI
- Ordinary Least Squares

Overview of the Course

Prerequisites

- Basic probability & Stats
- Proofs
- Multivariate calculus
- Linear algebra

Why is now a great time to study ML?

Research progress

ImageNet Challenge

IMAGENET

- 1,000 object classes (categories).
- Images:
  - 1.2 M train
  - 100k test.

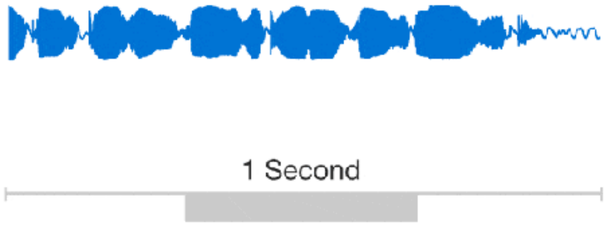


Image classification **Audio synthesis** **Games**

Why is now a great time to study ML?

Products

Voice recognition

Google Translate

Translation

Self-driving cars

Progress in ML is driven by…

- More compute
- More data
- Better algorithms

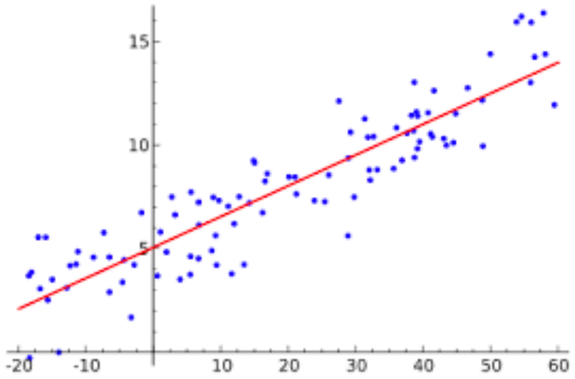
Need more people who understand the algorithms!

Machine Learning

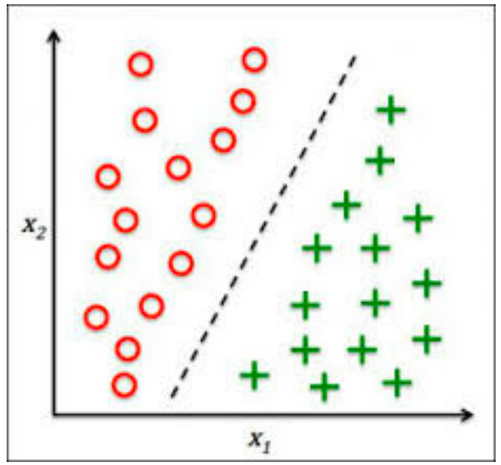
How to discover patterns from data

Types of machine learning

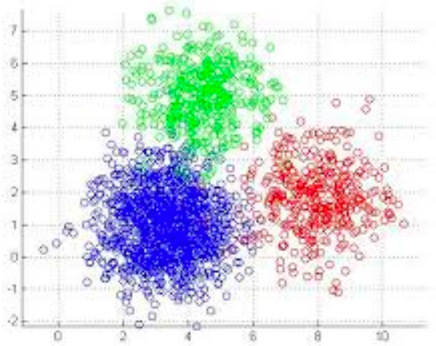
Regression



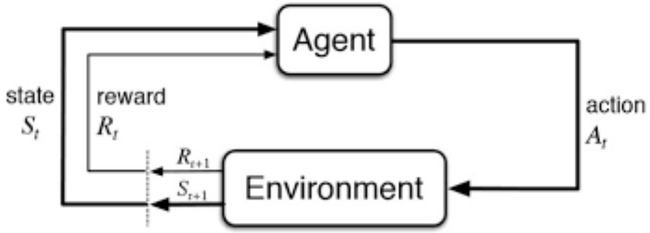
Classification



Unsupervised



Reinforcement



Rough course overview

- **Part 1 (8-10 lecture):** Key ML ideas in context of linear regression
- **Part 2 (~3-4 lectures):** Nonlinear regression
- **Part 3 (~3-4 lectures):** Classification
- **Part 4 (~8 lectures):** Advanced classification
- **Part 5 (remainder)**: Other advanced topics

Outline for today

- Administrative stuff
- Overview of the course
- **ML in the context of AI**
- Ordinary Least Squares

What is Artificial Intelligence (AI)?

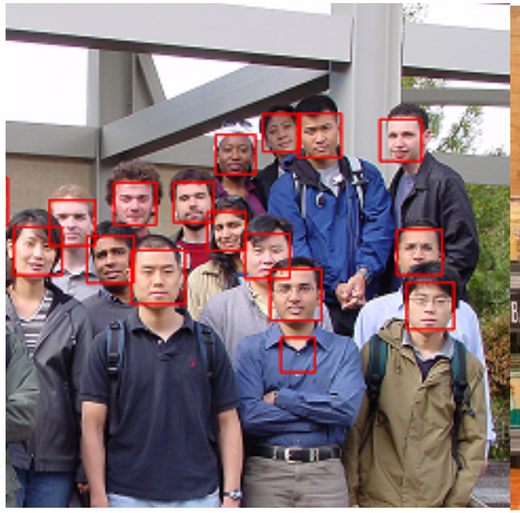
- The study of how to engineer *intelligent* systems/machines.
- What is *intelligence*? Anything that humans can do that machines can't do easily.
  - The ability to see and interpret visual input
  - The ability to read and understand language
  - The ability to move and interact with the world
  - The ability to reason and perform logical deduction

What is Artificial Intelligence (AI)?

- The study of how to engineer *intelligent* systems/machines.
- What is *intelligence*? Anything that humans can do that machines can't do easily.
  - The ability to see and interpret visual input — Computer Vision
  - The ability to read and understand language — Natural Language Processing (NLP)
  - The ability to move and interact with the world — Robotics
  - The ability to reason and perform logical deduction — Traditional AI

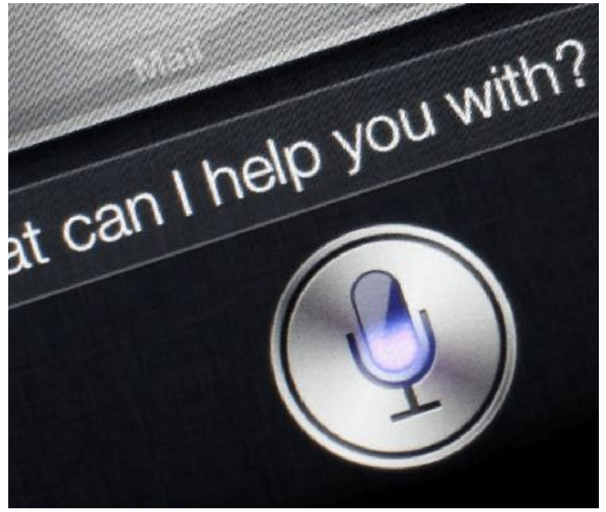
Successes of AI

Computer Vision

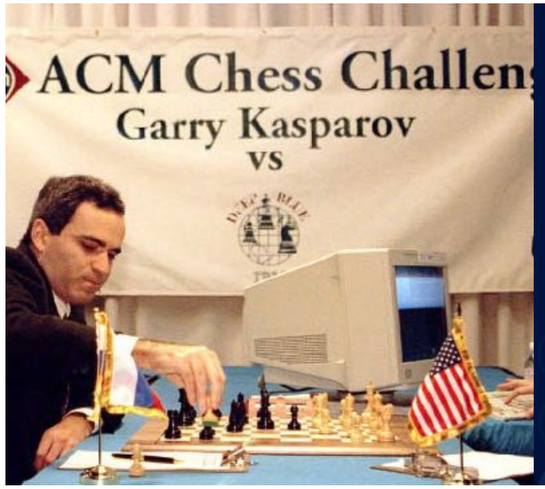


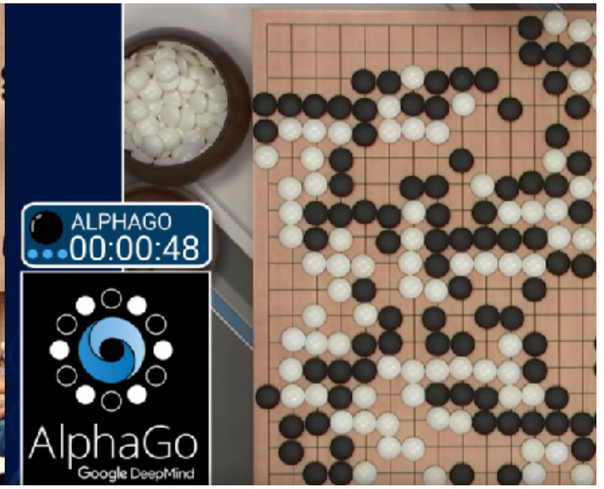
Natural Language Processing (NLP)

Google Translate



Robotics





Traditional AI

Why is AI hard?

- Suppose we want to write a program that recognizes chairs.

- First attempt:
    - A chair is something that has a seat, a back and four legs.

Images taken from ShapeNet

Why is AI hard?

- Suppose we want to write a program that recognizes chairs.

- Second attempt:
    - A chair is something that has a seat, a back and multiple legs.

Images taken from ShapeNet

Why is AI hard?

- Suppose we want to write a program that recognizes chairs.

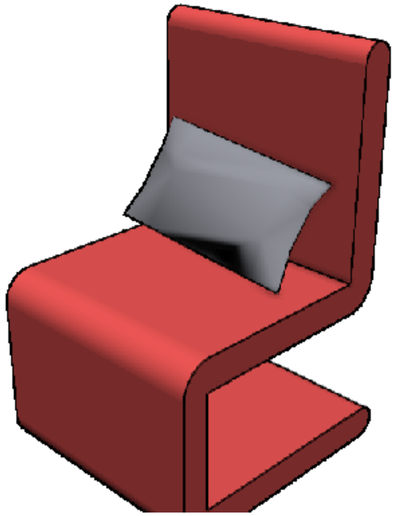
- Third attempt:
    - A chair is something that has a seat, a back and a frame.

Images taken from ShapeNet

Why is AI hard?

- Suppose we want to write a program that recognizes chairs.



- Fourth attempt:
  - A chair is something that has a seat and a back.

Images taken from ShapeNet

Why is AI hard?

- Suppose we want to write a program that recognizes chairs.

- Fifth attempt:
  - A chair is something that has a seat.

Images taken from ShapeNet

Why is AI hard?

- Suppose we want to write a program that recognizes chairs.

- Why is this not a chair?
- There are exceptions to every rule, and exceptions to every exception.

Images taken from ShapeNet

Why is AI hard?

- Problem: The inner workings of our brain are not well understood.
- We don't know *how* our brain converts input to output, so we can't write a program to do so.
- This problem perplexed early computer scientists:

"The Analytical Engine[1] has no pretensions to *originate* anything. It can do *whatever we know how to order it* to perform. "

Ada Lovelace

[1]The Analytical Engine was the first conception of a general-purpose (i.e. Turing complete) computer.

Learning Machines

- Alan Turing proposed the concept of a *learning machine* in 1950 (in the same paper that proposed the Turing test).

- Idea: Divide the problem into two parts:

  - A machine that simulates a child's brain (analogous to a blank notebook: should function by simple mechanisms and have lots of blank sheets)

  - A way of teaching the child machine (should be simple since we know how to teach a human child)

- Teacher rewards good behaviour and penalizes bad behaviour.

Learning Machines

"An important feature of a learning machine is that its teacher will often be very largely ignorant of quite what is going on inside"

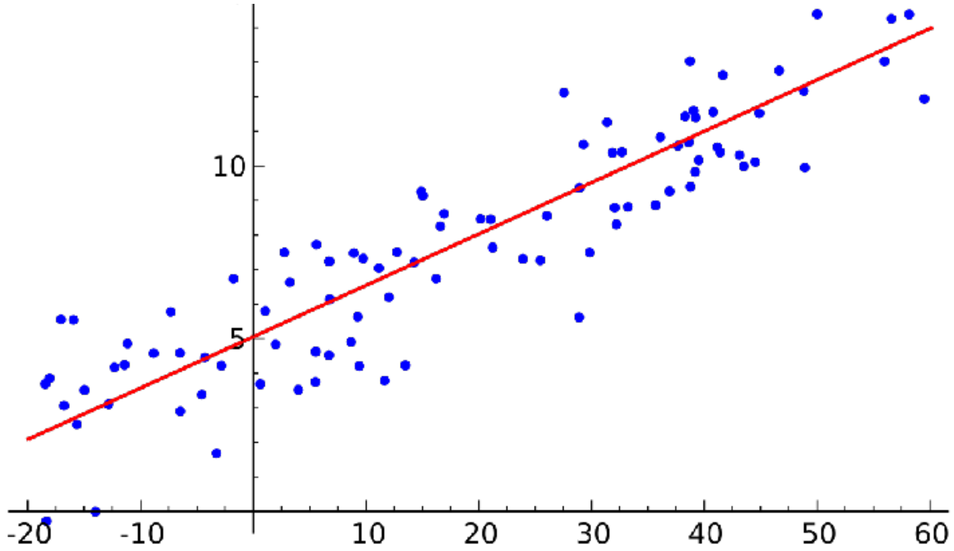
Alan Turing

- While we don't know *how* our brain converts input to output, we know what the output should be for every input.

- We can use this knowledge to teach the machine.

Machine Learning

- In modern terms:
  - Child machine: *Model*
  - Blank sheets: *Model parameters*
  - Teacher: *Loss function*



Input

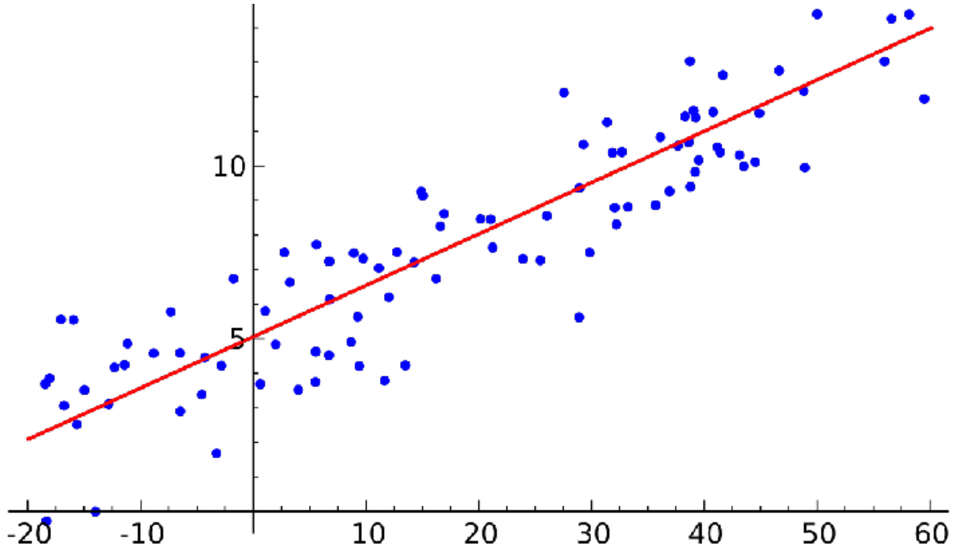
Predicted Output → $\hat{y} = wx + b$

$L = (y - \hat{y})^2$

Desired Output

Machine Learning

- In modern terms:
  - Child machine: *Model*
  - Blank sheets: *Model parameters*
  - Teacher: *Loss function*



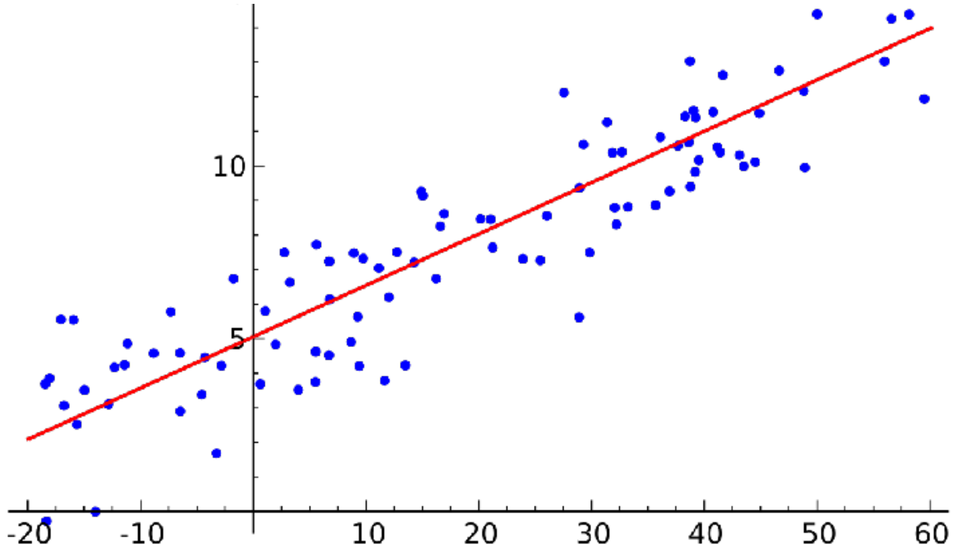
Parameters

Model → $\hat{y} = wx + b$

Loss Function → $L = (y - \hat{y})^2$

Machine Learning

- In modern terms:
  - Child machine: *Model*
  - Blank sheets: *Model parameters*
  - Teacher: *Loss function*



We want to find the parameter values that minimize the loss:

$$w^*, b^* = \arg\min_{w,b} L$$

Outline for today

- Administrative stuff
- Overview of the course
- ML in the context of AI
- **Ordinary Least Squares**

Ordinary least squares

$$y \approx Xw$$

Ordinary least squares

$$y \approx Xw$$

$$\begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix}$$

Ordinary least squares

$$y \approx Xw$$

$$\begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} \approx \begin{bmatrix} \vec{x}_1^T \\ \vdots \\ \vec{x}_n^T \end{bmatrix}$$

Ordinary least squares

$$y \approx Xw$$

$$\begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} \approx \begin{bmatrix} \vec{x}_1^T \\ \vdots \\ \vec{x}_n^T \end{bmatrix} \cdot \begin{bmatrix} w_1 \\ \vdots \\ w_d \end{bmatrix}$$

Ordinary least squares

$$y \approx Xw$$

$$\begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} \approx \begin{bmatrix} \vec{x}_1^T \\ \vdots \\ \vec{x}_n^T \end{bmatrix} \cdot \begin{bmatrix} w_1 \\ \vdots \\ w_d \end{bmatrix}$$

Goal: find the best w

Four levels for ML problems

1. Data & application

1. Data & application

$$y \approx Xw$$

1. Data & application

$$y \approx Xw$$

$$X = \begin{bmatrix} \vec{x}_1^T \\ \vdots \\ \vec{x}_n^T \end{bmatrix}$$

1. Data & application

$$y \approx Xw$$

$$X = \begin{bmatrix} \vec{x}_1^T \\ \vdots \\ \vec{x}_n^T \end{bmatrix} \qquad \vec{x}_i = \begin{bmatrix} x_i^1 \\ \vdots \\ x_i^d \end{bmatrix}$$

1. Data & application

$$y \approx Xw$$

$$X = \begin{bmatrix} \vec{x}_1^T \\ \vdots \\ \vec{x}_n^T \end{bmatrix} \qquad \vec{x}_i = \begin{bmatrix} x_i^1 \\ \vdots \\ x_i^d \end{bmatrix} \qquad \vec{y} = \begin{bmatrix} y_i \\ \vdots \\ y_n \end{bmatrix}$$

Four levels for ML problems

1. Data & application
2. Model

2. Model

$$y \approx Xw$$

$$\hat{y} = Xw$$

Four levels for ML problems

1. Data & application
2. Model
3. Optimization problem

3. Optimization problem

$$y \approx Xw$$

3. Optimization problem

$$y \approx Xw$$

Goal: find the best w

3. Optimization problem

$$y \approx Xw$$

$$\min_{w}$$

Goal: find the best w

3. Optimization problem

$$y \approx Xw$$

$$\min_{w} ||Xw - y||_2^2$$

Goal: find the best w

3. Optimization problem

$$||Xw - y||_2^2$$

3. Optimization problem

$$||Xw - y||_2^2 = \sum_{i=1}^{n} \left(\vec{x_i}^T w - y_i \right)^2$$

3. Optimization problem

$$||Xw - y||_2^2 = \sum_{i=1}^{n} \left(\vec{x_i}^T w - y_i \right)^2$$

$$= \sum_{i=1}^{n} \left(\hat{y}_i - y_i \right)^2$$

Four levels for ML problems

1. Data & application
2. Model
3. Optimization problem
4. Optimization algorithm

Four levels for ML problems

1. Data & application
2. Model
3. Optimization problem
4. **Optimization algorithm**

4. Optimization algorithms

$$\min_{w} ||Xw - y||_2^2$$

4. Optimization algorithms

$$\min_{w} ||Xw - y||_2^2$$

$$\hat{w} = \left(X^T X\right)^{-1} X^T y$$

Reminders

- Sign up for Piazza
- Look out for HW0 and HW1
- Discussion sections start Wednesday