10-601 Introduction to Machine Learning

Machine Learning Department
School of Computer Science
Carnegie Mellon University

Machine Learning in Practice + k-Nearest Neighbors

Intro Readings:
Mitchell 1
HTF 1, 2
Murphy 1
Bishop 1

KNN Readings:
Mitchell 8.2
HTF 13.3
Murphy ---
Bishop 2.5.2

Matt Gormley
Lecture 2
January 23, 2016

Reminders

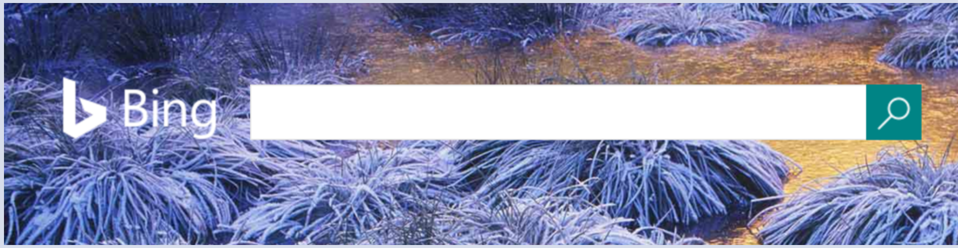
- Background Test
  - Tue, Jan. 24 at 6:30pm
  - **Your test location depends on your registration status – see Piazza for details
- Background Exercises (Homework 1)
  - Released: Tue, Jan. 24 after the test
  - Due: Mon, Jan. 30 at 5:30pm

Machine Learning & Ethics

What ethical responsibilities do we have as machine learning experts?

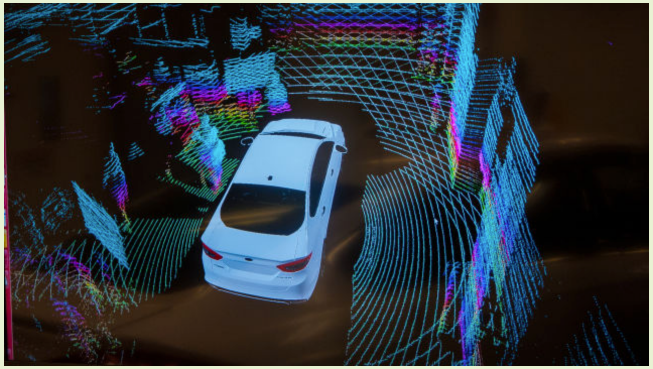
Some topics that we won't cover are probably deserve an entire course

If our search results for news are optimized for ad revenue, might they reflect gender / racial / socio-economic biases?



http://bing.com/

http://arstechnica.com/



How do autonomous vehicles make decisions when all of the outcomes are likely to be negative?

Should restrictions be placed on intelligent agents that are capable of interacting with the world?

http://vizdoom.cs.put.edu.pl/

Outline

- **Defining Learning Problems**
  - Artificial Intelligence (AI)
  - Mitchell's definition of learning
  - Example learning problems
  - Data annotation
  - The Machine Learning framework
- **Classification**
  - Binary classification
  - 2D examples
  - Decision rules / hypotheses
- **k-Nearest Neighbors (KNN)**
  - KNN for binary classification
  - Distance functions
  - Special cases
  - Choosing k

Covered Next Lecture

This section is based on Chapter 1 of (Mitchell, 1997)

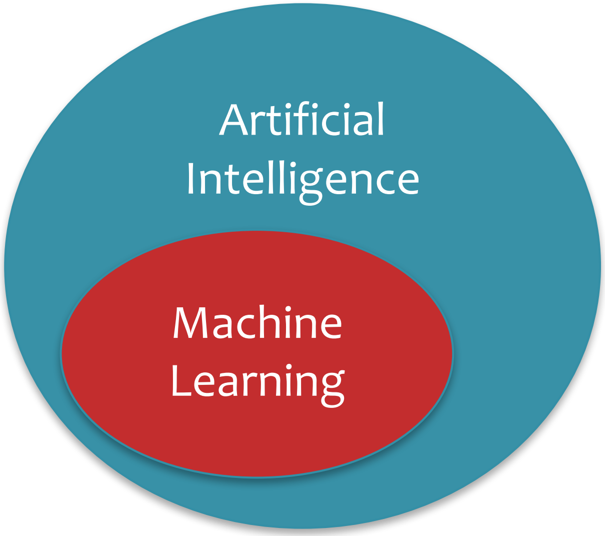
DEFINING LEARNING PROBLEMS

Artificial Intelligence

The basic goal of AI is to develop intelligent machines.

This consists of many sub-goals:

- Perception
- Reasoning
- Control / Motion / Manipulation
- Planning
- Communication
- Creativity
- Learning



Amazon Go

https://www.amazon.com/b?node=16008589011

https://www.youtube.com/watch?v=NrmMk1Myrxc

Artificial Intelligence (AI): Example Tasks:

- Identify objects in an image
- Translate from one human language to another
- Recognize speech
- Assess risk (e.g. in loan application)
- Make decisions (e.g. in loan application)
- Assess potential (e.g. in admission decisions)
- Categorize a complex situation (e.g. medical diagnosis)
- Predict outcome (e.g. medical prognosis, stock prices, inflation, temperature)
- Predict events (default on loans, quitting school, war)
- Plan ahead under perfect knowledge (chess)
- Plan ahead under partial knowledge (Poker, Bridge)

Well-Posed Learning Problems

Three components:

1. Task, *T*
2. Performance measure, *P*
3. Experience, *E*

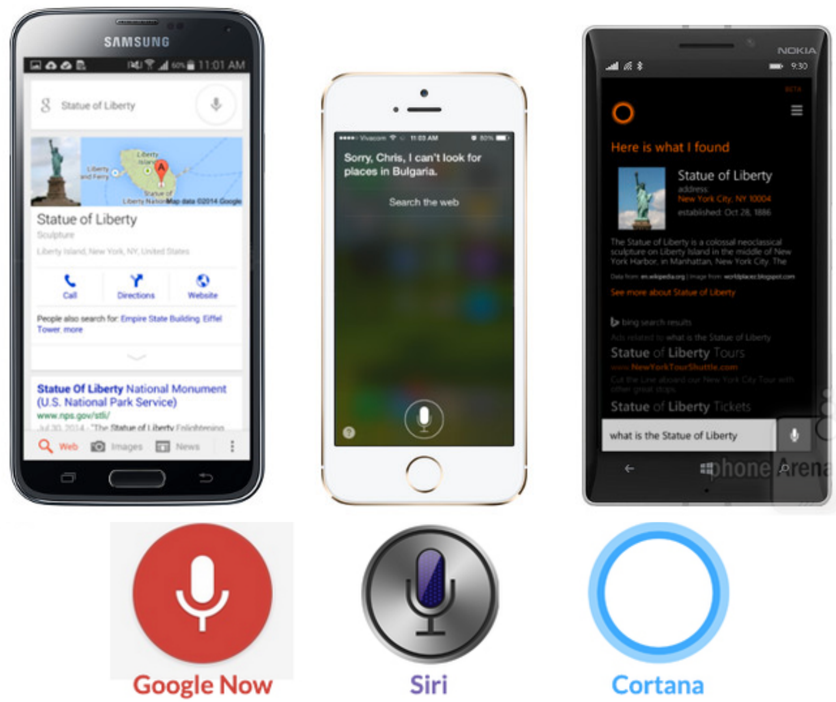
Mitchell's definition of learning:

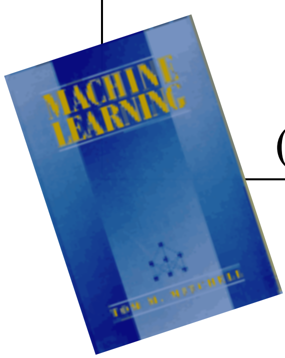
A computer program **learns** if its performance at tasks in *T*, as measured by *P*, improves with experience *E*.

Definition from (Mitchell, 1997)

Example Learning Problems (historical perspective)

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)	

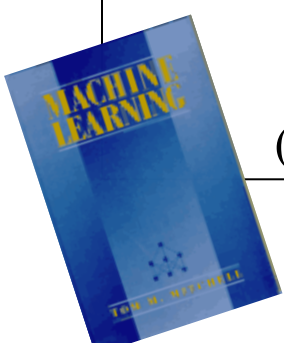


Source: https://www.stonetemple.com/great-knowledge-box-showdown/#VoiceStudyResults

Example Learning Problems (historical perspective)

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)	 waymo.com

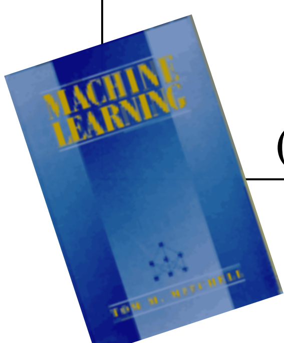


Example Learning Problems (historical perspective)

2. Learning to drive an autonomous vehicle

THEN

"…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)

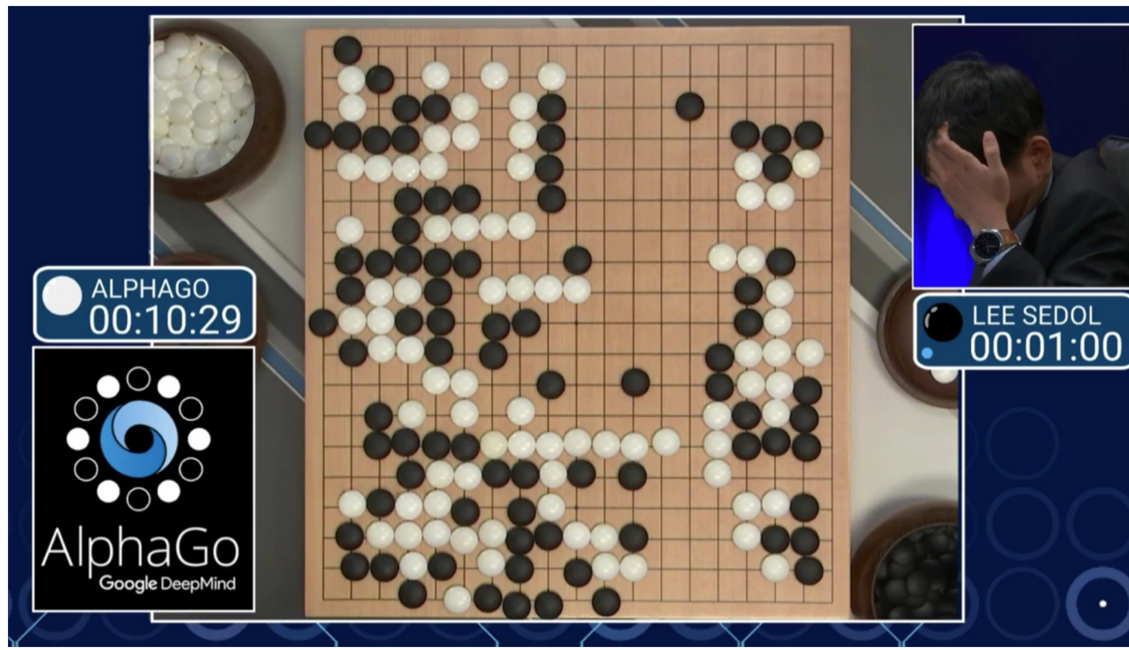
NOW

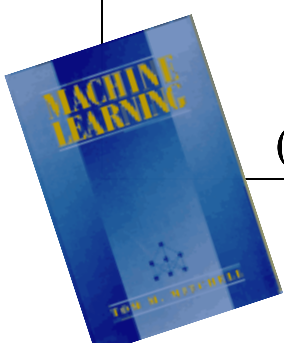
https://www.geek.com/wp-content/uploads/2016/03/uber.jpg

Example Learning Problems (historical perspective)

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)	





Example Learning Problems

3. Learning to beat the masters at **chess**
   1. Task, *T*:

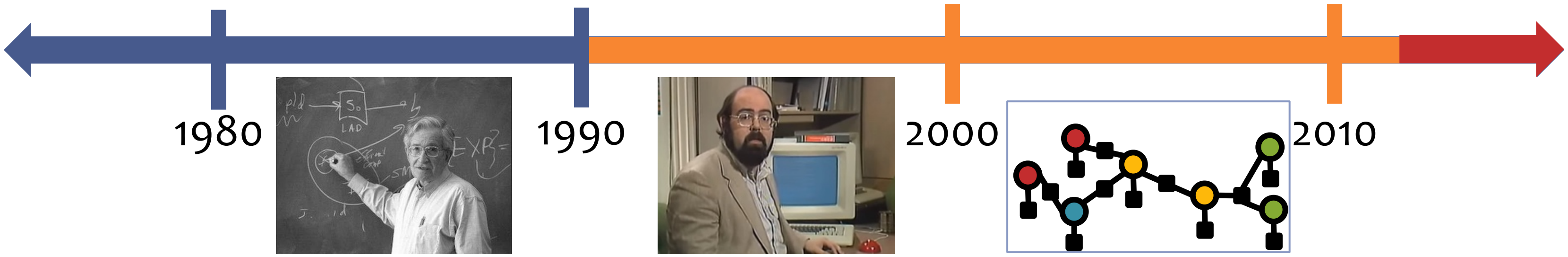
   2. Performance measure, *P*:

   3. Experience, *E*:

Example Learning Problems

4. Learning to **respond to voice commands (Siri)**
    1. Task, *T*:
    2. Performance measure, *P*:
    3. Experience, *E*:

Capturing the Knowledge of Experts



Solution #1: Expert Systems

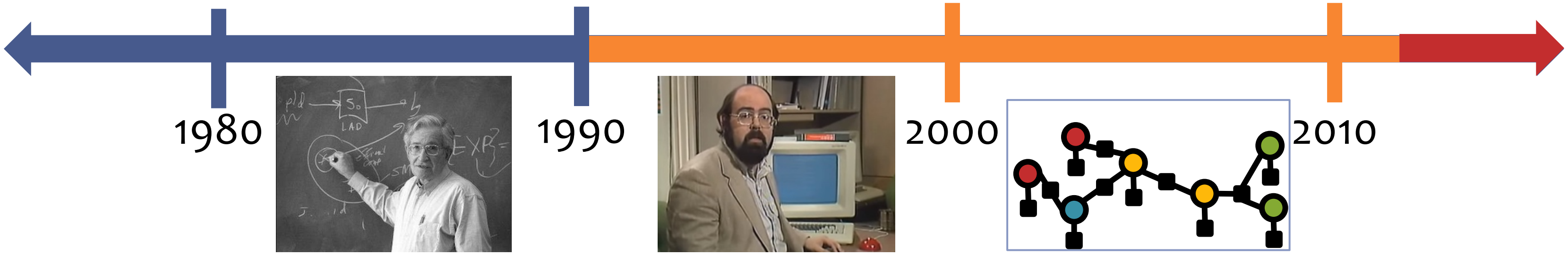
- Over 20 years ago, we had rule based systems
- Ask the expert to
  1. Obtain a PhD in Linguistics
  2. Introspect about the structure of their native language
  3. Write down the rules they devise

Give me directions to Starbucks
If: "give me directions to X" Then: directions(here, nearest(X))

How do I get to Starbucks?
If: "how do i get to X" Then: directions(here, nearest(X))

Where is the nearest Starbucks?
If: "where is the nearest X" Then: directions(here, nearest(X))

Capturing the Knowledge of Experts



Solution #1: Expert Systems

- Over 20 years ago, we had rule based systems
- Ask the expert to
  1. Obtain a PhD in Linguistics
  2. Introspect about the structure of their native language
  3. Write down the rules they devise

I need directions to Starbucks

```
If: "I need directions to X"
Then: directions(here, nearest(X))
```

Starbucks directions

```
If: "X directions"
Then: directions(here, nearest(X))
```

Is there a Starbucks nearby?

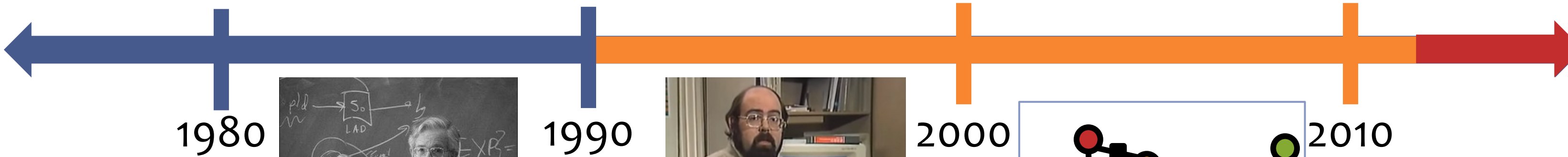
```
If: "Is there an X nearby"
Then: directions(here, nearest(X))
```

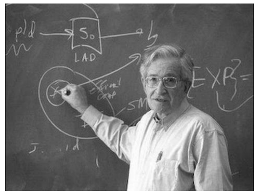
Capturing the Knowledge of Experts

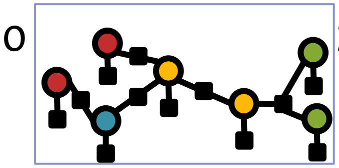
Solution #2: Annotate Data and Learn

- Experts:
  - **Very good at** answering questions about specific cases
  - **Not very good at** telling **HOW** they do it
- 1990s: So why not just have them tell you what they do on **SPECIFIC CASES** and then let **MACHINE LEARNING** tell you how to come to the same decisions that they did

Capturing the Knowledge of Experts







Solution #2: Annotate Data and Learn

1. Collect raw sentences $\{x_1, \ldots, x_n\}$
2. Experts annotate their meaning $\{y_1, \ldots, y_n\}$

x_1: How do I get to Starbucks?

y_1: `directions(here, nearest(Starbucks))`

x_2: Show me the closest Starbucks

y_2: `map(nearest(Starbucks))`

x_3: Send a text to John that I'll be late

y_3: `txtmsg(John, I'll be late)`

x_4: Set an alarm for seven in the morning

y_4: `setalarm(7:00AM)`

Example Learning Problems

4. Learning to **respond to voice commands (Siri)**
   1. Task, *T*:
      predicting action from speech
   2. Performance measure, *P*:
      percent of correct actions taken in user pilot study
   3. Experience, *E*:
      examples of (speech, action) pairs

The Machine Learning Framework

- Formulate a task as a mapping from input to output
  - Task examples will usually be pairs: (input, correct_output)
- Formulate performance as an error measure
  - or more generally, as an objective function (aka Loss function)
- Examples:
  - Medical Diagnosis
    - mapping input to one of several classes/categories ➔ Classification
  - Predict tomorrow's Temperature
    - mapping input to a number ➔ Regression
  - Chance of Survival: From patient data to p(survive >= 5 years)
    - mapping input to probability ➔ Density estimation
  - Driving recommendation
    - mapping input into a plan ➔ Planning

Choices in ML Formulation

Often, the same task can be formulated in more than one way:

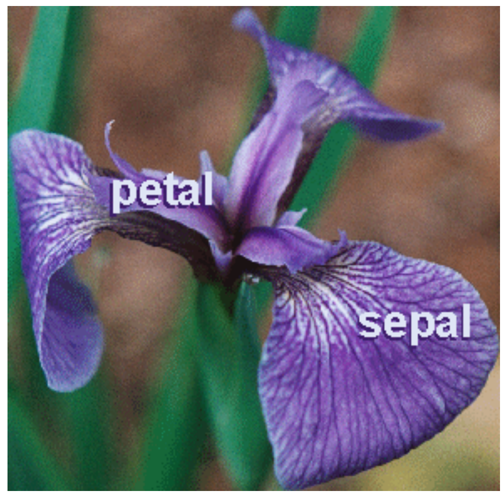
- Ex. 1: Loan applications
  - creditworthiness/score (regression)
  - probability of default (density estimation)
  - loan decision (classification)
- Ex. 2: Chess
  - Nature of available training examples/experience:
    - expert advice (painful to experts)
    - games against experts (less painful but limited, and not much control)
    - experts' games (almost unlimited, but only "found data" – no control)
    - games against self (unlimited, flexible, but can you learn this way?)
  - Choice of target function: board→move vs. board→score

How to Approach a Machine Learning Problem

1. Consider your goal → definition of task **T**
   – E.g. make good loan decisions, win chess competitions, …
2. Consider the nature of available (or potential) experience **E**
   – How much data can you get? What would it cost (in money, time or effort)?
3. Choose type of output **O** to learn
   – (Numerical? Category? Probability? Plan?)
4. Choose the Performance measure **P** (error/loss function)

5. Choose a representation for the input X
6. Choose a set of possible solutions **H** (hypothesis space)
   – set of functions h: X ➔ O
   – (often, by choosing a representation for them)
7. Choose or design a learning algorithm
   – for using examples (**E**) to converge on a member of **H** that optimizes **P**

CLASSIFICATION


petal
sepal

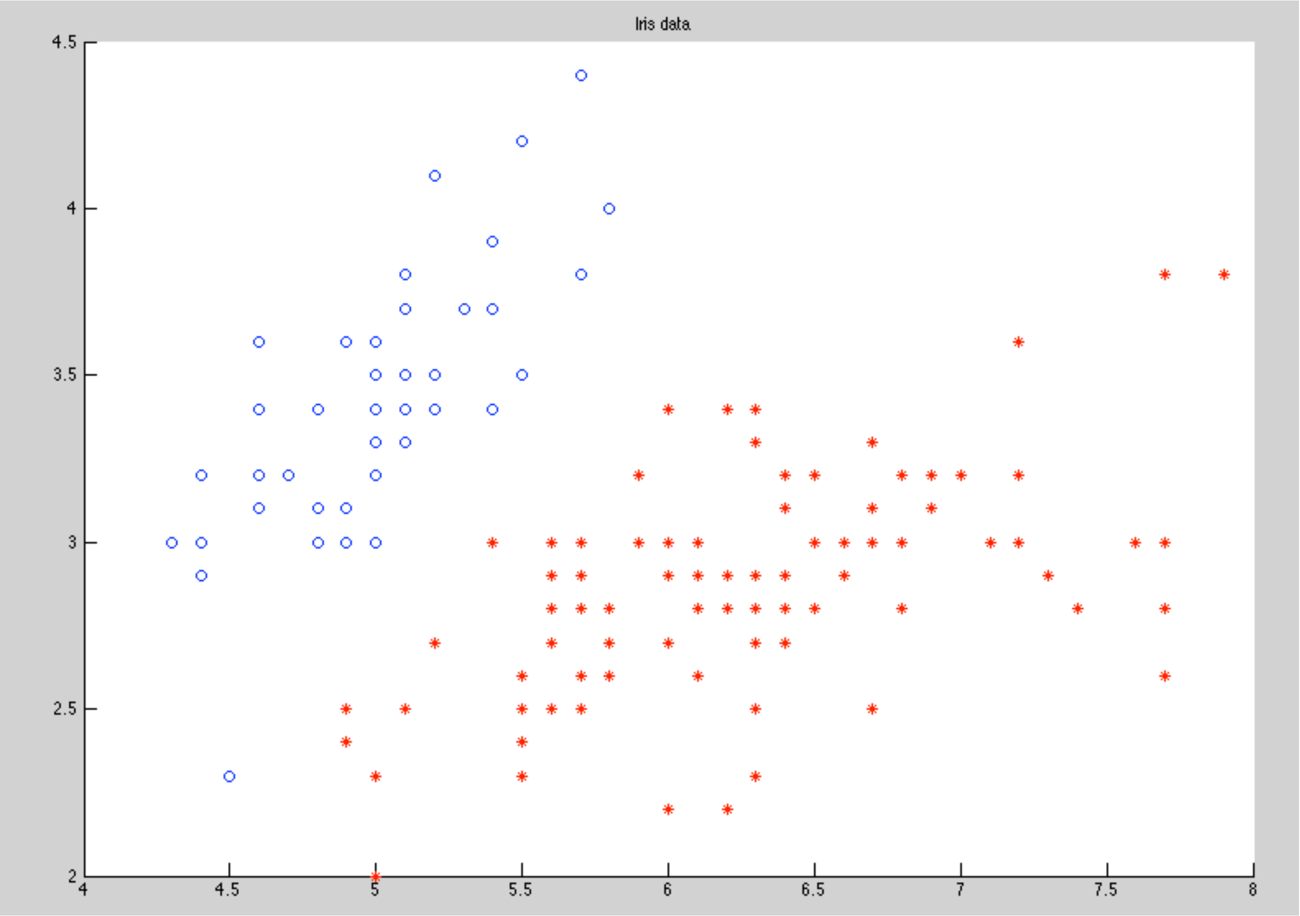
Fisher Iris Dataset

Fisher (1936) used 150 measurements of flowers from 3 different species: Iris setosa (0), Iris virginica (1), Iris versicolor (2) collected by Anderson (1936)

Species	Sepal Length	Sepal Width	Petal Length	Petal Width
0	4.3	3.0	1.1	0.1
0	4.9	3.6	1.4	0.1
0	5.3	3.7	1.5	0.2
1	4.9	2.4	3.3	1.0
1	5.7	2.8	4.1	1.3
1	6.3	3.3	4.7	1.6
1	6.7	3.0	5.0	1.7

Full dataset: https://en.wikipedia.org/wiki/Iris_flower_data_set

Fisher Iris Dataset



Classification

Whiteboard:

- Binary classification
- 2D examples
- Decision rules / hypotheses

K-NEAREST NEIGHBORS

k-Nearest Neighbors

Whiteboard:

- KNN for binary classification
- Distance functions

Takeaways

- **Learning Problems**
  – Defining a learning problem is tricky
  – Formalizing exposes the many possibilities
- **k-Nearest Neighbors**
  – KNN is an extremely simple algorithm for classification