Unsupervised and Reinforcement Learning

Nupur Chatterji, Kenny Marino, Colin White

Outline

- Clustering
- Clustering Questions
- PCA
- PCA Questions
- Reinforcement Learning

Clustering

Unsupervised Learning - unlabeled data

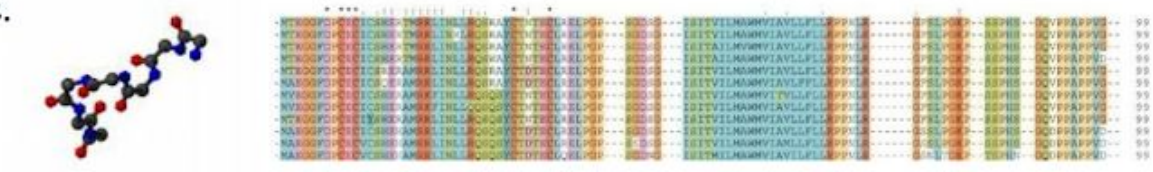
- Automatically organize data
- Understand structure in data
- Preprocessing for further analysis

Applications (Clustering comes up everywhere…)

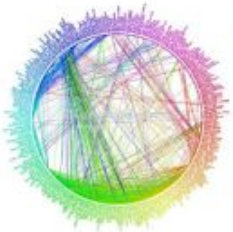
- Cluster news articles or web pages or search results by topic.

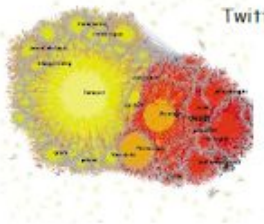


- Cluster protein sequences by function or genes according to expression profile.



- Cluster users of social networks by interest (community detection).


Facebook network


Twitter Network

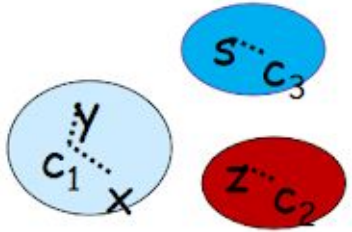
Objective Based Clustering

Input: A set S of n points, also a distance/dissimilarity measure specifying the distance d(x,y) between pairs (x,y).

E.g., # keywords in common, edit distance, wavelets coef., etc.

Goal: output a partition of the data.

- k-means: find center pts $c_1, c_2, \ldots, c_k$ to minimize $\sum_{i=1}^{n} \min_{j\in\{1,\ldots,k\}} d^2(\mathbf{x^i}, \mathbf{c_j})$
- k-median: find center pts $\mathbf{c_1}, \mathbf{c_2}, \ldots, \mathbf{c_k}$ to minimize $\sum_{i=1}^{n} \min_{j\in\{1,\ldots,k\}} d(\mathbf{x^i}, \mathbf{c_j})$



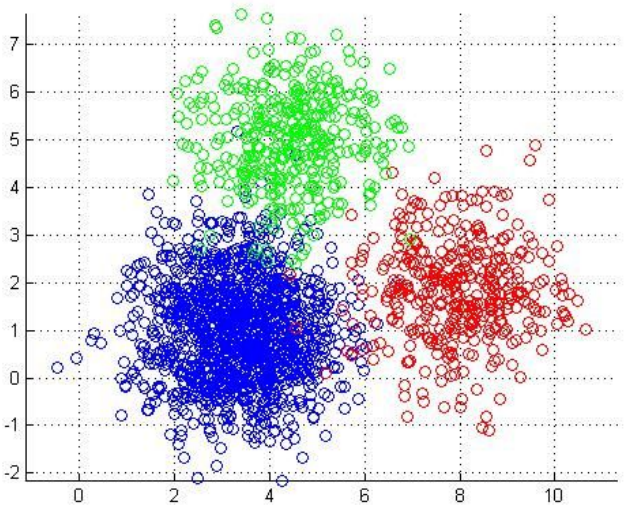
- K-center: find partition to minimize the maximum radius

Euclidean k-means Clustering

- **Input:** a set of n points, $x^1, x^2, \ldots, x^n$, in R^d, an integer k
- **Output:** k "centers" $c_1, c_2, \ldots, c_k$

Try to minimize the distance from each point x^i to its closest center

$$\sum_{i=1}^{n} \min_{j \in \{1,\ldots,k\}} \left|\left|\mathbf{x^i} - \mathbf{c_j}\right|\right|^2$$



K-means complexity

- Hard to solve even when k=2 and d=2
- k=1 is easy to solve
- d=1 is easy to solve (dynamic programming)

Common Heuristic in Practice: The Lloyd's method

[Least squares quantization in PCM, Lloyd, IEEE Transactions on Information Theory, 1982]

Input: A set of n datapoints $\mathbf{x^1}, \mathbf{x^2}, \ldots, \mathbf{x^n}$ in $\mathrm{R^d}$

Initialize centers $\boldsymbol{c_1}, \boldsymbol{c_2}, \ldots, \boldsymbol{c_k} \in \mathrm{R^d}$ and clusters $\mathrm{C_1}, \mathrm{C_2}, \ldots, \mathrm{C_k}$ in any way.

Repeat until there is no further change in the cost.

- For each j: $\mathrm{C_j} \leftarrow \{x \in S$ whose closest center is $\mathbf{c_j}\}$
- For each j: $\mathbf{c_j} \leftarrow$ mean of $\mathrm{C_j}$

Holding $\boldsymbol{c_1}, \boldsymbol{c_2}, \ldots, \boldsymbol{c_k}$ fixed, pick optimal $\mathrm{C_1}, \mathrm{C_2}, \ldots, \mathrm{C_k}$

Holding $\mathrm{C_1}, \mathrm{C_2}, \ldots, \mathrm{C_k}$ fixed, pick optimal $\boldsymbol{c_1}, \boldsymbol{c_2}, \ldots, \boldsymbol{c_k}$

Lloyd's initialization

Initialization is very important for Lloyd's method

- Random initialization
- Farthest-first traversal: iteratively choose farthest point from current set
- d^2-sampling (k-means++) iteratively choose a point v with probability $d_{min}(v,C)^2$, where C is the list of current centers

Theorem: k-means++ always attains an O(log k) approximation to the optimal k-means solution in expectation.

K-means runtime

- K-means++ initialization O(nkd) time
- Lloyd's method: O(nkd) time

Exponential number of rounds in the worst case

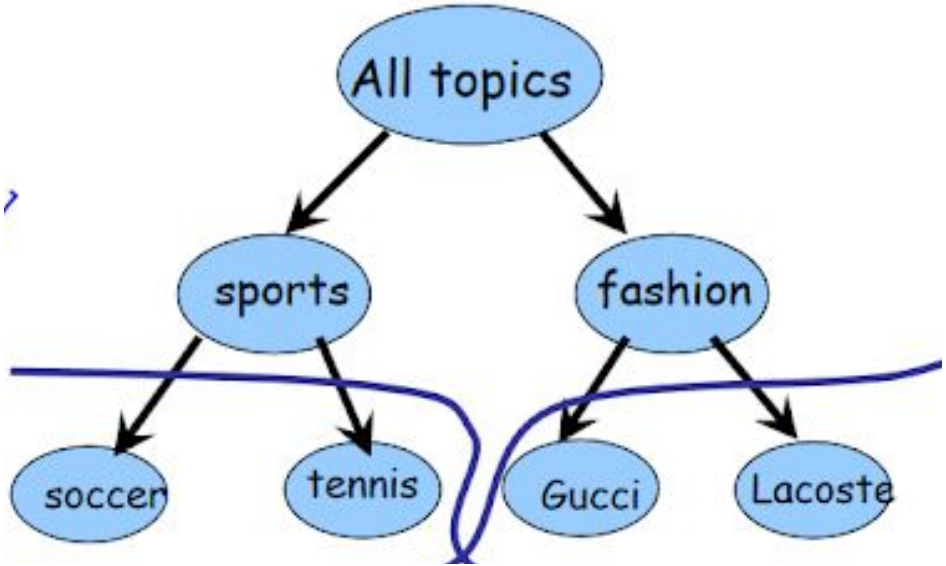
Small number of rounds in practice

Expected number of rounds is polynomial time under *smoothed analysis*

Hierarchical Clustering

- What if we don’t know the right value of k? (often the case)

Often leads to natural solutions

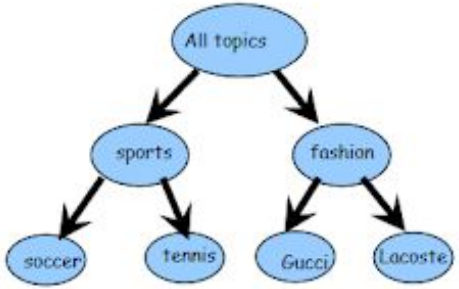


Bottom-Up (agglomerative)

Have a distance measure on pairs of objects.

d(x,y) – distance between x and y

E.g., # keywords in common, edit distance, etc



- Single linkage: $\text{dist}(C, C') = \min_{x \in C, x' \in C'} \text{dist}(x, x')$
- Complete linkage: $\text{dist}(C, C') = \max_{x \in C, x' \in C'} \text{dist}(x, x')$
- Average linkage: $\text{dist}(C, C') = \text{avg}_{x \in C, x' \in C'} \text{dist}(x, x')$

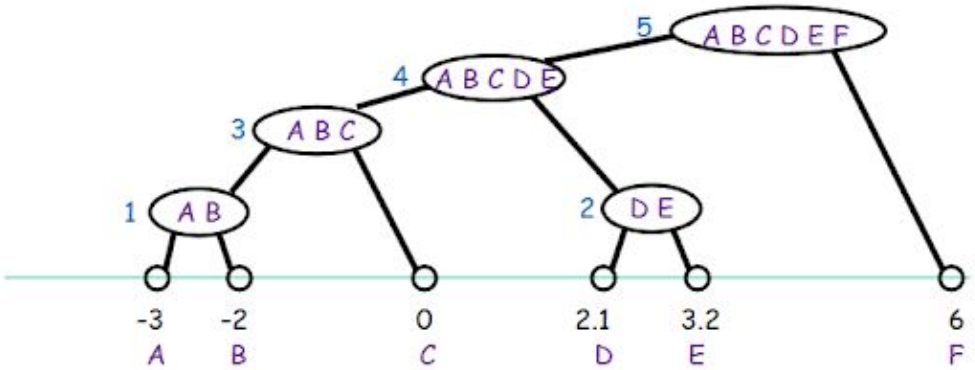
Single Linkage

Bottom-up (agglomerative)

- Start with every point in its own cluster.
- Repeatedly merge the "closest" two clusters.

Single linkage: $\text{dist}(C, C') = \min_{x \in C, x' \in C'} \text{dist}(x, x')$

Dendogram



Runtime:

- $O(n^3)$ is easy
- Can achieve $O(n^2 \log n)$

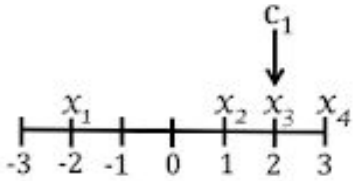
k-Means%%

- Choose the first cluster center, c_1, uniformly at random from among the data points.
- For $j = 2, \ldots, k$ iteratively choose c_j to be one of the data points according to the following weighted probability

$$P(c_j = x) \propto \begin{cases} 0 & \text{if } x = c_\ell \text{ for } \ell = 1, \ldots, j-1 \\ \min_{\ell<j} \frac{1}{||x - c_\ell||} & \text{otherwise} \end{cases}$$

Assume that $x \in \mathbb{R}^1$. Answer the following questions about "K-means %%":

(a) [4 pts] Assume we have three data points $(x_1 = -2), (x_2 = 1), (x_3 = 2), (x_4 = 3)$ and we set $k = 2$. If we choose the first cluster center to be $c_1 = 2$, compute the following probabilities for the second cluster center, c_2.



- Is this a good initialization algorithm?

We are given data drawn from two independent multivariate Gaussian distributions. 100 data points are drawn from $N([0,0], I * 0.3)$ and 10 data points are drawn from $N([2,2], I * 0.3)$. The data are shown in Figure 1 with the initial cluster centers as red crosses.

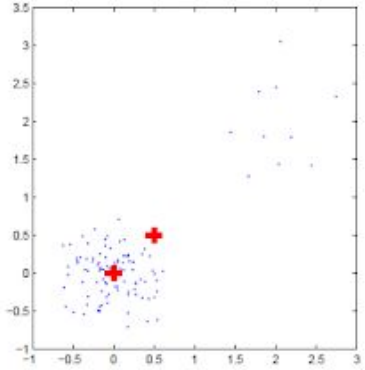
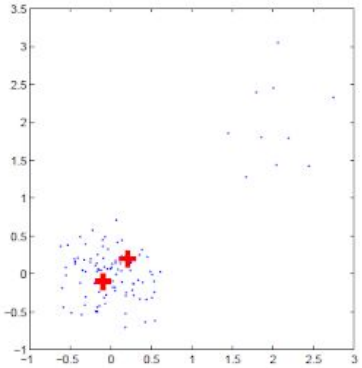
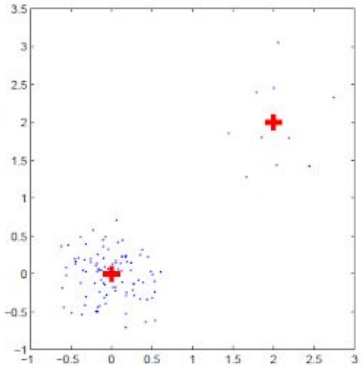
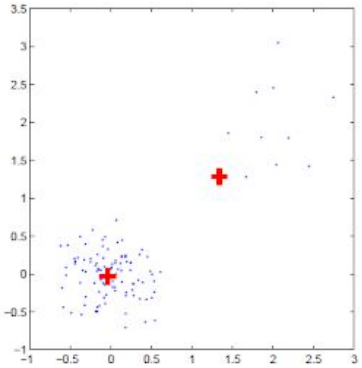
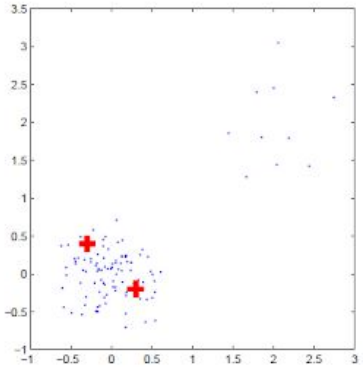


Figure 1: Initial data and cluster centers









- What are the new centers after one Lloyd iteration?

2.1 General K-means questions (circle one answer)

(a) We are given a dataset of n samples, $x_1, ..., x_n$ where $x \in \mathbb{R}^2$ and asked to run k-means. If we choose the value for k to optimize the objective function how many clusters will be used (i.e. what value of k will we choose)?

(i) 1 (ii) 2 (iii) n (iv) n^2

(b) Why do we use Lloyds algorithm to compute the K-means solution instead of the brute force solution? Recall that the brute force solution for some value k requires searching over all possible assignments of the data points into k clusters.

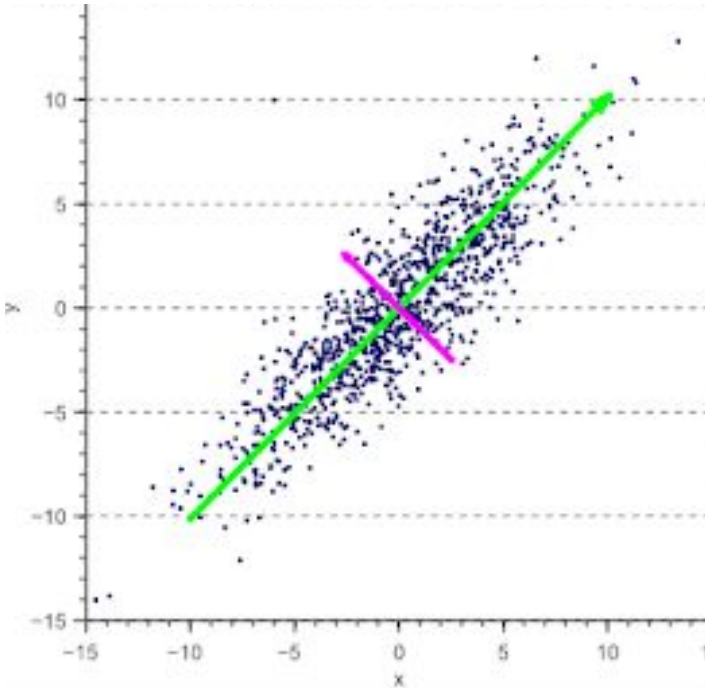
(i) Lloyd's algorithm is guaranteed to find a solution.

(ii) Lloyd's algorithm is guaranteed to converge in $O(n)$ steps.

(iii) The brute force algorithm can take an exponential number of steps (in worst case).

(iv) The brute force algorithm can take a polynomial number of steps (in worst case).

Principal Component Analysis

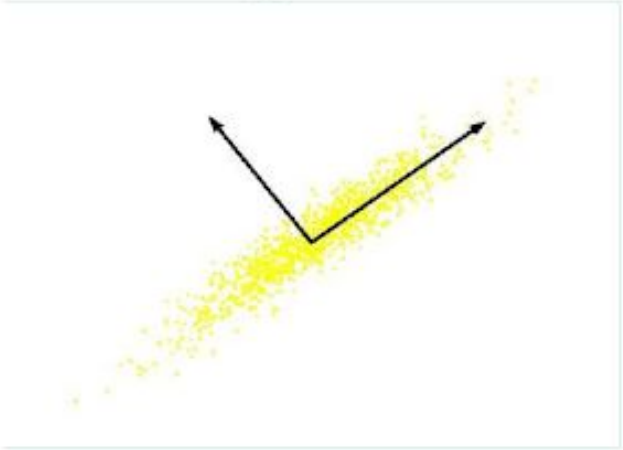


PCA, Kernel PCA, ICA: Powerful unsupervised learning techniques for extracting hidden (potentially lower dimensional) structure from high dimensional datasets.

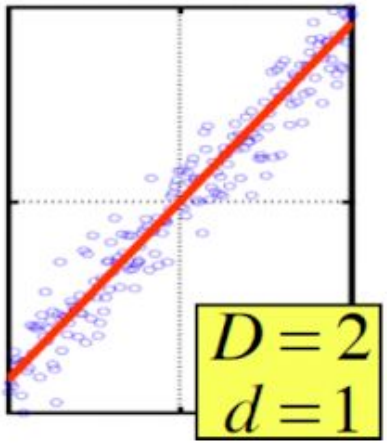
Useful for:

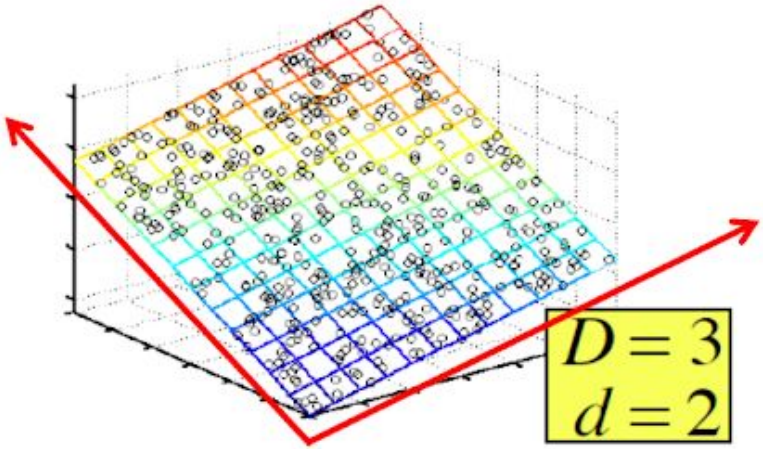
- Visualization
- More efficient use of resources (e.g., time, memory, communication)
- Statistical: fewer dimensions → better generalization
- Noise removal (improving data quality)
- Further processing by machine learning algorithms

What is PCA: Unsupervised technique for extracting variance structure from high dimensional datasets.



- PCA is an orthogonal projection or transformation of the data into a (possibly lower dimensional) subspace so that the variance of the projected data is maximized.

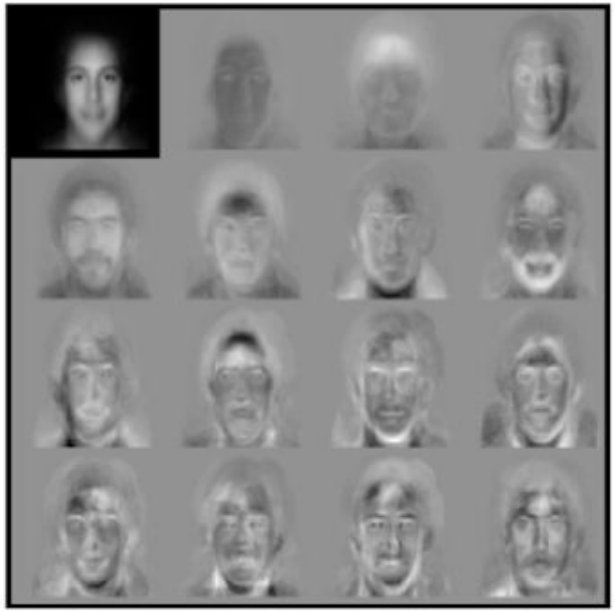




In case where data lies on or near a low d-dimensional linear subspace, axes of this subspace are an effective representation of the data.

Identifying the axes is known as Principal Components Analysis, and can be obtained by using classic matrix computation tools (Eigen or Singular Value Decomposition).

Example: faces



Eigenfaces from 7562 images:

top left image is linear combination of rest.

Sirovich & Kirby (1987)
Turk & Pentland (1991)

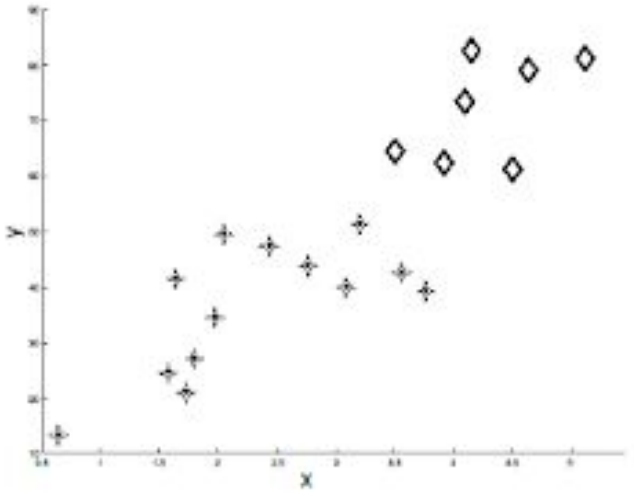
Can represent a face image using just 15 numbers!

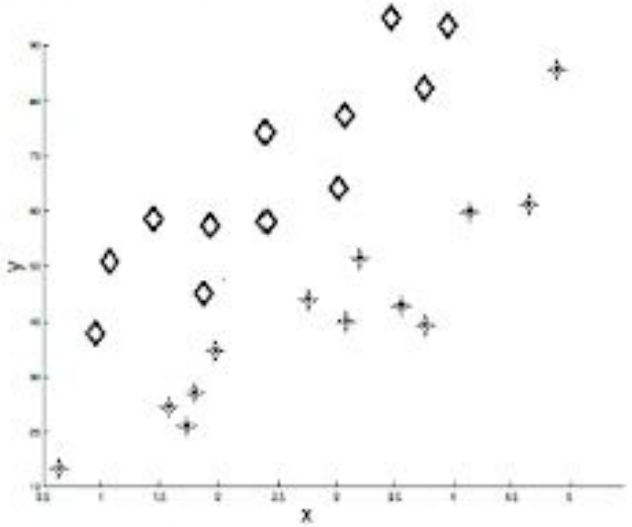
(a) In the following plots, a train set of data points belonging to two classes on a two dimensional space are given, where the original features are the coordinates (x, y). For each, answer the following questions:

(i) Draw the principal components

(ii) Can you use the PC representation of the data to create a classification rule that has 100% accuracy on the train set? Explain how this classification rules would look like or why you cannot do it.

Dataset 1:

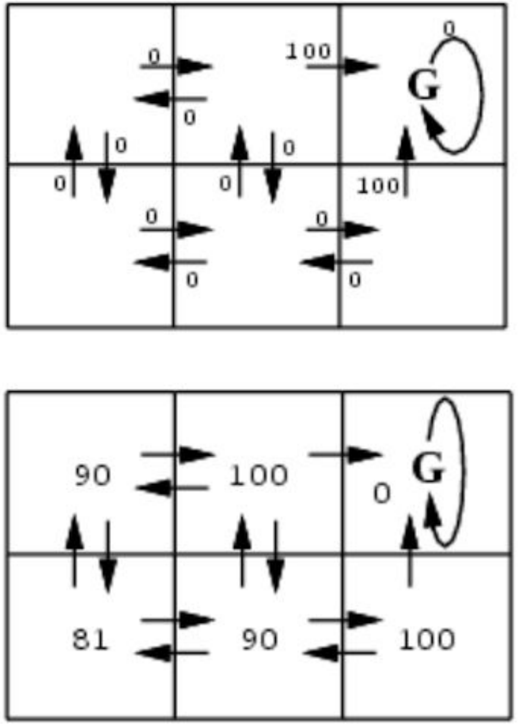




Reinforcement Learning

- Different from ML pbs so far:
  - Our decisions influence the next example we see. Decisions we make will be about actions to take (e.g., a robot deciding which way to move next), which will influence what we see next.
  - Goal will be not just to predict (say, whether there is a door in front of us or not) but to decide what to do.
- Model: Markov Decision Processes.

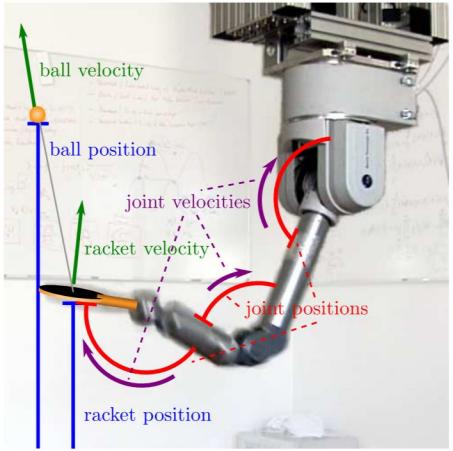
Reinforcement Learning



$$V^*(s) = E[r_t + \gamma\, r_{t+1} + \gamma^2 r_{t+2} + \ldots]$$

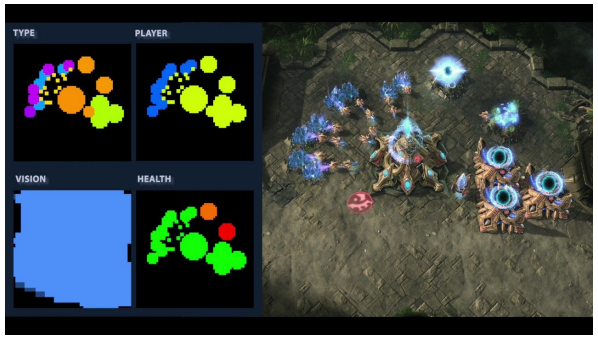
Tom Mitchell, April 2011

Applications



Robotics

Go



Video Games

Reinforcement learning - MDPs

- Set of states S
- Set of actions A
- Transition between states $P(s_{t+1} | s_t, a_t, s_{t-1}, a_{t-1}, \ldots)$
  - Markov assumption!
  - $P(s_{t+1} | s_t, a_t, s_{t-1}, a_{t-1}, \ldots) = P(s_{t+1} | s_t, a_t)$
- Rewards (also Markov assumption)
  - $P(r_t | s_t, a_t)$
- Objective, learn a policy Π: S -> A that maximizes
  - $E[r_t + \gamma r_{t+1}, + \gamma^2 r_{t+2} + \ldots], 0 < \gamma <= 1$

Why it's hard

- Sparse rewards
  - Most practical MDP problems only have rewards for some states
  - Can't just use a simple supervised technique to learn Π: S -> A for every state s
- Often care about sample efficiency
  - In real-world problem, only have limited number of experiences we can use to learn from
- The attribution problem
  - Agent executes a sequence of actions a_1, a_2, a_3, ... a_N, and get a bad reward R
  - What action led to the sequence failing?
- Still an active area of research
  - Lots of methods do poorly in practice

Ways to solve

- Model-based methods
  - Assume we have (or can learn) the transition function $P(s_{t+1} | s_t, a_t)$
  - Value iteration learning
    - Propagate the value of a state based on value of neighboring states
- Model-free methods
  - Don’t assume we know the transition function
  - Q-learning
    - Directly update values of state, action pairs
- Monte Carlo Methods
  - Lots and lots of methods
  - In a nutshell, at each state s_t, estimate the average discounted reward we get all of the times we have been in that state before (roll-out)

Questions?

Answers:

-2: 1/9, -1: 0, 1: 4/9, 3: 4/9. Not a good init method

Circle 3rd image.