Artificial Intelligence Programming

Introduction to Machine Learning

Chris Brooks

Department of Computer Science
University of San Francisco

21-2: Introduction

- We've talked about learning previously in the contact of specific algorithms.
- Purpose: discuss learning more generally.
- Give a flavor of other approaches to learning
- Talk more carefully about how to evaluate the performance of a learning algorithm.
  - This will come in handy for project 3.

21-3: Defining Learning

- So far, we've defined a learning agent as one that can improve its performance over time.
- We've seen two learning algorithms:
  - Decision tree
  - Bayesian Learning
- Let's define the problem a bit more precisely.

21-4: Defining Learning

- A program is said to learn from experiences E with respect to a set of tasks T and a performance measure P if its performance on T, as measured by P, improves with experience E.
- This means that, for a well-formulated learning problem, we need:
  - A set of tasks the agent must perform
  - A way to measure its performance
  - A way to quantify the experience the agent receives

21-5: Examples

- Speech recognition
  - Task: successfully recognize spoken words
  - Performance measure: fraction of words correctly recognized
  - Experience: A database of labeled, spoken words
- Learning to drive a car
  - Task: Drive on a public road using vision sensors
  - Performance: average distance driven without error
  - Experience: sequence of images and reactions from a human driver.
- Learning to play backgammon
  - Task: play backgammon
  - Performance measure: number of games won against humans of the appropriate caliber.
  - Experience: Playing games against itself.

21-6: Discussion

- Notice that not all performance measures are the same.
  - In some cases, we want to minimize all errors. In other cases, some sorts of errors can be more easily tolerated than others.
- Also, not all experience is the same.
  - Are examples labeled?
  - Does a learning agent immediately receive a reward after selecting an action?
  - How is experiential data represented? Symbolic? Continuous?
- Also: What is the final product?
  - Do we simply need an agent that performs correctly?
  - Or is it important that we understand why the agent performs correctly?

21-7: Other types of learning

- In this class, we'll focus on inductive supervised learning
  - Well-understood, mature, many applications.
- There are other types of learning
  - Deductive learning
  - Unsupervised learning
  - Reinforcement learning

21-8: Deductive Learning

- Recall that induction develops a general hypothesis from specific data.
- Deductive learning develops rules about specific situations from general principles.
  - "Knowledge-based" learning might be a better name - some induction may take place.
- For example, a deductive learning agent might cache the solution to a previous search problem so that it doesn't need to re-solve the problem.
- It might even try to generalize some of the specifics of the solution to apply to other instances.
  - Soar uses this style of learning.
  - Case-based reasoning is another example of this style of learning.

21-9: Unsupervised Learning

- In **unsupervised learning**, there is no teacher who has presented the learner with labeled examples.
- Instead, all the learner has is data.
- Problem: find a hypothesis (or pattern) that explains the data.

21-10: Clustering

- One example of unsupervised learning is **clustering**
- Given a collection of data, group the data into k clusters, such that similar items are in the same cluster.
- Challenge: don't know the class definitions in advance.

21-11: Agglomerative Clustering of Text

- One place where this is often applied is in document processing.
- Given a collection of documents, organize them into clusters based on topic.
- No preset list of potential categories, or labeled documents.
- Algorithm:
- $D = \{d_1, d_2, ..., d_n\}$
- While $|D| > k$:
  - Find the documents d_i and d_j that are closest according to some similarity measure.
  - Remove them from D
  - Construct a new d' that is the "union" of d_i and d_j and add it to D
- Result: a set of categories emerges from a collection of documents.

21-12: Reinforcement Learning

- In some cases, an agent must learn through interaction with the environment.
- Agent selects and executes an action and receives a reward as a result.
- Learning problem: What is the best action to take in a given state?
- Issues - since learning is integrated with execution, we can't just explore every possibility.
- Approach (in a nutshell) - try different actions to see how they do.
- The more confidence we have in our estimate of action values, the more likely we are to take the best-looking action.

21-13: Q-learning

- We want to learn a *policy*
  - This is a function that maps states to actions.
- What we get from the environment are state: reward pairs.
- We'll use this to learn a $Q(s,a)$ function that estimates the reward for taking action a in state s.
- This is a form of **model-free** learning
  - We do no reasoning about how the world works - we just map states to rewards.
  - This means we can apply the same algorithm to a wide variety of environments.

21-14: Q-learning

- We keep a table that maps state-action pairs to Q values.
- Every time we are in state s and take an action a, we receive a reward r, and winds up in state s'.
- We update our table as follows:
  - $Q(s,a) += \alpha(r + \gamma max_{a'} Q(s',a') - Q(s,a))$
- In other words, we add in the reward for taking an action in this state, plus acting optimally from that point.
- α is the learning rate.

21-15: Q-learning

- Q-learning has a distinct difference from other learning algorithms we've seen:
- The agent can select actions and observe the rewards they get.
- This is called **active learning**
- Issue: the agent would also like to maximize performance
  - This means trying the action that currently looks best
  - But if the agent never tries "bad-looking" actions, it can't recover from mistakes.
- Intuition: Early on, Q is not very accurate, so we'll try non-optimal actions. Later on, as Q becomes better, we'll select optimal actions.

21-16: Boltzmann exploration

- One way to do this is using Boltzmann exploration.
- We take an action with probability:
- $P(a|s) = \frac{k^{Q(s,a)}}{\sum_j k^{Q(s,a_j)}}$
- Where k is a temperature parameter.
- This is the same formula we used in simulated annealing.
- We'll return to Q-learning after we discuss MDPs
  - They're closely related.

21-17: Return to supervised learning

- Let's step back and think about supervised learning.
- Given some labeled data, find a hypothesis that best explains this data.
- This can be done using symbolic or numeric data.

21-18: A symbolic example

- Consider (once again) our playTennis example.
- Suppose we have the following experience:
  - Sunny, overcast, high, weak : yes
  - Sunny, overcast, low, strong: yes
  - Rainy, overcast, normal, weak: no
- We need to select a hypothesis that explains all of these examples
  - H1: sunny : yes
  - H2: Sunny and Overcast: yes
  - H3: ¬Rainy, overcast, normal, weak : yes
- Which do we pick?