CS 188: Artificial Intelligence  
Spring 2011

Lecture 12: Probability  
3/2/2011

Pieter Abbeel – UC Berkeley  
Many slides adapted from Dan Klein.

Announcements

- P3 due on Monday (3/7) at 4:59pm
- W3 going out tonight
- Midterm Tuesday 3/15  5pm-8pm
- Closed notes, books, laptops. May use one-page (two-sided) cheat sheet of your own design (group design OK but not recommended).
- Monday 3/14 : no lecture at usual 5:30-7:00pm time
  - Midterm review? Practice midterm?

Today

- MDP’s and Reinforcement Learning
  - Generalization
  - One of the most important concepts in machine learning!
  - Policy search
- Next, we’ll start studying how to reason with probabilities
  - Diagnosis
  - Tracking objects
  - Speech recognition
  - Robot mapping
  - … lots more!
- Third part of course: machine learning

The Story So Far: MDPs and RL

Things we know how to do:

- We can solve small MDPs exactly, offline
- We can estimate values $V^\pi(s)$ directly for a fixed policy $\pi$.
- We can estimate $Q^*(s,a)$ for the optimal policy while executing an exploration policy

Techniques:

- Value and policy iteration
- Temporal difference learning
- Q-learning
- Exploratory action selection

Q-Learning

- In realistic situations, we cannot possibly learn about every single state!
  - Too many states to visit them all in training
  - Too many states to hold the q-tables in memory
- Instead, we want to generalize:
  - Learn about some small number of training states from experience
  - Generalize that experience to new, similar states
  - This is a fundamental idea in machine learning, and we’ll see it over and over again

Example: Pacman

- Let’s say we discover through experience that this state is bad:

  - In naïve q learning, we know nothing about this state or its q states:

  - Or even this one!
Feature-Based Representations

- Solution: describe a state using a vector of features
- Features are functions from states to real numbers (often 0/1) that capture important properties of the state
- Example features:
  - Distance to closest ghost
  - Distance to closest dot
  - Number of ghosts
  - $1/d$ (dot to dot)
  - Is Pacman in a tunnel? (0/1)
  - ... etc.
- Can also describe a q-state $(s, a)$ with features (e.g. action moves closer to food)

Linear Feature Functions

- Using a feature representation, we can write a q function (or value function) for any state using a few weights:
  $$V(s) = w_1 f_1(s) + w_2 f_2(s) + \ldots + w_n f_n(s)$$
  $$Q(s, a) = w_1 f_1(s, a) + w_2 f_2(s, a) + \ldots + w_n f_n(s, a)$$
- Advantage: our experience is summed up in a few powerful numbers
- Disadvantage: states may share features but be very different in value!

Function Approximation

- Q-learning with linear q-functions:
  $$Q(s, a) = w_1 f_1(s, a) + w_2 f_2(s, a) + \ldots + w_n f_n(s, a)$$
- intuitive interpretation:
  - Adjust weights of active features
  - E.g. if something unexpectedly bad happens, disprefer all states with that state’s features
- Formal justification: online least squares

Example: Q-Pacman

- $Q(s, a) = 4.0 f_{DOT}(s, a) - 1.0 f_{GST}(s, a)$
- $f_{DOT}(s, \text{NORTH}) = 0.5$
- $f_{GST}(s, \text{NORTH}) = 1.0$
- $Q(s, a) = +1$
- $R(s, a, s') = -500$
- Error = $-501$
  $$w_{DOT} = 4.0 + a[501] 0.5$$
  $$w_{GST} = -1.0 + a[501] 1.0$$
  $$Q(s, a) = 3.0 f_{DOT}(s, a) - 3.0 f_{GST}(s, a)$$

Linear regression

- Given examples $(x_i, y_i)_{i=1...n}$
- Predict $y_{n+1}$ given a new point $x_{n+1}$

- Prediction $\hat{y}_i = w_0 + w_1 x_i$
- Prediction $\hat{y}_i = w_0 + w_1 x_{i1} + w_2 x_{i2}$
**Ordinary Least Squares (OLS)**

\[ E(w) = \frac{1}{2} \sum_i \left( \sum_k f_k(x_i)w_k - y_i \right)^2 \]

\[ \frac{\partial E}{\partial w_m} = \sum_i \left( \sum_k f_k(x_i)w_k - y_i \right) f_m(x_i) \]

\[ E \leftarrow E + \alpha \sum_i \left( \sum_k f_k(x_i)w_k - y_i \right) f_m(x_i) \]

Value update explained:

\[ w_i \leftarrow w_i + \alpha \{ \text{error} \} f_i(s,a) \]

---

**Minimizing Error**

**Overfitting**

Degree 15 polynomial

**Policy Search**

- **Problem:** often the feature-based policies that work well aren’t the ones that approximate V / Q best
- **Solution:** learn the policy that maximizes rewards rather than the value that predicts rewards
- **This is the idea behind policy search, such as what controlled the upside-down helicopter**

---

**Policy Search**

- **Simplest policy search:**
  - Start with an initial linear value function or Q-function
  - Nudge each feature weight up and down and see if your policy is better than before

- **Problems:**
  - How do we tell the policy got better?
  - Need to run many sample episodes!
  - If there are a lot of features, this can be impractical
MDPs and RL Outline

- **Markov Decision Processes (MDPs)**
  - Formalism
  - Value iteration
  - Expectimax Search vs. Value Iteration
  - Policy Evaluation and Policy Iteration
- **Reinforcement Learning**
  - Model-based Learning
  - Model-free Learning
    - Direct Evaluation [performs policy evaluation]
    - Temporal Difference Learning [performs policy evaluation]
    - Q-Learning [learns optimal state-action value function Q*]
    - Policy Search [learns optimal policy from subset of all policies]

To Learn More About RL

- Online book: Sutton and Barto
- Graduate level courses at Berkeley with reading material/lecture notes online:
  [http://inst.eecs.berkeley.edu/~cs294-40/fa08/](http://inst.eecs.berkeley.edu/~cs294-40/fa08/)

Take a Deep Breath…

- We’re done with Part I Search and Planning!
- **Part II: Probabilistic Reasoning**
  - Diagnosis
  - Tracking objects
  - Speech recognition
  - Robot mapping
  - Genetics
  - Error correcting codes
  - … lots more!
- **Part III: Machine Learning**

Part II: Probabilistic Reasoning

- **Probability**
- Distributions over LARGE Numbers of Random Variables
  - Representation
  - Independence
  - Inference
    - Variable Elimination
    - Sampling
  - Hidden Markov Models

Probability

- **Probability**
  - Random Variables
  - Joint and Marginal Distributions
  - Conditional Distribution
  - Inference by Enumeration
  - Product Rule, Chain Rule, Bayes’ Rule
  - Independence
- You’ll need all this stuff A LOT for the next few weeks, so make sure you go over it now and know it inside out! The next few weeks we will learn how to make these work computationally efficiently for LARGE numbers of random variables.

Inference in Ghostbusters

- A ghost is in the grid somewhere
- Sensor readings tell how close a square is to the ghost
  - On the ghost: red
  - 1 or 2 away: orange
  - 3 or 4 away: yellow
  - 5+ away: green
- Sensors are noisy, but we know P(Color | Distance)

<table>
<thead>
<tr>
<th>Color</th>
<th>Distance</th>
</tr>
</thead>
<tbody>
<tr>
<td>red</td>
<td>0.05</td>
</tr>
<tr>
<td>orange</td>
<td>0.15</td>
</tr>
<tr>
<td>yellow</td>
<td>0.5</td>
</tr>
<tr>
<td>green</td>
<td>0.3</td>
</tr>
</tbody>
</table>
Uncertainty

- General situation:
  - Evidence: Agent knows certain things about the state of the world (e.g., sensor readings or symptoms)
  - Hidden variables: Agent needs to reason about other aspects (e.g., where an object is or what disease is present)
  - Model: Agent knows something about how the known variables relate to the unknown variables
- Probabilistic reasoning gives us a framework for managing our beliefs and knowledge

Random Variables

- A random variable is some aspect of the world about which we (may) have uncertainty
  - $R$: Is it raining?
  - $D$: How long will it take to drive to work?
  - $L$: Where am I?
- We denote random variables with capital letters
- Like variables in a CSP, random variables have domains
  - $R$: in {true, false} (sometimes write as $\{+r, -r\}$)
  - $D$: in $[0, \infty)$
  - $L$: in possible locations, maybe $\{(0,0), (0,1), \ldots\}$

Random Variables

- A random variable is some aspect of the world about which we (may) have uncertainty
  - $R$: Is it raining?
  - $D$: How long will it take to drive to work?
  - $L$: Where am I?
- We denote random variables with capital letters
- Like variables in a CSP, random variables have domains
  - $R$: in {true, false} (sometimes write as $\{+r, -r\}$)
  - $D$: in $[0, \infty)$
  - $L$: in possible locations, maybe $\{(0,0), (0,1), \ldots\}$

Probability Distributions

- Unobserved random variables have distributions
  - $P(T)$
  - $P(W)$

<table>
<thead>
<tr>
<th></th>
<th>P</th>
<th>W</th>
<th>P</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>0.5</td>
<td>rain</td>
<td>0.1</td>
</tr>
<tr>
<td>T</td>
<td>0.5</td>
<td>fog</td>
<td>0.3</td>
</tr>
<tr>
<td>T</td>
<td>0.5</td>
<td>meteor</td>
<td>0.0</td>
</tr>
</tbody>
</table>

- A distribution is a TABLE of probabilities of values
- A probability (lower case value) is a single number
  - $P(\text{rain}) = 0.1$
- Must have: $\forall x P(x) \geq 0$, $\sum x P(x) = 1$

Joint Distributions

- A joint distribution over a set of random variables: $X_1, X_2, \ldots, X_n$
  - $P(x_1, x_2, \ldots, x_n)$ specifies a real number for each assignment (or outcome)

<table>
<thead>
<tr>
<th></th>
<th>T</th>
<th>W</th>
<th>P</th>
</tr>
</thead>
<tbody>
<tr>
<td>hot</td>
<td>sun</td>
<td>0.4</td>
<td></td>
</tr>
<tr>
<td>hot</td>
<td>rain</td>
<td>0.1</td>
<td></td>
</tr>
<tr>
<td>cold</td>
<td>sun</td>
<td>0.2</td>
<td></td>
</tr>
<tr>
<td>cold</td>
<td>rain</td>
<td>0.3</td>
<td></td>
</tr>
</tbody>
</table>

- Size of distribution if $n$ variables with domain sizes $d$?
- Must obey: $P(x_1, x_2, \ldots, x_n) \geq 0$
- $\sum_{(x_1, x_2, \ldots, x_n)} P(x_1, x_2, \ldots, x_n) = 1$
- For all but the smallest distributions, impractical to write out

Events

- An event is a set $E$ of outcomes
  - $P(E) = \sum_{(x_1, x_2, \ldots, x_n) \in E} P(x_1, x_2, \ldots, x_n)$
- From a joint distribution, we can calculate the probability of any event
  - Probability that it’s hot AND sunny?
  - Probability that it’s hot?
  - Probability that it’s hot OR sunny?
- Typically, the events we care about are partial assignments, like $P(T=\text{hot})$
Marginal Distributions

- Marginal distributions are sub-tables which eliminate variables
- Marginalization (summing out): Combine collapsed rows by adding

\[
P(T, W) = \sum_{t} P(t, s)
\]

\[
P(T) = \sum_{t} P(t)
\]

\[
P(W) = \sum_{t} P(s)
\]

\[
P(X_1 = x_1) = \sum_{x_2} P(X_1 = x_1, X_2 = x_2)
\]

Conditional Probabilities

- A simple relation between joint and conditional probabilities
- In fact, this is taken as the definition of a conditional probability

\[
P(a | b) = \frac{P(a, b)}{P(b)}
\]

Normalizing Trick

- A trick to get a whole conditional distribution at once:
  - Select the joint probabilities matching the evidence
  - Normalize the selection (make it sum to one)

\[
P(T, r)
\]

\[
P(T' | r')
\]

Probabilistic Inference

- Probabilistic inference: compute a desired probability from other known probabilities (e.g., conditional from joint)
- We generally compute conditional probabilities
  - \(P(\text{on time} | \text{no reported accidents}) = 0.90\)
  - These represent the agent’s beliefs given the evidence
- Probabilities change with new evidence:
  - \(P(\text{on time} | \text{no accidents, 5 a.m.}) = 0.95\)
  - \(P(\text{on time} | \text{no accidents, 5 a.m., raining}) = 0.80\)
  - Observing new evidence causes beliefs to be updated

Inference by Enumeration

- \(P(\text{sun})\)?
- \(P(\text{sun} | \text{winter})?\)
- \(P(\text{sun} | \text{winter, warm})?\)
Inference by Enumeration

- General case:
  - Evidence variables: $E_1 \ldots E_k = e_1 \ldots e_k$
  - Query* variable:
  - Hidden variables: $H_1 \ldots H_l$

- We want: $P(Q|e_1 \ldots e_k)$

  - First, select the entries consistent with the evidence
  - Second, sum out $H$ to get joint of Query and evidence:
    $$P(Q, e_1 \ldots e_k) = \sum_{h_1 \ldots h_l} P(Q, e_1 \ldots e_k, h_1 \ldots h_l)$$

  - Finally, normalize the remaining entries to conditionize

- Obvious problems:
  - Worst-case time complexity $O(d^n)$
  - Space complexity $O(d^n)$ to store the joint distribution

The Product Rule

- Sometimes have conditional distributions but want the joint
  $$P(x|y) = \frac{P(x,y)}{P(y)} \quad \iff \quad P(x,y) = P(x|y)P(y)$$

- Example:

<table>
<thead>
<tr>
<th>$P(W)$</th>
<th>D</th>
<th>W</th>
<th>P</th>
</tr>
</thead>
<tbody>
<tr>
<td>sun</td>
<td>0.2</td>
<td>wet</td>
<td>0.1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>dry</td>
<td>0.9</td>
</tr>
</tbody>
</table>

| $P(D|W)$ | D | W | P  |
|---------|---|---|----|
| dry     | 0.7 | wet | 0.1 |
|         |     | dry | 0.72 |

The Chain Rule

- More generally, can always write any joint distribution as an incremental product of conditional distributions

$$P(x_1, x_2, x_3) = P(x_1)P(x_2|x_1)P(x_3|x_1, x_2)$$

$$P(x_1, x_2, \ldots, x_n) = \prod_i P(x_i|x_1 \ldots x_{i-1})$$

- Why is this always true?

Bayes’ Rule

- Two ways to factor a joint distribution over two variables:
  $$P(x, y) = P(y|x)P(x)$$

- Dividing, we get:
  $$P(x|y) = \frac{P(y|x)P(x)}{P(y)}$$

- Why is this at all helpful?
  - Lets us build one conditional from its reverse
  - Often one conditional is tricky but the other one is simple
  - Foundation of many systems we’ll see later (e.g. ASR, MT)

- In the running for most important AI equation!

Inference with Bayes’ Rule

- Example: Diagnostic probability from causal probability:
  $$P(Cause|Effect) = \frac{P(Effect|Cause)P(Cause)}{P(Effect)}$$

- Example:
  - $m$ is meningitis, $s$ is stiff neck
  - $P(s|m) = 0.8$
  - $P(m) = 0.0001$
  - $P(s) = 0.1$

- $P(m|s) = \frac{P(s|m)P(m)}{P(s)} = \frac{0.8 \times 0.0001}{0.1} = 0.0008$

- Note: posterior probability of meningitis still very small
- Note: you should still get stiff necks checked out! Why?

Ghostbusters, Revisited

- Let’s say we have two distributions:
  - Prior distribution over ghost location: $P(G)$
    - Let’s say this is uniform
  - Sensor reading model: $P(R|G)$
    - Given: we know what our sensors do
    - $R =$ reading color measured at $(1,1)$
    - E.g. $P(R = yellow | G=(1,1)) = 0.1$

- We can calculate the posterior distribution $P(G|r)$ over ghost locations given a reading using Bayes’ rule:
  $$P(g|r) \propto P(r|g)P(g)$$
Independence

- Two variables are independent in a joint distribution if:
  \[ P(X, Y) = P(X)P(Y) \quad \text{for all } x, y \]

  \[ P(x, y) = P(x)P(y) \]

  - Says the joint distribution factors into a product of two simple ones
  - Usually variables aren’t independent!

- Can use independence as a modeling assumption
  - Independence can be a simplifying assumption
  - Empirical joint distributions: at best “close” to independent
  - What could we assume for (Weather, Traffic, Cavity)?

- Independence is like something from CSPs: what?

Example: Independence?

<table>
<thead>
<tr>
<th>P(T)</th>
<th>P(W)</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>P(T)</td>
</tr>
<tr>
<td>warm</td>
<td>0.6</td>
</tr>
<tr>
<td>cold</td>
<td>0.4</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>P1(T, W)</th>
<th>P2(T, W)</th>
</tr>
</thead>
<tbody>
<tr>
<td>warm</td>
<td>0.3</td>
</tr>
<tr>
<td>cold</td>
<td>0.2</td>
</tr>
<tr>
<td>sun</td>
<td>0.6</td>
</tr>
<tr>
<td>rain</td>
<td>0.4</td>
</tr>
</tbody>
</table>

Example: Independence

- N fair, independent coin flips:
  \[
P(X_1) = \begin{bmatrix} H & 0.5 \\ T & 0.5 \end{bmatrix} \quad \ldots \quad P(X_n) = \begin{bmatrix} H & 0.5 \\ T & 0.5 \end{bmatrix}
\]

  \[
P(X_1, X_2, \ldots X_n) = \begin{cases} 2^n & \text{if } X_1 = \ldots = X_n \\ 0 & \text{otherwise} \end{cases}
\]