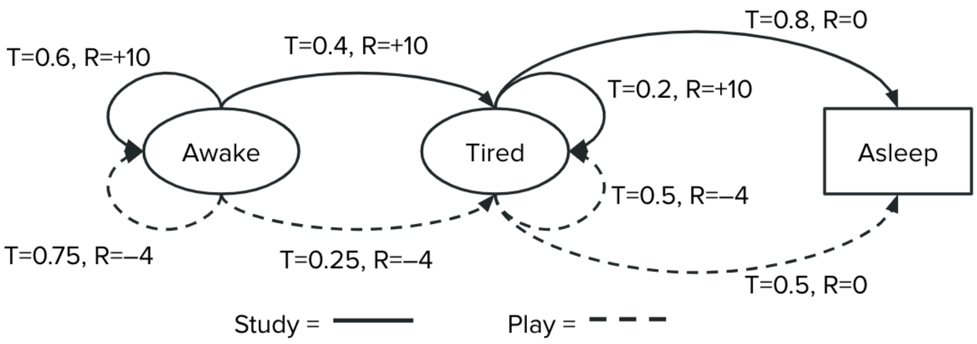
Introduction to Artificial Intelligence **Homework 6**
COMP 3703 November 11, 2019

1. **Markov Decision Processes**

The Markov Decision Process below models three states – Awake, Tired, and Asleep – and two actions – Study (the solid arrows) and Play (the dashed arrows). The transition value (i.e. probability) and reward for each (s, a, s') triple is provided on the arc of each arrow. The Asleep state is a terminal state with no actions. (Alternatively, you could think of the Asleep state as having a single non-action arc looping back to Asleep with probability 1.0 and reward 0.)



Perform value iteration on this MDP up to $V_2(s)$ for states Awake and Tired (Sleep will always have value 0). Remember that $V_0(s)$ is initialized to be 0 and the update function is given by:

$$V_{k+1}(s) = \max_a \sum_{s'} T(s,a,s')[R(s,a,s') + \gamma V_k(s')]$$

Update: The last term is $V_k(s')$, not $V_k(s)$. Additionally, a few of my calculations were incorrect but have now been corrected.

To keep things simple, assume $\gamma = 1$.

Solution:

$V_0(Awake) = V_0(Tired) = 0$
$V_1(Awake) = 10$

Study: $(Awake, Study, Awake) + (Awake, Study, Tired)$
$= 0.6[10 + 1 * 0] + 0.4[10 + 1 * 0] = 6 + 4 = 10$
Play: $(Awake, Play, Awake) + (Awake, Play, Tired)$
$= 0.75[-4 + 1 * 0] + 0.25[-4 + 1 * 0] = -3 - 1 = -4$
$V_1(Awake) = max(10, -4) = 10$

$V_1(Tired) = 2$

Study: $(Tired, Study, Tired) + (Tired, Study, Asleep)$
$= 0.2[10 + 1 * 0] + 0.8[0 + 1 * 0] = 2 + 0 = 2$
Play: $(Tired, Play, Tired) + (Tired, Play, Asleep)$
$= 0.5[-4 + 1 * 0] + 0.5[0 + 1 * 0] = -2 + 0 = -2$
$V_1(Tired) = max(2, -2) = 2$

$V_2(Awake) = 16.8$

Study: $(Awake, Study, Awake) + (Awake, Study, Tired)$
$= 0.6[10 + 1 * 10] + 0.4[10 + 1 * 2] = 12 + 4.8 = 16.8$
Play: $(Awake, Play, Awake) + (Awake, Play, Tired)$
$= 0.75[-4 + 1 * 10] + 0.25[-4 + 1 * 2] = 4.5 - 0.5 = 4$
$V_1(Awake) = max(16.8, 4) = 16.8$

$V_2(Tired) = 2.4$

Study: $(Tired, Study, Tired) + (Tired, Study, Asleep)$
$= 0.2[10 + 1 * 2] + 0.8[0 + 1 * 0] = 2.4 + 0 = 2.4$
Play: $(Tired, Play, Tired) + (Tired, Play, Asleep)$
$= 0.5[-4 + 1 * 2] + 0.5[0 + 1 * 0] = -1 + 0 = -1$
$V_1(Tired) = max(2.4, -1) = 2.4$

2. **Reinforcement Learning**

Consider the following gridworld with states $A, B, G1$, and $G2$.

Rewards

+10	+1

State names

A	B
G1	G2

From state A, possible actions are Right (→) and Down (↓). From state B, possible actions are Left (←) and Down (↓). For states $G1$ and $G2$, the only possible action is Exit. Upon exiting, we receive a reward (+10 and +1 respectively) and end at the end-of-game absorbing state X (which always has value $V(X) = 0$). The discount $\gamma = 1$ and there is no noise (i.e. all actions are deterministic).

Consider the following episodes. The following questions will use various sequences of these episodes as examples.

Episode 1 ($E1$)

s	a	s'	r
A	↓	$G1$	0
$G1$	exit	X	10

Episode 2 ($E2$)

s	a	s'	r
B	↓	$G2$	0
$G2$	exit	X	1

Episode 3 ($E3$)

s	a	s'	r
A	→	B	0
B	↓	$G2$	0
$G2$	exit	X	1

Episode 4 ($E4$)

s	a	s'	r
B	←	A	0
A	↓	$G1$	0
$G1$	exit	X	10

Note: The terminal state X is not visualized above. You can consider X to be the state that $G1$ and $G2$ transition to when the action *exit* is taken. X always has value 0 since no actions can be taken from X.

(a) Suppose the observed sequence is $\langle E1, E2, E3, E4 \rangle$. Determine the value $V(s)$ of each state using **direct evaluation**. Update: Your calculations are always based on the episodes resulting from the actions taken, which serve as your sample set for estimating the MDP model. The fixed policy I originally provided here did not make sense for this question.

Solution

$V(A) = ((0 + 10) + (0 + 0 + 1) + (0 + 10))/3 = (10 + 1 + 10)/3 = 7$
$V(B) = ((0 + 1) + (0 + 1) + (0 + 0 + 10))/3 = (1 + 1 + 10)/3 = 4$

(b) Suppose the observed sequence is $\langle E3, E4, E1, E2 \rangle$. Determine the value $V(s)$ of each state using **temporal difference learning**. All values are initialized to zero and the learning rate is $\alpha = 0.5$. For reference, here is the update for each sample:

$$\text{sample} = R(s, \pi(s), s') + \gamma V^\pi(s')$$
$$V^\pi(s) = (1-\alpha)V^\pi(s) + (\alpha)\text{sample}$$

Solution

$(A \rightarrow B\ 0)$ and $(B \downarrow G2\ 0)$ do not change $V(A)$ and $V(B)$ (they remain 0).
$(G2\ exit\ X\ 1)$: sample $= 1 + (1)0 = 1 \Rightarrow V^\pi(G2) = 0.5(0) + 0.5(1) = 0.5$
$(B \leftarrow A\ 0)$ and $(A \downarrow G1\ 0)$ do not change $V(A)$ and $V(B)$ (they remain 0).
$(G1\ exit\ X\ 10)$: sample $= 10 + (1)0 = 10 \Rightarrow V^\pi(G1) = 0.5(0) + 0.5(10) = 5$ Update: Slight miscalculation here; has been fixed.
$(A \downarrow G1\ 0)$: sample $= 0 + (1)5 = 5 \Rightarrow V^\pi(A) = 0.5(0) + 0.5(5) = 2.5$
$(G1\ exit\ X\ 10)$: sample $= 10 + (1)0 = 10 \Rightarrow V^\pi(G1) = 0.5(5) + 0.5(10) = 7.5$
$(B \downarrow G2\ 0)$: sample $= 0 + (1)0.5 = 0.5 \Rightarrow V^\pi(B) = 0.5(0) + 0.5(0.5) = 0.25$
$(G1\ exit\ X\ 10)$: sample $= 1 + (1)0 = 1 \Rightarrow V^\pi(B) = 0.5(0.5) + 0.5(1) = 0.75$

The last values for each state were:
$V^\pi(A) = 2.5$, $V^\pi(B) = 0.25$, $V^\pi(G1) = 7.5$, $V^\pi(G2) = 0.75$

(c) Consider using **Q-Learning** to learn the q-values of this gridworld. For which of the following sequences, if repeated an infinite number of times, would the q-values for *all* state-action pairs (s, a) converge to their optimal value $Q^*(s, a)$?

i. $\langle E1, E2, E1, E2, ... \rangle$
ii. $\langle E3, E4, E3, E4, ... \rangle$
iii. $\langle E1, E2, E3, E4, ... \rangle$

Solution

(i) will not converge since we never explore $(A, \rightarrow)$ and $(B, \leftarrow)$. (ii) and (iii) explore all possible (s, a) pairs, so the q-values will converge eventually.

3. Naive Bayes

Consider the following labeled corpus of text messages, with punctuation and capitalization removed for simplicity:

(Spam) `you have a chance to win $100`
(Spam) `send your love with an exclusive offer`
(Spam) `you have an offer for free tickets`
(Ham) `you up`
(Ham) `i am here`
(Ham) `have you seen crazy stupid love`
(Ham) `yo terron canceled the quiz you wanna get drinks`

For the following problems, you need only represent probabilities with fractions, not computed decimal values.

(a) Compute the following probabilities:

Prior probability of spam: $P(Y = spam)$
Prior probability of ham: $P(Y = ham)$
Probability of the word `you` given spam: $P(W = you|Y = spam)$
Probability of the word `you` given ham: $P(W = you|Y = ham)$

Solution

The first two come from the number of spam and ham texts: $P(Y = spam) = 3/7$, $P(Y = ham) = 4/7$

For the conditionals, we need the total number of words in spam texts (21) and the total number of words in ham texts (20). Then we count the number of times `you` shows up in spam and ham texts: $P(W = you|Y = spam) = 2/21$, $P(W = you|Y = ham) = 3/20$

(b) What is the probability of spam and the probability of ham given the following text: `love you`. That is, what are $P(Y = spam|X = \texttt{love you})$ and $P(Y = ham|X = \texttt{love you})$? You answer should be written as a product of probabilities (i.e. you do not have to compute the exact value of the product).

Note: Bayes' Rule does specify a denominator to divide this product by, but since we only care about relative probabilities, it is not necessary to include for classification tasks.

Solution

Spam: $P(Y = spam) * P(W = love|Y = spam) * P(W = you|Y = spam) = (3/7)(1/21)(2/21)$

Ham: $P(Y = ham) * P(W = love|Y = ham) * P(W = you|Y = ham) = (4/7)(1/20)(3/20)$

(c) Consider the text: `are you crazy`. What are the values of $P(Y = spam|X = \texttt{are you crazy})$ and $P(Y = ham|X = \texttt{are you crazy})$? (Hint: You shouldn't need to write out the entire product to determine this.) Why is this the case and how can we account for the issue observed here?

Solution

Both probabilities would be 0 because `are` is an unseen event (and `crazy` is unseen in spam texts). This can be mitigated by using Laplace smoothing. For example, add-one smoothing adds 1 to all word counts so that probabilities of words we've seen are hardly affected while unseen words receive small, non-zero probabilities.

4. **Multiclass Perceptron**

Suppose for some multiclass classification problem (e.g. genres of music, categories for news articles, etc.) we have three labels, simplified here to the labels 1, 2, and 3. As such, we have three weight vectors, one per label. There are three features, so each vector contains three values. At some iteration during the update process, the weight vectors have the following values:

$$w_1 = [1, 2, -2], w_2 = [3, -2, -1], w_3 = [-1, 2, 4]$$

(a) On the next iteration, we sample a random data point x_i with label $y = 1$ and extract the feature vector $f(x_i) = [2, 2.5, -1]$. Perform the multiclass perceptron update for x_i and compute the values of any updated weight vectors.

Solution

$w_1 \cdot f(x_i) = 2(1) + 2.5(2) + (-1)(-2) = 9$
$w_2 \cdot f(x_i) = 2(3) + 2.5(-2) + (-1)(-1) = 2$
$w_3 \cdot f(x_i) = 2(-1) + 2.5(2) + (-1)(4) = -1$

The weight vector with the highest activation is w_1, so the perceptron would classify x_i as $y = 1$. This is correct, so the weight vectors remain unchanged.

(b) After performing the update in part (a), we sample another point x_j with label $y = 2$ and feature vector $f(x_j) = [1, -0.5, 3]$. Perform the multiclass perceptron update for x_j and compute the values of any updated weight vectors.

Solution

$w_1 \cdot f(x_j) = 1(1) + (-0.5)(2) + 3(-2) = -6$
$w_2 \cdot f(x_j) = 1(3) + (-0.5)(-2) + 3(-1) = 1$
$w_3 \cdot f(x_j) = 1(-1) + (-0.5)(2) + 3(4) = 10$

The weight vector with the highest activation is w_3, so the perceptron would classify x_j as $y = 3$. This is incorrect since the correct label is $y = 2$. So, we must update w_3 (the weight vector associated with the label we chose) by subtracting $f(x_j)$. Additionaly, we update w_2 (the weight vector associated with the correct label) by adding $f(x_j)$.
w_3 becomes $w_3 - f(x_j) = [-1, 2, 4] - [1, -0.5, 3] = [-2, 2.5, 1]$
w_2 becomes $w_2 + f(x_j) = [3, -2, -1] + [1, -0.5, 3] = [4, -1.5, 2]$

5. **Bias**

It is easy to think that machine learning algorithms are objectively correct because they are based on data. However, while the algorithm may optimally separate, cluster, or otherwise detect patterns in data, a key thing to keep in mind is how biased your data can be. As an extreme example, if a classifier is trained on a data set that only contains pictures of dogs, it will not be able to recognize pictures of cats.

For each of the following data sets, state how the data set may be biased for the task at hand:

(a) Data: A hand-selected set of student essays for an English class
Task: Autograding English essays

Solution

This is an example of **selection bias**. Possible biases could be:

- Racial bias against student names like Raj or Jamal, for example
- An assumption that the best essays can only come from non-international students
- Only including essays that follow a certain format
- Only including essays that exhibit a certain opinion

(b) Data: Collection of hand-drawn sketches by people who were told to draw a “shoe”
Task: Classifying a sketch as being a shoe or not

Solution

This is an example of **interaction bias**. Possible biases could be:

- There are many types of shoes, and not all could be represented here
- Depending on the demographic of users, not everyone may think of shoes that are prevalent in foreign countries

(c) Data: Former presidents of the United States
Task: Predicting who will be president next

Solution

This is an example of **latent bias**. Possible biases could be:

- Racial bias since all former presidents except one were white
- Gender bias since no president (yet!) has been female
- Not so much the case for the United States, but there could be political bias if many presidents represented one particular party