Apprendimento Automatico
Esame del 21/02/2008 – A.A. 2007/2008
Prof. Roberto Navigli

1 Programmazione Genetica

Esercizio 1.1. Un labirinto viene rappresentato mediante una matrice $M = \{m_{ij}\}$ di dimensione arbitraria tale che m_{ij} vale 0 se la cella è percorribile, 1 se non è percorribile perché vi è presente un muro. Un programma consiste in una sequenza di azioni che portano l'agente da un punto di partenza a un punto di arrivo (entrambi i punti sono costanti). Le azioni possibili sono le seguenti: $\{\rightarrow, \leftarrow, \downarrow, \uparrow\}$. Un programma è valido se consente di raggiungere un punto di arrivo da un punto di partenza passando solo per celle percorribili.

Ad esempio, dato il punto di partenza $(1, 1)$ e il punto di arrivo $(3, 3)$, e dato il seguente labirinto:

	1	**2**	**3**	**4**
1	0	0	0	0
2	0	1	0	1
3	0	0	0	1

i seguenti sono programmi validi: $\rightarrow\rightarrow\downarrow\downarrow$, $\downarrow\downarrow\rightarrow\rightarrow$, $\rightarrow\rightarrow\rightarrow\leftarrow\downarrow\downarrow$, ecc. Al contrario, non sono validi quei programmi che portano l'agente a passare attraverso un muro, quali $\rightarrow\downarrow$, $\rightarrow\rightarrow\rightarrow\downarrow\downarrow\leftarrow$, ecc. oppure programmi che non conducono a destinazione, quali $\rightarrow$, $\rightarrow\rightarrow\rightarrow$, e così via.

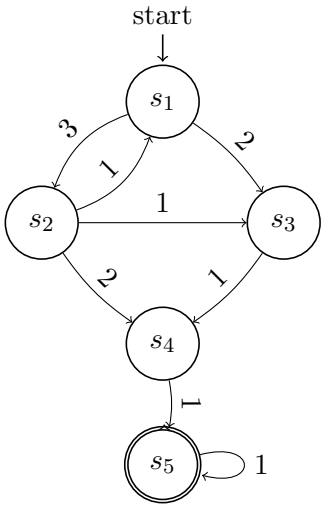
Si vogliono apprendere mediante programmazione genetica uno o più programmi validi che minimizzino il numero di azioni da effettuare per giungere al punto di arrivo.

a) Definire una funzione di fitness ragionevole per l'apprendimento di tali programmi **(3 punti)**;

b) Fornire una strategia ragionevole per la mutazione di un programma **(2 punti)**.

2 Apprendimento con Rinforzo

Esercizio 2.1 (6 punti). Sia dato il seguente ambiente:



Applicare il primo passo dell'algoritmo di Q-learning per calcolare la prima approssimazione delle stime $\hat{Q}$ per $\gamma = 0.8$. Il passo consiste nella visita degli stati: s_1, s_2, s_4, s_5. Le azioni sono rappresentate dagli archi del grafo. Su ciascun arco viene specificata la ricompensa relativa all'azione corrispondente.

Esercizio 2.2 (3 punti). Si considerino due politiche π_1 e π_2 tali che π_1 ha prestazioni migliori di π_2 se l'agente inizia da un certo stato s_1, ma π_2 ha prestazioni migliori di π_1 se esso inizia da un certo altro stato s_2. In altre parole, si ha: $V^{\pi_1}(s_1) > V^{\pi_2}(s_1)$, ma $V^{\pi_1}(s_2) < V^{\pi_2}(s_2)$. Spiegare perché esiste sempre un'unica politica ottima π^* che massimizza $V^{\pi^*}(s)$ per ogni stato iniziale s. In altre parole, spiegare perché nell'apprendimento con rinforzo è sempre possibile ottenere una politica ottima π^* tale che $\forall \pi, s, V^{\pi^*}(s) \geq V^{\pi}(s)$.

3 Alberi di Decisione

Esercizio 3.1 (4 punti). Dato il seguente insieme di addestramento, apprendere un albero di decisione mediante l'algoritmo ID3:

a	**b**	**classe**
T	T	$+$
T	T	$+$
T	F	$-$
F	F	$+$
F	T	$-$
F	T	$-$
F	T	$+$

Esercizio 3.2 (3 punti). Come viene classificata la nuova istanza (F, T) dall'albero di decisione dell'esercizio precedente? Qual è il supporto per tale decisione?

4 Reti Neurali

Esercizio 4.1 (4 punti). Ottenere i pesi di un percettrone che calcoli la funzione logica *OR*.

Quesito 4.2 (2 punti). Esistono più soluzioni per l'esercizio 4.1? Motivare la risposta.

Quesito 4.3 (3 punti). E' possibile rappresentare la funzione logica *XOR* mediante un percettrone? Motivare la risposta.

5 Spazio delle Versioni

***Esercizio 5.1 (3 punti).** *Questo esercizio può essere svolto in sostituzione di un altro esercizio da 3 punti.*

Enumerare le istanze presenti nello spazio delle versioni, dati gli insiemi di confine S e G seguenti:

$$S = \{(a_1, b_1, ?, d_1)\} \text{ e } G = \{(a_1, ?, ?, ?), (?, b_1, ?, ?)\}.$$

6 Soluzioni

Esercizio 1.1.

a) I fattori da tenere in considerazione sono la lunghezza del programma e la validità dello stesso. Definiamo una funzione $len(h)$ che restituisce il numero di azioni del programma h e $valido(h) = 1$ se e solo se h è un programma valido (0 altrimenti). Possiamo quindi definire la funzione di fitness come segue:

$$Fitness(h) = \frac{1}{len(h)} \cdot valido(h)$$

Volendo definire una funzione di fitness che consideri migliori quei programmi non validi che si avvicinano di più al punto di arrivo, si può definire una funzione $dist(a, h)$ che calcola la distanza (in termini di numero di celle) tra il punto di arrivo a e il punto di arrivo del programma h (0, se coincidono). In tal caso, la funzione di fitness può essere definita come segue:

$$Fitness(h) = \frac{1}{len(h) \cdot (dist(a, h) + 1)}$$

o, alternativamente:

$$Fitness(h) = \frac{1}{len(h) + dist(a, h)}$$

b) Mutare una singola azione di un programma h è una strategia naive se il programma non è valido. Si può quindi definire la seguente strategia: se il programma è valido (ovvero, raggiunge il punto di arrivo), si muta un'azione a caso, altrimenti si muta un'azione che porta su una cella non percorribile.

Esercizio 2.1. I valori calcolati durante la prima iterazione sono i seguenti:
$Q(s_1, s_1 \rightarrow s_2) = 3 + \gamma \cdot 0 = 3$
$Q(s_2, s_2 \downarrow s_4) = 2 + \gamma \cdot 0 = 2$
$Q(s_4, s_4 \downarrow s_5) = 1 + \gamma \cdot 0 = 1$

Procedendo a ritroso:
$Q(s_2, s_2 \rightarrow s_4) = 2 + \gamma \cdot 1 = 2.8$
$Q(s_1, s_1 \rightarrow s_2) = 3 + \gamma \cdot 2.8 = 5.24$

Esercizio 2.2. Esiste un'unica politica ottima π^* poiché essa può essere definita a partire da politiche ottime sui singoli stati. In particolare, π^* si ottiene come segue:

$$\pi^*(s_1) = \pi_1(s_1)$$
$$\pi^*(s_2) = \pi_2(s_2)$$

Esercizio 3.1. Calcoliamo prima l'entropia della collezione D:

$$H(D) = -\frac{3}{7}log_2\frac{3}{7} - \frac{4}{7}log_2\frac{4}{7} = 0.98$$

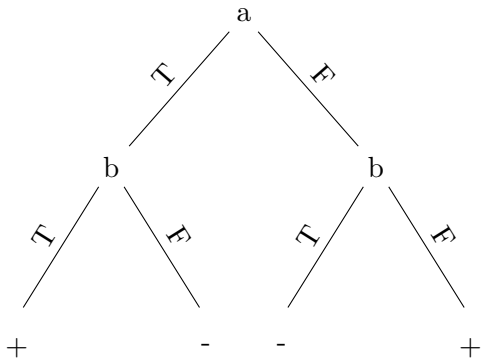
Quindi, determiniamo l'entropia dei sottoinsiemi di D, uno per ogni scelta di valore per l'attributo a:

$$H(D_{a=T}) = -\frac{2}{3}log_2\frac{2}{3} - \frac{1}{3}log_2\frac{1}{3} = 0.91$$
$$H(D_{a=F}) = -\frac{2}{4}log_2\frac{2}{4} - \frac{2}{4}log_2\frac{2}{4} = 1$$
$$Gain(D,a) = H(D) - \sum_{v\in\{T,F\}} \frac{|D_{a=v}|}{|D|} H(D_{a=v}) = 0.98 - \frac{3}{7}\cdot 0.91 - \frac{4}{7}\cdot 1 = 0.018$$

Per l'attributo b:

$$H(D_{b=T}) = -\frac{2}{5}log_2\frac{2}{5} - \frac{3}{5}log_2\frac{3}{5} = 0.97$$
$$H(D_{b=F}) = -\frac{1}{2}log_2\frac{1}{2} - \frac{1}{2}log_2\frac{1}{2} = 1$$
$$Gain(D,b) = 0.98 - \frac{5}{7}\cdot 0.97 - \frac{2}{7}\cdot 1 = 0.0014$$

L'attributo con guadagno informativo maggiore è a, quindi esso sarà scelto come nodo radice dell'albero di decisione. In entrambi i casi ($a = T$, $a = F$), il secondo attributo sarà b. L'albero di decisione ottenuto è il seguente:



Esercizio 3.2. La classificazione dell'esempio (F, T) è negativa (-). Si noti che nella scelta della classe del ramo $a = F, b = T$ si è scelta la classe $-$ poichè il supporto include due esempi negativi e uno positivo (ovvero le ultime tre istanze della collezione D).

Esercizio 4.1. La tabella di verità dell'OR è la seguente:

x_1	x_2	$OR(x_1, x_2)$
0	0	0
0	1	1
1	0	1
1	1	1

per cui dobbiamo trovare un vettore di pesi $\vec{w} = (w_0, w_1, w_2)$ tali che:

$$w_0 + w_1 \cdot x_1 + w_2 \cdot x_2 > 0$$

nei casi in cui $\vec{x} = (0,1), (1,0), (1,1)$. Una possibile soluzione è $\vec{w} = (-0.1, 1, 1)$. Infatti: $-0.1 + x_1 + x_2 > 0$ tranne quando $x_1 = x_2 = 0$ (unico caso in cui l'OR vale 0).

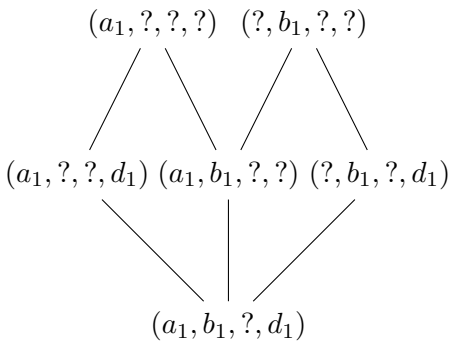
Esercizio 4.2. Esistono infinite soluzioni al problema. E' sufficiente scegliere $\vec{w} = (-b, a, a)$, con $a, b > 0$ e $a > b$.

Esercizio 4.3. Non è possibile rappresentare la funzione XOR con un singolo percettrone, perché la funzione non è linearmente separabile (basta disegnare i punti sul piano bidimensionale per convincersene).

Esercizio 5.1. Lo spazio delle versioni è:

$$\{(a_1, ?, ?, ?), (?, b_1, ?, ?), (a_1, b_1, ?, ?), (a_1, ?, ?, d_1), (?, b_1, ?, d_1), (a_1, b_1, ?, d_1)\}.$$

Infatti si ha:


$(a_1, ?, ?, ?)$
$(?, b_1, ?, ?)$
$(a_1, ?, ?, d_1)$
$(a_1, b_1, ?, ?)$
$(?, b_1, ?, d_1)$
$(a_1, b_1, ?, d_1)$