

Relazione AI

Francesco Bettazzi

28 Marzo 2021

1 Obiettivo

Lo scopo del progetto è quello di implementare un programma per la propagazione delle probabilità condizionali per arbitrarie evidenze e query, assumendo di conoscere il Junction Tree di una rete Bayesiana su un universo U . Successivamente confrontarlo con il risultato ottenuto applicando la definizione di probabilità condizionale.

2 Implementazione

Di seguito i dettagli implementativi delle principali classi e funzioni presenti nel programma.

2.1 BeliefTable

È una classe che implementa una tabella di probabilità. Ha quindi associati una tabella e la lista delle variabili coinvolte in tale tabella. Fornisce un metodo *putEvidence()*, che fa il reset di alcuni valori nella tabella che cambiano in conseguenza della variabile su cui viene messa l'evidenza e del valore stesso dell'evidenza.

2.2 BayesianNetwork

È una classe che implementa una rete Bayesiana. La rete ha un nome, alcuni nodi e tiene traccia delle evidenze che sono state inserite. Ogni nodo ha una tabella di probabilità associata. La classe fornisce alcuni metodi: *addNode()*, che aggiunge un nodo alla rete Bayesiana dato il nome, la tabella associata e gli eventuali nodi genitori; *setEvidence()*, inserisce un'evidenza nella rete Bayesiana date le variabili e i corrispettivi valori di evidenza True/False; *getProbUniverse()*, ritorna la tabella di probabilità congiunta di tutte le variabili presenti nella rete; *getProb()*, date una serie di variabili, ritorna le tabelle di probabilità associate alle singole variabili in input.

2.3 JunctionTree

È una classe che implementa un Junction Tree. Ha un nome, una lista di clusters e tiene traccia delle evidenze che vengono inserite. Un cluster ha una lista di variabili associata e due tabelle di probabilità, una delle quali denominata “old_table”, essenziale per il message passing. Inoltre ha una lista di vicini e un attributo che identifica i separatori dai clusters veri e propri. Anch’essa fornisce i metodi *setEvidence()* e *getProb()* che hanno la stessa funzione delle rispettive nella classe BayesianNetwork (Sezione 2.2); i metodi *collectEvidence()* e *distributeEvidence()* per computare il message passing.

2.4 Altre funzioni

Nell’implementazione vengono usate altre funzioni relative alle operazioni sulle tabelle di probabilità.

2.4.1 *marginalize()*

È una funzione che prende in input una tabella e una variabile. Restituisce la tabella di probabilità della variabile in input.

2.4.2 *marginalizeMSG()*

È una funzione che prende in input una tabella e una lista di variabili. Restituisce la tabella di probabilità marginalizzata rispetto alle variabili prese in input.

2.4.3 *multiplyBTs()*

È una funzione che prende in input due tabelle di probabilità e ne restituisce il prodotto.

2.4.4 *divideBTs()*

È una funzione che prende in input due tabelle di probabilità e ne restituisce il quoziente.

3 Risultati e conclusione

I calcoli sono stati eseguiti su modelli standard di reti bayesiane, in particolare sui modelli *asia*, *cancer* e *sprinkler*. Le reti e i junction trees sono stati costruiti manualmente insieme alle relative tabelle di probabilità. Di seguito riporto i risultati dei test su evidenze inserite in modo arbitrario, riportando per ogni modello sia le evidenze inserite che le probabilità di tutte le variabili presenti nel modello.

ASIA: (evidences: [['asia', False], ['smoke', True], ['dysp', True]])

variable	prob	exact_prob
asia	[0.0, 100.0]	[0.0, 100.0]
bronc	[86.16, 13.84]	[86.0, 14.0]
dysp	[100.0, 0.0]	[100.0, 0.0]
either	[18.03, 81.97]	[18.61, 81.39]
lung	[17.13, 82.87]	[17.08, 82.92]
smoke	[100.0, 0.0]	[100.0, 0.0]
tub	[1.0, 99.0]	[1.71, 98.29]
xray	[21.77, 78.23]	[22.31, 77.69]

CANCER: (evidences: [['pollution', False], ['smoker', True]])

variable	prob	exact_prob
cancer	[3.0, 97.0]	[3.0, 97.0]
dyspnea	[31.05, 68.95]	[31.05, 68.95]
pollution	[0.0, 100.0]	[0.0, 100.0]
smoker	[100.0, 0.0]	[100.0, 0.0]
xray	[22.1, 77.9]	[22.1, 77.9]

SPRINKLER: (evidences: [['cloudy', True], ['wet_grass', True]])

variable	prob	exact_prob
cloudy	[100.0, 0.0]	[100.0, 0.0]
rain	[97.56, 2.44]	[97.56, 2.44]
sprinkler	[90.91, 9.09]	[90.91, 9.09]
wet_grass	[100.0, 0.0]	[100.0, 0.0]

Conclusione Si può notare che per reti piccole non si riscontra alcuna differenza tra i risultati ottenuti con il message passing e quelli derivati dalla formula di probabilità condizionale (che è intrattabile). Per reti un po' più consistenti si nota una leggera differenza tra i due metodi.