RELAZIONE SULL'ELABORATO D'ESAME

Intelligenza artificiale (2020/2021)

Matteo Goldin

7006216

1. SCOPO DELL'ELABORATO

Lo scopo dell'elaborato è quello di realizzare un algoritmo per l'apprendimento bayesiano di parametri in una rete bayesiana a partire da un dataset, conoscendo già la struttura di tale rete. Inoltre vogliamo calcolare la divergenza di Jensen-Shannon tra le due distribuzioni di probabilità, quella nota e quella appresa, al variare del numero di campioni di cui è costituito il dataset.

2. INTRODUZIONE

2.1 RETI BAYESIANE

Una rete bayesiana è un modello grafico probabilistico che, attraverso un DAG (grafo aciclico e diretto), rappresenta un insieme di variabili aleatorie (i nodi del grafo) con le loro dipendenze condizionali (gli archi del grafo). La rete bayesiana esprime la distribuzione di probabilità congiunta di un insieme di n variabili aleatorie. Tale distribuzione di probabilità congiunta è data da:

$$p(x) = \prod_{i=1}^{n} p(x_i|pa_i)$$

Dove $p(x_i|pa_i)$ sono le distribuzioni locali di probabilità.

2.2 PARAMETRI

Nel caso di questo elaborato tutte le variabili aleatorie che compongono la rete bayesiana hanno una distribuzione multinomiale. Questo significa che ciascuna v.a. X_i è discreta e può assumere r_i possibili valori. Ciascuna distribuzione locale di probabilità è una collezione di distribuzioni multinomiali, una per ogni configurazione dei padri di X_i. Ciascuna di queste distribuzioni rappresenta un parametro:

$$p(x_i = k|pa_i = j) = \theta_{ijk}$$

L'insieme dei parametri di X_i può quindi essere espresso come $\left(\left(\theta_{ijk}\right)_{k=1}^{r_i}\right)_{j=1}^{q_i}$ (dove q_i è il numero di configurazioni dei padri di X_i).

3. L'ALGORITMO

3.1 IMPLEMENTAZIONE

L'algoritmo per l'apprendimento dei parametri in una rete bayesiana utilizzato è l'approccio bayesiano descritto in (Heckerman 1997) ed è stato implementato in linguaggio python. La funzione utilizzata per il campionamento (*forward_sample)* appartiene alla libreria *pgmpy*. Le restanti classi e funzioni sono state sviluppate in maniera autonoma.

Sono state create tre classi:

- **BayesianNetwork**: che rappresenta la rete bayesiana.
- **Node**: che rappresenta ciascun nodo che compone la rete
- **Parameter**: che rappresenta ciascuno dei parametri da apprendere

Inoltre sono state implementate due funzioni:

- **BayesianLearning**: che, presi in ingresso una rete e un dataset, realizza l'apprendimento dei parametri.
- **JS_divergence**: che, prese in ingresso due reti bayesiane, calcola la divergenza di Jensenn-Shannon.

3.2 FUNZIONAMENTO

Per convenienza definiamo il vettore di parametri $\theta_{ij} = (\theta_{ij1} \dots \theta_{ijr_i})$.

L'algoritmo di apprendimento si basa su due assunzioni. La prima assunzione è che il dataset sia completo. La seconda assunzione è che i vettori di parametri θ_{ij} siano mutualmente indipendenti. Di conseguenza possiamo dire che:

$$p(\theta|D) = \prod_{i=1}^{n} \prod_{j=1}^{q_i} p(\theta_{ij}|D)$$

Possiamo quindi aggiornare ciascun vettore θ_{ij} in maniera indipendente. Assumendo che ciascuno di tali vettori abbia come distribuzione a priori $Dir(\theta_{ij}|\alpha_{ij1}, \dots, \alpha_{ijr_i})$ otteniamo la distribuzione a posteriori:

$$p(\theta_{ij}|D) = Dir(\theta_{ij}|\alpha_{ij1} + N_{ij1}, \dots, \alpha_{ijr_i} + N_{ijr_i})$$

Dove N_{ijk} è il numero di volte in cui $x_i = k$ e $pa_i = j$ nel dataset.

Questo ci permette di ricavare la probabilità di $p(x_{N+1}|D)$ con $x_i = k$ e $pa_i = j$ (con k e j che dipendono da i):

$$p(x_{N+1}|D) = \int \prod_{i=1}^{n} \theta_{ijk}\, p(\theta|D)\, d\theta = \prod_{i=1}^{n} \int \theta_{ijk}\, p(\theta_{ij}|D)\; d\theta_{ij} = \prod_{i=1}^{n} E_{p(\theta_{ij}|D)}(\theta_{ijk})$$
$$= \prod_{i=1}^{n} \frac{\alpha_{ijk} + N_{ijk}}{\alpha_{ij} + N_{ij}}$$

Dove $\alpha_{ij} = \sum_{k=1}^{r_i} \alpha_{ijk}$ e $N_{ij} = \sum_{k=1}^{r_i} N_{ijk}$

3.3 DIVERGENZA DI JENSEN-SHANNON

La divergenza di Jensen-Shannon è uno strumento che permette di confrontare due distribuzioni di probabilità e di quantificarne la differenza. Questo avviene assegnando uno score che va da 0 (identiche) ad

1 (massima differenza) se si utilizza il logaritmo in base 2. Rispetto alla divergenza di Kullback-Leibler essa è simmetrica.

La divergenza di Jensen-Shannon tra due distribuzioni di probabilità p e q si calcola come:

$$JS(p,q) = \sum_U p(U) \log \frac{p(U)}{\frac{p(U)+q(U)}{2}} + \sum_U q(U) \log \frac{q(U)}{\frac{p(U)+q(U)}{2}}$$

4. ANALISI DEI RISULTATI

Per testare l'algoritmo sono state utilizzate le reti *asia.bif* e *cancer.bif* disponibili su https://github.com/ncullen93/pyBN/tree/master/data. A partire da tali reti sono stati generati dei dataset di dimensione crescente. A partire da ciascun dataset sono stati appresi i parametri, con l'algoritmo presentato in precedenza, usando come priors pseudo-counts unitari, e infine è stata calcolata la divergenza di Jensen-Shannon tra la distribuzione nota e quella appresa.

In particolare sono stati generati dataset di dimensione 10, 100, 200, 500, 1000, 2000 e 5000. Per ciascuna dimensione sono stati generati più dataset, in numero maggiore per le dimensioni più piccole, in modo da poter calcolare il valor medio della divergenza per ognuno di questi valori.

Dai grafici in figura possiamo vedere che il comportamento della curva, che mostra il valore della divergenza al variare della dimensione del dataset, è quello auspicato, ovvero che la divergenza si riduce al crescere del numero di campioni del dataset.

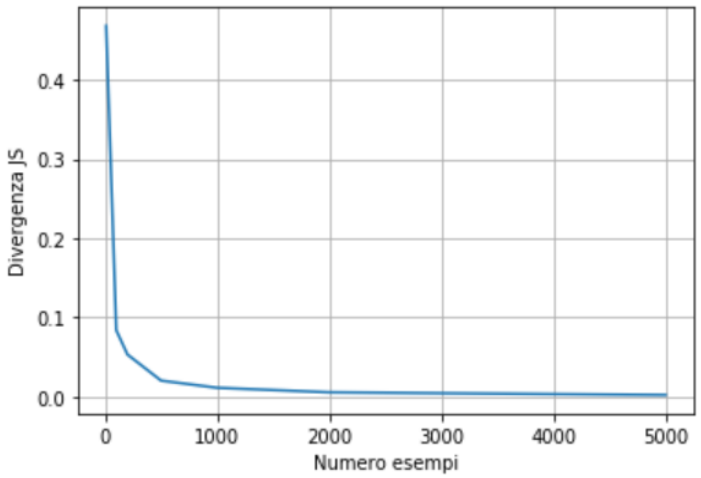

Figura 1: Rete asia.bif

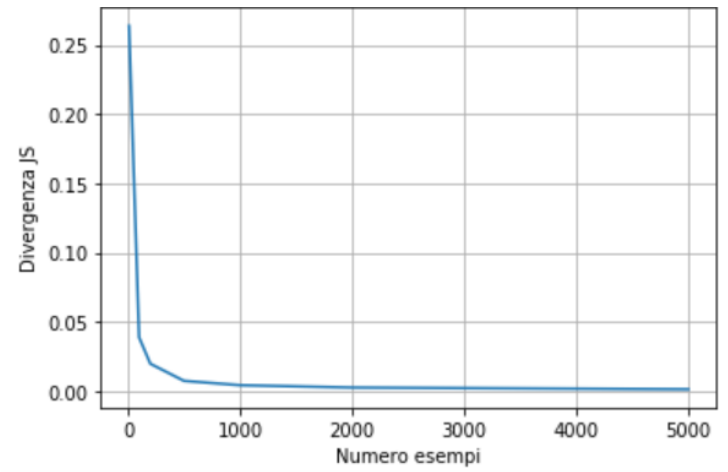

Figura 2: Rete cancer.bif

5. FONTI

- Heckerman 1997
- https://machinelearningmastery.com/divergence-between-probability-distributions/
- https://it.wikipedia.org/wiki/Rete_bayesiana