UNIVERSITÀ
DEGLI STUDI
FIRENZE

Apprendimento dei Parametri in Reti Bayesiane

Analisi della Convergenza tramite Divergenza di Jensen-Shannon

Simone Suma

Anno Accademico 2024/2025

1. Introduzione e Scopo del Progetto

Le Reti Bayesiane rappresentano uno strumento fondamentale nell'ambito dei modelli grafici probabilistici, consentendo di modellare relazioni complesse e di ragionare in condizioni di incertezza. Una delle fasi cruciali nel loro utilizzo pratico è l'apprendimento dei parametri a partire da dati osservati. Dato un grafo aciclico diretto (DAG) che definisce la struttura delle dipendenze, il compito consiste nello stimare le distribuzioni di probabilità condizionata (CPT) per ciascun nodo.

L'obiettivo di questo progetto è duplice. In primo luogo, implementare un algoritmo robusto per l'apprendimento di questi parametri, basato su un approccio Bayesiano con priori non informative (smoothing di Laplace). In secondo luogo, verificare sperimentalmente una delle proprietà teoriche più importanti dell'apprendimento statistico: la convergenza della stima al modello reale all'aumentare dell'evidenza empirica.

Per raggiungere tale scopo, in questa relazione verrà prima descritta la metodologia adottata (Sezione 2), includendo la rete di riferimento, la generazione dei dati e la metrica di valutazione. Successivamente, verranno presentati e analizzati i risultati sperimentali (Sezione 3 e 4), culminanti nella discussione della curva di apprendimento ottenuta.

2. Metodologia

L'esperimento è stato condotto seguendo una pipeline ben definita per garantire la riproducibilità dei risultati.

2.1 Rete di Riferimento

Come ground truth è stata utilizzata la rete Bayesiana **Andes**, un modello complesso utilizzato per la valutazione delle conoscenze degli studenti in ambito fisico. Questa rete, caricata dal file `andes.bif`, è significativamente più grande e strutturata rispetto a modelli più semplici, fornendo un benchmark più impegnativo per il nostro algoritmo. Questa rete di riferimento è indicata come p.

2.2 Generazione dei Dati

Per ogni dimensione del campione n testata (da 50 a 10.000), è stato generato un dataset di training tramite **campionamento ancestrale** a partire dalla rete di riferimento p. Questo processo assicura che i dati riflettano la distribuzione di probabilità vera.

2.3 Apprendimento dei Parametri

Per ogni dataset generato, i parametri della rete (le CPT) sono stati stimati da zero utilizzando un approccio Bayesiano. Specificamente, si è utilizzato lo **smoothing di Laplace** (o "add-one smoothing"), che corrisponde all'uso di una prior di Dirichlet con pseudo-conteggi unitari. Questo approccio previene l'assegnazione di probabilità nulle a eventi non osservati nel campione.

2.4 Misura dell'Errore

La "distanza" tra la distribuzione vera p e quella appresa q_n è stata quantificata calcolando la media della **Divergenza di Jensen-Shannon (JS)** su tutte le distribuzioni condizionali della rete. La formula della divergenza JS è:

$$JS(p, q_n) = \frac{1}{2} \sum_U p(U) \log \frac{p(U)}{M(U)} + \frac{1}{2} \sum_U q_n(U) \log \frac{q_n(U)}{M(U)}$$

dove $M(U) = \frac{p(U)+q_n(U)}{2}$. Un valore di JS pari a 0 indica una corrispondenza perfetta.

3. Risultati Sperimentali

L'esperimento è stato eseguito per diverse dimensioni del campione n. Per ciascuna dimensione, è stato calcolato l'errore medio tra la rete vera e quella appresa tramite la Divergenza JS. I risultati numerici sono riportati nella Tabella 1.

Numero di Campioni (n)	**Divergenza JS Media**
50	0.0454
100	0.0354
250	0.0206
500	0.0135
1000	0.0081
2500	0.0047
5000	0.0027
10000	0.0016

Tabella 1: Errore di stima (Divergenza JS) in funzione della dimensione del campione.

Questi dati sono visualizzati nella curva di apprendimento in Figura 1. Il grafico utilizza una scala logaritmica per l'asse X per rappresentare in modo efficace l'ampio intervallo di dimensioni del campione testate.

Dal grafico si evince una relazione inversa, fluida e monotona tra il numero di campioni e l'errore di stima. Il calo dell'errore è particolarmente marcato nella fase iniziale (da 50 a 1000 campioni), indicando che i maggiori guadagni in termini di accuratezza si ottengono con i primi dati a disposizione. Superata tale soglia, ulteriori aumenti nella dimensione del campione portano a miglioramenti più marginali, con la stima che tende asintoticamente al modello vero (errore zero), mostrando una convergenza robusta e consistente.

4. Dettagli Implementativi

Il progetto è stato sviluppato in Python 3, con un'attenzione particolare alla modularità del codice, alla scelta di strutture dati efficienti e alla robustezza degli algoritmi.

4.1 Struttura del Codice

Il codice è suddiviso in tre file principali per separare le responsabilità:

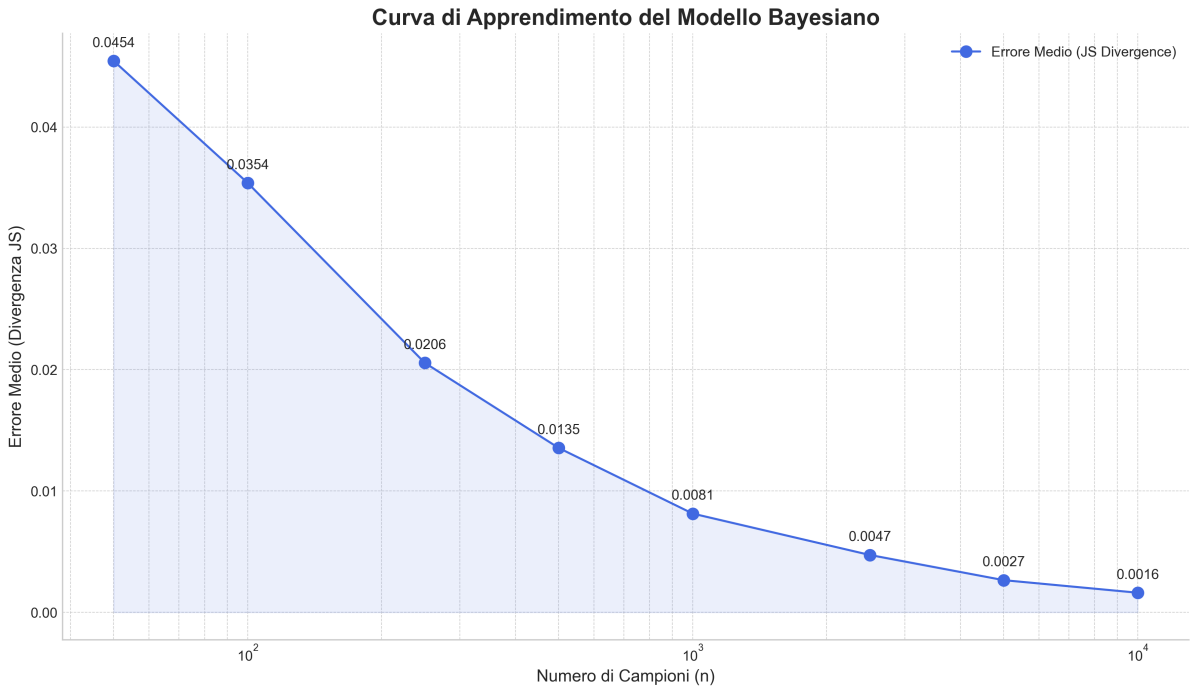

Figura 1: Curva di apprendimento che mostra la Divergenza JS media in funzione del numero di campioni.

- `rete_bayesiana.py`: Definisce le classi `Nodo` e `ReteBayesiana`. Questa astrazione permette di rappresentare la rete in modo orientato agli oggetti, incapsulando la logica di parsing e di generazione dei dati.
- `algoritmi.py`: Contiene le funzioni "pure" che implementano la logica matematica, come l'apprendimento dei parametri e il calcolo della divergenza. Questa separazione rende gli algoritmi indipendenti dalla specifica rappresentazione dei dati.
- `main.py`: Funge da orchestratore, gestendo il flusso dell'esperimento e la visualizzazione dei risultati.

4.2 Scelta delle Strutture Dati

La performance e la chiarezza del codice dipendono criticamente dalle strutture dati scelte:

- **Rappresentazione della Rete:** La rete è implementata come un dizionario che mappa i nomi dei nodi (stringhe) ai rispettivi oggetti `Nodo`. Questa scelta garantisce un accesso ai nodi in tempo costante O(1), essenziale per un'efficiente navigazione del grafo.
- **Tabelle di Probabilità Condizionata (CPT):** Le CPT sono state implementate tramite dizionari nidificati. Per i nodi con genitori, le chiavi del dizionario principale sono **tuple** che rappresentano le combinazioni degli stati dei genitori. Le tuple sono state preferite alle liste in quanto immutabili e quindi utilizzabili come chiavi di dizionario, garantendo un accesso diretto e performante alla distribuzione di probabilità corretta per una data evidenza.

4.3 Ordinamento Topologico

Per garantire il corretto funzionamento degli algoritmi di campionamento, è essenziale processare i nodi in un ordine in cui i genitori precedano sempre i figli. A tal fine, è stato implementato l'**algoritmo di Kahn**. Questa soluzione generica, basata sulla gestione dei gradi d'ingresso dei nodi, calcola l'ordine topologico per qualsiasi grafo aciclico diretto (DAG), rendendo il codice applicabile a qualsiasi rete valida e non solo a quella di test.

4.4 Campionamento Ancestrale

La generazione dei dati di training è stata realizzata tramite il metodo del **campionamento ancestrale**. L'algoritmo, implementato in `genera_campione`, opera come segue:

1. I nodi della rete vengono processati secondo l'ordine topologico.
2. Per ogni nodo, si seleziona la distribuzione di probabilità appropriata dalla sua CPT, accedendovi tramite i valori dei suoi nodi genitori, già campionati nei passi precedenti.
3. Viene estratto un valore casuale per il nodo corrente utilizzando la funzione `random.choices`, che permette di effettuare una scelta pesata basata sulla distribuzione di probabilità selezionata.

Questo processo assicura che ogni campione generato sia una realizzazione coerente e statisticamente valida della distribuzione di probabilità congiunta definita dalla rete.

4.5 Apprendimento Bayesiano

La funzione `impara_parametri` implementa lo smoothing di Laplace con una strategia a due fasi per garantire la robustezza.

- **Fase 1: Inizializzazione dei Conteggi.** Prima di analizzare i dati, la funzione precalcola tutte le possibili combinazioni degli stati dei nodi genitori utilizzando `itertools.product`. Per ciascuna combinazione, viene creata una struttura di conteggio in cui ogni possibile stato del nodo figlio è inizializzato a 1.
- **Fase 2: Aggiornamento e Normalizzazione.** Successivamente, il dataset viene iterato una sola volta. Per ogni campione, viene identificata la condizione dei genitori e il risultato del nodo figlio, e il contatore corrispondente viene incrementato. Infine, i conteggi vengono normalizzati per ottenere le probabilità stimate.

Questo approccio previene errori di tipo `KeyError` dovuti a combinazioni di evidenza non presenti nel dataset di training e garantisce che la CPT appresa abbia sempre la stessa struttura di quella originale.

5. Analisi e Conclusioni

L'esperimento condotto ha validato con successo l'ipotesi centrale del progetto: la precisione di un modello Bayesiano appreso dai dati converge verso il modello reale all'aumentare della quantità di evidenza disponibile. La curva di apprendimento, ottenuta sulla complessa rete

Andes, dimostra una relazione inversa chiara e monotona tra la dimensione del campione e l'errore di stima, misurato tramite la Divergenza di Jensen-Shannon. Questo risultato non è una mera constatazione numerica, ma la dimostrazione pratica di un principio fondamentale che sta alla base di gran parte dell'intelligenza artificiale moderna: la capacità di generalizzare e di costruire modelli accurati del mondo dipende intrinsecamente dalla ricchezza e dalla qualità dei dati.

Le scelte implementative si sono rivelate adeguate alla complessità del problema. L'adozione di un algoritmo di ordinamento topologico generico ha garantito la flessibilità del software, mentre la strategia di pre-inizializzazione dei conteggi nella fase di apprendimento ha assicurato la robustezza del modello, anche in presenza di combinazioni di eventi non osservate nei campioni più piccoli. L'uso dello smoothing di Laplace, in particolare, si è dimostrato essenziale per evitare il problema della probabilità nulla, un fenomeno che può rendere un modello probabilistico inutilizzabile.

È tuttavia fondamentale riconoscere i limiti del presente lavoro, che aprono la strada a interessanti sviluppi futuri. L'assunzione di una struttura della rete nota a priori è una semplificazione significativa; in scenari reali, la struttura stessa è spesso sconosciuta e deve essere appresa dai dati. Un'estensione naturale del progetto consisterebbe nell'implementare algoritmi di **apprendimento della struttura** (es. score-based o constraint-based) per scoprire autonomamente le dipendenze causali. Un'altra limitazione è l'assunzione di dati completi. Il mondo reale è disordinato e i dataset contengono spesso valori mancanti. L'integrazione di algoritmi come l'**Expectation-Maximization (EM)** per gestire l'incertezza derivante da dati incompleti rappresenterebbe un passo cruciale verso un'applicazione più realistica.

In conclusione, il progetto ha raggiunto pienamente i suoi obiettivi, fornendo non solo un'implementazione funzionante e robusta dell'apprendimento Bayesiano, ma anche una profonda comprensione pratica dei meccanismi di convergenza che governano i modelli statistici.

6. Riferimenti Bibliografici e Fonti

Riferimenti bibliografici

[1] D. Heckerman, *A Tutorial on Learning with Bayesian Networks*. Microsoft Research, Technical Report MSR-TR-95-06, 1995 (revised 1997).

[2] S. Russell, P. Norvig, *Intelligenza artificiale. Un approccio moderno*, 4ª ed., Pearson, 2022. (In particolare, la Sezione 13.4.1 per l'apprendimento con dati completi).

[3] N. Cullen, *pyBN GitHub Repository*: `https://github.com/ncullen93/pyBN`. La rete `andes.bif` è stata prelevata dalla directory `/data` di questo repository.