Multivariate Analysis and Statistical Learning Implementazione PC Algorithm

Authors: Alex Foglia, Tommaso Puccetti

Università degli Studi di Firenze

21/12/2018

Accenni teorici (1)

- Le reti Bayesiane possono essere rappresentate attraverso l'utilizzo di un grafo aciclico diretto, o **DAG**.
- Un DAG è un grafo diretto in cui non compaiono cicli, dove per ciclo si intende un qualunque cammino finito che, a partire da un nodo iniziale v termini in v.

Accenni teorici (2)

Sia $G = (V, E)$ un DAG su un insieme finito $X = \{X_v \forall v \in V\}$ di variabili casuali, allora:

$$\forall u, v \in V \text{ non adiacenti } | v \in nd(u) \Rightarrow u \perp\!\!\!\perp v | nd(u) - v$$

Dove $nd(u)$ è l'insieme dei nodi *non discendenti* di u, ossia tutti quei nodi u' per cui non esiste un cammino da u a u'.

PC-Algorithm

Dato un insieme di variabili con distribuzione di probabilità congiunta gaussiana, è possibile imparare il DAG sottostante al campione osservato attraverso l'utilizzo del **PC-Algorithm**.
Esso è composto da due sotto-funzioni che risolvono due diversi problemi:

1. La costruzione dello scheletro (o grafo morale)
2. La costruzione del DAG a partire da un dato scheletro

Pseudocodice: generazione dello scheletro

Algorithm 1: The PC-algorithm for the skeleton

Input: z-transform of estimated partial correlations, tuning parameter α
Output: Skeleton of CPDAG G, separation sets S (used later for directing the skeleton)

Form the complete undirected graph $\tilde{G}$ on the set $\{1, \ldots, p\}$;
$l = -1; G = \tilde{G}$;
repeat
  $l = l + 1$;
  repeat
    Select an ordered pair of adjacent variables i, j in G such that $|adj(i, G)\backslash\{j\}| \geq l$;
    repeat
      Choose $K \subseteq adj(i, G)\backslash\{j\}$ with $|K| = l$;
      if $\sqrt{n - |K| - 3}|Z(i, j \mid K)| \leq \Phi^{-1}(1 - \alpha/2)$ **then**
        Delete edge i, j;
        Denote this new graph by G;
        Save K in $S(i, j)$ and $S(j, i)$;
    until *edge i, j is deleted or all $K \subseteq adj(i, G)\backslash\{j\}$ with $|K| = l$ have been chosen* ;
  until *all ordered pairs of adjacent variables i and j, such that $|adj(i, G)\backslash\{j\}| \geq l$ and $K \subseteq adj(i, G)\backslash\{j\}$ with $|K| = l$, have been tested for conditional independence* ;
until *for each ordered pair of adjacent nodes i,j: $|adj(i, G)\backslash\{j\}| < l$* ;

Pseudocodice: costruzione del DAG

Algorithm 2: The PC-algorithm: extending the skeleton to a CPDAG

Input: Skeleton G of CPDAG, separation sets S

Output: CPDAG

forall *pairs of nonadjacent variables* i, j *with common neighbor* k **do**
  if $k \notin S(i,j)$ **then**
    Replace $i - k - j$ in Skeleton of G by $i \rightarrow k \leftarrow j$;
repeat
  R1 Orient $j - k$ into $j \rightarrow k$ whenever there is an arrow $i \rightarrow j$ such that i and k are nonadjacent;
  R2 Orient $i - j$ into $i \rightarrow j$ whenever there is a chain $i \rightarrow k \rightarrow j$;
  R3 Orient $i - j$ into $i \rightarrow j$ whenever there are two chains $i \rightarrow k \rightarrow j$ and $i \rightarrow l \rightarrow j$ such that k and l are nonadjacent;
until *no more orienting of undirected edges is possible by the rules* **R1** *to* **R3** ;

Librerie Utilizzate

Per implementare l'algoritmo abbiamo utilizzato le seguenti librerie:

- **itertools**: ci ha fornito la funzione combinations per generare coppie ordinate di archi
- **scipy**: ci ha fornito la funzione norm sia per generare dati secondo una normale, sia per approssimare la funzione di densità cumulativa della normale
- **math**: ci ha fornito operazioni matematiche basilari come la radice quadrata
- **networkx**: ci ha fornito la struttura dati grafo
- **matplotlib**: ci ha fornito la possibilità di disegnare il grafo in una finestra
- **pandas**: ci ha fornito le funzioni per leggere dati da file