Bayesian Data Analysis:

reti Bayesiane nel caso di studio car insurance

F.M.Stefanini

Dipartimento di Statistica, Informatica, Applicazioni
Università degli Studi di Firenze, Italia

Aprile-Luglio 2017

Istruzioni

La consegna consiste nel caricare su Moodle il file Rmd e il conseguente file html come cartella zippata, ad esempio "stefanini_2017.zip", quindi il cognome deve essere il nome del file zip.

Il caso di studio Insurance riguarda una collezione di variabili … "for evaluating car insurance risks."

The insurance data set contains the following 27 variables:

GoodStudent (good student): a two-level factor with levels False and True.
Age (age): a three-level factor with levels Adolescent, Adult and Senior.
SocioEcon (socio-economic status): a four-level factor with levels Prole, Middle, UpperMiddle and Wealthy.
RiskAversion (risk aversion): a four-level factor with levels Psychopath, Adventurous, Normal and Cautious.
VehicleYear (vehicle age): a two-level factor with levels Current and older.
ThisCarDam (damage to this car): a four-level factor with levels None, Mild, Moderate and Severe.
RuggedAuto (ruggedness of the car): a three-level factor with levels EggShell, Football and Tank.
Accident (severity of the accident): a four-level factor with levels None, Mild, Moderate and Severe.

MakeModel (car's model): a five-level factor with levels SportsCar, Economy, FamilySedan, Luxury and SuperLuxury.
DrivQuality (driving quality): a three-level factor with levels Poor, Normal and Excellent.
Mileage (mileage): a four-level factor with levels FiveThou, TwentyThou, FiftyThou and Domino.
Antilock (ABS): a two-level factor with levels False and True.
DrivingSkill (driving skill): a three-level factor with levels SubStandard, Normal and Expert.
SeniorTrain (senior training): a two-level factor with levels False and True.
ThisCarCost (costs for the insured car): a four-level factor with levels Thousand, TenThou, HundredThou and Million.
Theft (theft): a two-level factor with levels False and True.
CarValue (value of the car): a five-level factor with levels FiveThou, TenThou, TwentyThou, FiftyThou and Million.
HomeBase (neighbourhood type): a four-level factor with levels Secure, City, Suburb and Rural.
AntiTheft (anti-theft system): a two-level factor with levels False and True.
PropCost (ratio of the cost for the two cars): a four-level factor with levels Thousand, TenThou, HundredThou and Million.
OtherCarCost (costs for the other car): a four-level factor with levels Thousand, TenThou, HundredThou and Million.
OtherCar (other cars involved in the accident): a two-level factor with levels False and True.
MedCost (cost of the medical treatment): a four-level factor with levels Thousand, TenThou, HundredThou and Million.
Cushioning (cushioning): a four-level factor with levels Poor, Fair, Good and Excellent. Airbag (airbag): a two-level factor with levels False and True.
ILiCost (inspection cost): a four-level factor with levels Thousand, TenThou, HundredThou and Million.
DrivHist (driving history): a three-level factor with levels Zero, One and Many.

Riferimenti

Binder J, Koller D, Russell S, Kanazawa K (1997). "Adaptive Probabilistic Networks with Hidden Variables". Machine Learning, 29(2-3), 213-244.

Assegnazione

1.0 Caricare la libreria *bnlearn*, e poi *gRain* quindi *igraph*. Caricare la workspace *insuranceDF.RData*. Quante unità statistiche contiene?
Le variabili sono tutte oggetti fattore?
Quanti missing values ci sono in ciascuna variabile?

2.1 Effettuare l'apprendimento strutturale di una rete bayesiana impiegando hill climbing e score "bde". Rappresentare in una figura il grafo ottenuto usando *plot()*.
Per ogni nodo, stampare a schermo i suoi figli (hint: selezionare un sottodataframe ottenuto con *arcs()*).

2.2 Con la rete precedentemente ottenuta:
- Estrarre gli archi orientati (hint: usa *arcs()*). Quanti archi orientati contiene?
- Trova i nodi radice della rete, ovvero quelli che sono senza parenti ma con discendenti.
- Trova i nodi isolati, se ce ne sono (quelli senza parenti e senza figli).

3.0 Effettuare l'apprendimento strutturale di una rete bayesiana impiegando hill climbing con score "bde": il risultato NON DEVE contenere gli archi orientati sotto riportati (opzione *blacklist*). Nella tabella *qualsiasi altro* denota tutti i nodi meno quello già considerato.

Rappresentare in una figura il grafo ottenuto in output usando la funzione *plot()*. Quanti archi orientati contiene?

```
       from                to
1       <qualsiasi altro>   AGE
2       PropCost            <qualsiasi altro>
3       DrivHist            <qualsiasi altro>
```

4.1 Effettuare l'apprendimento strutturale di una rete bayesiana (da assegnare con nome mod.3) impiegando hill climbing con score "bic": assicurarsi che il risultato contenga gli archi orientati sotto riportati (opzione *whitelist*). Rappresentare in una figura il grafo ottenuto in output usando la funzione *plot()*. Quanti archi orientati contiene?

```
  from            to
"SeniorTrain"  "ThisCarDam"
"MedCost"      "RuggedAuto"
"SocioEcon"    "GoodStudent"
"Age"          "RiskAversion"
"Age"          "ThisCarDam"
"RuggedAuto"   "VehicleYear"
```

4.2 Esegui il codice seguente e cambiare posizione agli archi del dag con mouse.

```
> modello3 <- graph_from_data_frame(arcs(mod.3), directed = TRUE, vertices = NULL)
> plot(modello3)
> # esegui una istruzione per volta
> idbn <-tkplot(modello3)# non chiudere la finestra che appare !!!
> tk_set_coords(idbn, 2*tk_coords(idbn))
> # ora puoi trascinare i  nodi dove preferisci
> coordinate <- tk_coords(idbn, norm = FALSE)
> tk_close(idbn) ## questo comando la chiude automaticamente !!!
```

5.0 Confrontare le due reti ottenute con hill climbing con *blacklist* verso con *whitelist* usando *compare()* (settare il target come rete ottenuta con *blacklist*).

6.0 Considera il DAG ottenuto al punto (3). Stima i parametri di tale rete bayesiana con *bn.fit()*. Estrai la distribuzione di probabilità condizionata di "Theft" condizionata a "ThisCarCost = TenThou" e "ThisCarDam = Moderate".

7.1 Impiegando il DAG ottenuto come mod.2, stabilire se *PropCost* è indipendente da *RiskAversion* condizionatamente a *MakeModel, DrivQuality, Theft* usando un teorema di separazione per DAG (hint funzione dsep() della library "pcalg": bisogna trasformare la classe dell'oggetto mod.2, ad esempio in igraph e poi in graphNEL).

7.2 Con *simulate* genera via simulazione Monte Carlo una dataset di 10000 realizzazioni dalla rete ottenuta come mod.2. Stampa a schermo le distribuzioni marginali delle frequenze relative di ogni variabile per tale dataset simulato.

8.0 Dalla rete ottenuta come mod.2, si assuma di aver rilevato le evidenze seguenti:
Age sia "Adult";
RiskAversion sia "Psychopath";
AntiTheft sia "True";
MakeModel sia "SportsCar";
PropCost sia "HundredThou";

Crea un oggetto gRain basato su mod.2 e calcola la probabilità condizionata dell'evento CarValue = "FiftyThou" date le evidenze sopra elencate. Confronta tale valore di probabilità con il valore prima di inserire l'evidenza (prima del condizionamento).

Last May 2017